File System Extended Attributes in NFSv4

Marc Eshel
Manoj Naik

February 15, 2014

Problem Statement

- Do we need generic xattr support in NFS?
  - Yes, widely used by applications and well supported by most file systems
  - Since xattrs cannot be easily mapped to existing attributes in NFS, data loss occurs today if file with xattrs is copied over NFS
- So why aren't they supported?
  - xattrs are not part of any standard, so not portable

Why not use Named Attributes?

- Semantic mismatch with xattrs
  - Xattrs get/set operations typically atomic
  - Xattrs are small, named attrs are unbounded
  - Named attrs like files, different caching semantics
  - Disjoing namespaces
- Mapping one to another, at best, a compromise

What do we propose?

- Define a way to support xattrs
  - Opaque to clients and servers, but clear interfaces to get/set
  - Well-defined semantics
- Only user-specified xattrs
  - Discourage non-interoperable implementations

What are the options?

- Add to existing attribute bitmap
- Define new operations
- Modify existing operations

New Attributes

- `xattr_support (boolean)`
  - File system supports xattrs?
- `Maxxattrsize (int)`
  - Max size supported by file system
- `Xattrsize (int)`
  - Total size of all xattrs for a given file
- `Xattrall (boolean)`
  - Only for set, it can set all or subset of xattrs
  - No way to get single or subset of xattr

Option 1: Extend bitmap

- xattrs
  - Array of (name, value) pairs
- xattrnames
  - Array of names

```
typedef utf8str_cis     xattrname4;
typedef opaque          xattrvalue4<>;

struct xattr4 {
   xattrname4      xa_name;
   xattrvalue4     xa_value;
};
```

Other handling

- Caching, similar to other attributes like ACLs
- Value of length zero means delete the xattr
- Extensions to ACE Access Mask Attributes
  - Two new bitmask constants are proposed for the access mask field:
    - `const ACE4_READ_XATTRS = 0x00200000;`
    - `const ACE4_WRITE_XATTRS = 0x00400000;`

Issues

- Xattrs are treated collectively as a single file attribute
  - Single GETATTR/SETATTR request
- Not easy to differentiate between disparate xattrs
  - Can not Get a single xattr value
  - Can set a single or subset of xattr value

Option 2: New Operations

- GETXATTR, SETXATTR
  - Similar semantics as GETATTR/SETATTR but only used on xattrs
  - Allow listing, query value for a name, change, delete, etc.
  - Flexible, extensible
- Difficult to define consistency and atomicity semantics with other attributes
  - e.g. `time_metadata`

Option 3: Extend Operations

- GETATTR_PLUS, SETATTR_PLUS
  - Does everything that GETATTR/SETATTR does
  - Also, extend to allow more information in addition to bitmap
  - Main change to GETATTR request to include attribute name

Questions?

- Which option?
- Option 3 is the most complete alternative
  - It has the flexibility of handling a subset of the xattr list
  - It can provide atomicity with all other attributes

http://www.ietf.org/id/draft-naik-nfsv4-xattrs-00.txt