File System Extended Attributes in NFSv4

Manoj Naik
Marc Eshel

IETF 90
July 25, 2014

What has transpired so far...

- 10/2013: Mailing list discussion on xattrs
- 11/2013 IETF 88: Should we add xattrs in NFSv4?
  - Yes, widely used and well supported
  - Since xattrs cannot be easily mapped to existing attributes in NFS, data loss occurs today if file with xattrs is copied over NFS
- 3/2014 IETF 89: First draft
  - Extend existing bitmap4, propose other options
  - Consensus to define new operations for xattrs

What do we propose?

- Protocol enhancements to support xattrs
  - Clear interfaces for get/set
  - Well-defined semantics
- Only user-specified xattrs
  - Opaque to NFS clients and servers
  - Discourage non-interoperable implementations
  - Possible future extensions for namespaces, well-defined keys

Basic Operations

- Given a file, return a list of all of the file's assigned extended attribute keys (`listxattr`)
- Given a file and a key, return the corresponding value (`getxattr`)
- Given a file, a key, and a value, assign that value to the key (`setxattr`)
- Given a file and a key, remove that extended attribute from the file (`removexattr`)

Protocol Enhancements

- New RECOMMENDED attributes
  - Query xattr support
- New OPTIONAL operations
  - Get, set, list, remove xattrs
- Extensions to ACE Access Mask Attributes
  - New bitmask constants for the access mask field

New Attributes

- Extend bitmap4 for use with GETATTR

```
+--------------------+----+----------------+------+
|Name                | Id | Data Type      | Acc  |
+--------------------+----+----------------+------+
| maxxattrsize       | 82 | uint32_t       | R    |
| xattrsize          | 83 | uint32_t       | R    |
+--------------------+----+----------------+------+
```

  - `maxxattrsize`
    - Max size supported by file system
    - 0 if not supported
  - `xattrsize`
    - Total size of all xattrs for a given file
- No limits on number or size of individual xattrs

New Definitions

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


struct xattr4 {
   xattrname4     xa_name;
   xattrvalue4    xa_value;
};

const ACE4_GET_XATTRS   = 0x00200000;
const ACE4_SET_XATTRS   = 0x00400000;
```

GETXATTR: ARGUMENTS

```
enum getxattr_type4 {
      GETXATTR4_LIST = 0,
      GETXATTR4_ONE = 1,
      GETXATTR4_ALL = 2
};

union getxattr_args4 switch (getxattr_type4 ga_type) {
   case GETXATTR4_ONE:
        xattrname4    ga_name;
   default:
        void;
};

struct GETXATTR4args {
     /* CURRENT_FH: file */
     getxattr_type4   ga_type;
     getxattr_args4   ga_args;
};
```

GETXATTR: RESULTS

```
union getxattr_res4 switch (getxattr_type4 gr_type) {
   case GETXATTR4_LIST:
        xattrname4    gr_names<>;
   case GETXATTR4_ONE:
        xattrvalue4   gr_value;
   case GETXATTR4_ALL:
        xattr4        gr_xattrs<>;
};

union GETXATTR4res switch (nfsstat4 gr_status) {
   case NFS4_OK:
        getxattr_res4  gr_resok4;
   default:
        void;
};
```

SETXATTR: ARGUMENTS

```
enum setxattr_type4 {
        SETXATTR4_CREATE      = 0,
        SETXATTR4_REPLACE     = 1,
        SETXATTR4_DELETE      = 2,
        SETXATTR4_REPLACE_ALL = 3,
        SETXATTR4_DELETE_ALL  = 4
};
union setxattr_args4 switch (setxattr_type4 sa_type) {
   case SETXATTR4_CREATE:
   case SETXATTR4_REPLACE:
   case SETXATTR4_REPLACE_ALL:
        xattr4      sa_xattrs<>;
   case SETXATTR4_DELETE:
        xattrname4  sa_xattrnames<>;
   case SETXATTR4_DELETE_ALL:
        void;
};
struct SETXATTR4args {
        /* CURRENT_FH: file */
        setxattr_args4   sa_args;
};
```

SETXATTR: RESULTS

```
union setxattr_res4 switch (setxattr_type4 sr_type) {
   case SETXATTR4_CREATE:
   case SETXATTR4_REPLACE:
   case SETXATTR4_DELETE:
        nfsstat4   sr_res<>;
   case SETXATTR4_REPLACE_ALL:
   case SETXATTR4_DELETE_ALL:
        void;
};

union SETXATTR4res switch (nfsstat4 sr_status) {
   case NFS4_OK:
        void;
   default:
        setxattr_res4 sr_array;
};
```

Caching and Delegations

- Caching behavior similar to other attributes (not data)
- SETXATTR also modifies “change” attribute
- Clients without delegations
  - can cache (unmodified) xattrs, validate using change attribute
  - must write-through changes (synchronously), may need to wait for delegation to be recalled
- Owner of read/write delegations
  - can cache (modified) xattrs
  - respond with new “change” value to CB_GETATTR

Questions?

http://tools.ietf.org/html/draft-naik-nfsv4-xattrs-01