

Network File System Version 4
Internet-Draft
Intended status: Standards Track
Expires: 14 December 2025

R. Macklem
FreeBSD
12 June 2025

POSIX Draft ACL support for Network File System Version 4, Minor Version
2
draft-rmacklem-nfsv4-posix-acls-12

Abstract

This document describes a potential protocol extension involving the addition of four new attributes to be used by servers to provide support for POSIX ACLs. The term POSIX ACLs refers to the ACL component of the Portable Operating System Interface (POSIX) 1003.1e draft 17 document for which sponsorship was withdrawn in January 1998. Although the draft POSIX standard that describes these ACLs was never ratified, several POSIX-based operating systems, such as Linux, Solaris and FreeBSD include support for them. The NFS Version 4 (NFSv4) ACLs described in Network File System (NFS) Version 4 Minor Version 1 henceforth referred to as NFSv4 ACLs, use a different model and attempts to map between the ACLs of these two models have not been completely successful. In order to adequately support POSIX ACLs, this document proposes four new attributes that may optionally be used by an NFS Version 4, minor version 2 (NFSv4.2) server to support ACLs that conform to the aforementioned POSIX 1003.1e draft 17.

Note

This note is to be removed before publishing as an RFC.

Discussion of this draft occurs on the NFSv4 working group mailing list (nfsv4@ietf.org), archived at <https://mailarchive.ietf.org/arch/browse/nfsv4/>. Working Group information is available at <https://datatracker.ietf.org/wg/nfsv4/about/>.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 14 December 2025.

Copyright Notice

Copyright (c) 2025 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	3
2. Requirements Language	4
3. Protocol Extension Considerations	4
4. XDR definitions for new attributes	4
5. POSIX ACL Considerations	7
6. POSIX ACL vs NFSv4 ACL Considerations	8
7. ACL Related Attribute Atomicity	9
8. OPTIONAL New Attributes - List and Definition References . .	10
9. Definitions of new optional attributes	10
9.1. Attribute 89: acl_trueform	11
9.1.1. Rationale	11
9.2. Attribute 90: acl_trueform_scope	11
9.2.1. Rationale	11
9.3. Attribute 91: posix_default_acl	12
9.3.1. Rationale	13
9.4. Attribute 92: posix_access_acl	13
9.4.1. Rationale	14
10. Security Considerations	14
11. IANA Considerations	14
12. Implementation Status	14
12.1. FreeBSD NFS server and client	14
12.2. Linux kernel patch for the NFSv4 client and server . . .	15
13. References	15
13.1. Normative References	15
13.2. Informational References	16

Acknowledgments	17
Author's Address	17

1. Introduction

In response to the very different over-the-wire formats, attempts have been made to map between these two sorts of ACLs. However, because of the large number of semantic differences, implementation experience with mapping between NFSv4 and POSIX ACLs has not been completely successful. For example, if a NFSv4 ACL is applied to a server file via SETATTR on a server that stores POSIX ACLs and then retrieved via GETATTR/READDIR, the ACL will often not be the same, since the mapping algorithm cannot do an exact mapping between them. The expired IETF draft [Eriksen] explains the mapping algorithms. A server has the option of choosing to use these mapping algorithms and/or support the new attributes proposed by this document to set/get the POSIX ACLs.

In order to provide better support for POSIX ACLs, this document proposes four new attributes as an extension of NFSv4.2 which can be used by SETATTR/OPEN/CREATE and GETATTR/READDIR to handle POSIX ACLs. If a server chooses to support either the `acl_trueform` or `acl_trueform_scope` attributes, it MUST support both of them for all file objects on the server. If a server chooses to support either the `posix_default_acl` or `posix_access_acl` attributes for a file system, it MUST support both of these attributes for the file system and the `acl_trueform` and `acl_trueform_scope` for all file objects on the server. If the server chooses to support `posix_default_acl` and `posix_access_acl` for a file system, it MUST support the `mode/mode_umask` attributes for the file system.

If this extension is implemented, the VERIFY/NVERIFY operations SHOULD be able to compare the `acl_trueform` and `acl_trueform_scope` attributes. However, the VERIFY/NVERIFY operations MAY return NFS4ERR_INVAL when comparison of the `posix_default_acl` and/or `posix_access` are requested.

If the NFSv4 server does support comparison of the `posix_default_acl` or `posix_access_acl` attributes for VERIFY/NVERIFY, the following rules SHOULD be applied to the comparison:

- * If possible, the comparison should be done using the uid/gid that maps from the who field.
- * For two ACLs with the same set of ACEs but in different order, the ACLs SHOULD be considered the same.

The semantics relevant to the implementation of POSIX ACLs by the server are described in [Gr^端nbacher]. [RFC8275] describes a new attribute that is an extension to NFSv4.2 related to POSIX ACLs called `mode_umask`. A server that chooses to support the new attributes described in this document MUST also support the `mode_umask` attribute as described in [RFC8275]. In addition, issues related to the over-the-wire format of POSIX ACLs and the interactions among the various new attributes and with existing attributes as dealt with in this document.

For the purpose of this extension, NFSv4 AUDIT/ALARM ACEs are considered to be logically separate from ALLOW/DENY ACEs. These AUDIT/ALARM ACEs are unaffected by this extension. Deletion of a `dacl` does not imply any changes to the AUDIT/ALARM ACEs. For a server that supports AUDIT/ALARM ACEs for a file system, the server SHOULD support the `sacl` attribute to set/acquire them.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Protocol Extension Considerations

This document presents an extension to minor version 2 of the NFSv4 protocol as described in [RFC8178]. It describes new OPTIONAL features. NFSv4.2 servers and clients implemented without knowledge of this extension will continue to interoperate with clients and servers that are aware of the extension (whether or not they support it).

Note that [RFC7862] does not define NFSv4.2 as non-extensible, so [RFC8178] treats it as an extensible minor version. This Standards Track RFC extends NFSv4.2 but does not update [RFC7862] or [RFC8178].

4. XDR definitions for new attributes

This document contains the External Data Representation (XDR) [RFC4506] description of the new OPTIONAL ACL related attributes. The XDR description is embedded in this document in a way that makes it simple for the reader to extract into a ready-to-compile form. The reader can feed this document into the following shell script to produce the machine-readable XDR description of the flexible file layout type:

<CODE BEGINS>

```
#!/bin/sh
grep '^ *///' $* | sed 's?^ */// ??' | sed 's?^ *///$??'
```

<CODE ENDS>

That is, if the above script is stored in a file called "extract.sh" and this document is in a file called "spec.txt", then the reader can do:

```
sh extract.sh < spec.txt > flex_files_prot.x
```

The effect of the script is to remove leading white space from each line, plus a sentinel sequence of "///".

The embedded XDR file header follows.

Note that the XDR code contained in this document depends on types from the NFSv4.1 and the nfs4_prot.x file [RFC5662]. This includes both nfs type utf8str_mixed, as well as the more generic type uint32_t.

<CODE BEGINS>

```
/// enum aclmodel4 {
///     ACL_MODEL_NFS4          = 1,
///     ACL_MODEL_POSIX_DRAFT   = 2,
///     ACL_MODEL_NONE          = 3
/// };
///
/// enum aclscope4 {
///     ACL_SCOPE_FILE_OBJECT    = 1,
///     ACL_SCOPE_FILE_SYSTEM    = 2,
///     ACL_SCOPE_SERVER         = 3
/// };
///
/// enum posixacetag4 {
///     POSIXACE4_TAG_USER_OBJ   = 1,
///     POSIXACE4_TAG_USER       = 2,
///     POSIXACE4_TAG_GROUP_OBJ  = 3,
///     POSIXACE4_TAG_GROUP      = 4,
///     POSIXACE4_TAG_MASK       = 5,
///     POSIXACE4_TAG_OTHER      = 6
/// };
///
/// typedef uint32_t          posixaceperm4;
///
/// /* Bit definitions for posixaceperm4. */
/// const POSIXACE4_PERM_EXECUTE    = 0x00000001;
/// const POSIXACE4_PERM_WRITE      = 0x00000002;
/// const POSIXACE4_PERM_READ       = 0x00000004;
///
/// struct posixace4 {
///     posixacetag4    tag;
///     posixaceperm4   perm;
///     utf8str_mixed   who;
/// };
///
/// %/*
/// % * New for POSIX ACL extension
/// % */
/// const FATTR4_ACL_TRUEFORM      = 89;
/// const FATTR4_ACL_TRUEFORM_SCOPE = 90;
/// const FATTR4_POSIX_DEFAULT_ACL  = 91;
/// const FATTR4_POSIX_ACCESS_ACL   = 92;
```

<CODE ENDS>

5. POSIX ACL Considerations

A POSIX ACL always has one ACE for each of `POSIXACE4_TAG_USER_OBJ`, `POSIXACE4_TAG_GROUP_OBJ` and `POSIXACE4_TAG_OTHER`. A setting of a POSIX ACL that does not have these three ACEs is invalid. If the ACL consists only of these three ACEs, it is referred to as a minimal POSIX ACL. A POSIX ACL may also have one or more `POSIXACE4_TAG_USER` and/or `POSIXACE4_TAG_GROUP` ACE(s). Such a POSIX ACL is referred to as an extended POSIX ACL and must have one `POSIXACE4_TAG_MASK` ACE as well. A POSIX access ACL defines permissions for a file object. A POSIX default ACL can only be associated with directory objects and is used for inheritance. The POSIX default ACL has no effect on access.

In the `who` value within the `posixace4` structure that appear in these new attributes, the field is interpreted as follows:

- * For ACEs whose tag field is `POSIXACE4_TAG_USER` or `POSIXACE4_TAG_GROUP` the `who` value is a UTF8-encoded Unicode string, that has the same format as a user or group as represented within other NFSv4 operations and designates the same entity. In these cases, the distinction between users and groups derives from the tag rather than a flag bit, as is done in NFSv4 ACLs. This is in contrast to how the corresponding structures are described in [Gr^端nacher], where numeric uids and gids are specified.
- * For ACEs whose tag field has other values, the `who` field is ignored by the receiver and there is no reason for the sender to set it to any particular value. As such, a zero length `who` string is appropriate.

For POSIX ACLs, setting of the low-order nine bits of mode can change the ACL and setting of the POSIX access ACL can change the low-order nine bits of mode. As such, the ordering of setting the attributes related to mode and POSIX ACLs is important. Therefore, in a manner similar to [RFC8881] Section 6.4.1.3, if the low-order nine bits of mode is being set via the `mode/mode_set_masked` attributes in the same `SETATTR` as `posix_access_acl` and/or `posix_default_acl` attributes, the setting of `mode/mode_set_masked` MUST be done before setting the POSIX ACL.

For [IEEE], when a new object is created in a directory that has a POSIX default ACL on it, the inherited ACL includes the intersection between the mode specified by the POSIX system call and the `posixaceperm4` fields of the POSIX default ACL. Therefore, to maintain compatible semantics with the POSIX draft, for NFSv4 operations that create new file objects (`OPEN/OPEN4_CREATE`, `CREATE`) in a directory that has a POSIX default ACL, the low-order nine bits

of the mode MUST be specified by `mode_umask` in the setable attributes for the operation. (See [RFC8275] for details on how `mode_umask` is used.) If the `posix_access_acl` and/or `posix_default_acl` are also specified in the setable attributes for the operation, the setting of these attributes MUST be done after setting `mode_umask` and performing any POSIX ACL inheritance.

6. POSIX ACL vs NFSv4 ACL Considerations

The model for the ACL(s) stored on a file object and used to determine file access is referred to as the true form.

For servers that support the `posix_access_acl` and `posix_default_acl` attributes for a file system, each file object will have ACL(s) of one true form at any given time. For servers that return a `acl_trueform_scope` attribute value other than `ACL_SCOPE_FILE_OBJECT`, the true form MUST remain uniform for all file objects in the file system (or file server for a scope of `ACL_SCOPE_SERVER`). However, for servers that return a `acl_trueform_scope` attribute value of `ACL_SCOPE_FILE_OBJECT`, the value of `acl_trueform` can change, due to a `SETATTR` operation that sets any of the `acl/dacl/posix_access_acl/posix_default_acl` attributes. For servers that return a `acl_trueform_scope` attribute value of `ACL_SCOPE_FILE_OBJECT`, there might be different `acl_trueform` values for different file objects in the file system.

The low-order nine bits of mode SHOULD be synchronized with the true form ACL for the file object. If the true form is `ACL_MODEL_NFS4`, synchronization is described in [RFC8881]. If the true form is `ACL_MODEL_POSIX_DRAFT`, synchronization is described in [Grønhaugen]. If the true form is `ACL_MODEL_NONE`, there is no ACL to synchronize with and the low-order nine bits of mode are used to control access to the file object.

For servers configured to return a `acl_trueform_scope` attribute value of `ACL_SCOPE_FILE_OBJECT` and not for any others, the following apply:

- * The server MUST return a value of `ACL_MODEL_NONE` for `acl_trueform` if there is no true form ACL(s) associated with the file object.
- * The server MUST return a zero length `posixace4` array for `posix_default_acl` and `posix_access_acl` if the value of `acl_trueform` for the file object is not `ACL_MODEL_POSIX_DRAFT`.

- * Using SETATTR to set a zero length posixace4 array for posix_access_acl (for a file object with a true form ACL of ACL_MODEL_POSIX_DRAFT) will delete the true form ACL(s) from the file object and revert it to ACL_MODEL_NONE. If a server cannot do this, it MUST reply NFS4ERR_INVALID for the SETATTR.
- * Using SETATTR to set the dacl or acl attribute to any value will result in the object's true form being set to ACL_MODEL_NFS4. In addition, if the object's acl model had been ACL_MODEL_POSIX_DRAFT, any existing POSIX default and access true form ACL(s) are deleted. If a server cannot do this, it MUST reply NFS4ERR_INVALID for the SETATTR. For a client to set ALARM/AUDIT ACES without causing any change in a file object's true form, it will need to perform a SETATTR of the sacl attribute and not the acl attribute.
- * Using SETATTR to set a posix_default_acl or posix_access_acl of non-zero length posixace4 array (for a file object with a true form ACL not ACL_MODEL_POSIX_DRAFT) will result in the object's acl model being set to ACL_MODEL_POSIX_DRAFT. In addition, if the object's acl model had been ACL_MODEL_NFS4, any dacl (all ALLOW/DENY ACES) will be deleted from the stored ACL. If a server cannot do this, it MUST reply NFS4ERR_INVALID for the SETATTR. As noted above, any AUDIT/ALARM ACES stored for the file object are not affected.

For all other configurations of acl_trueform_scope and a acl_trueform of ACL_MODEL_POSIX_DRAFT, the server MUST always return at least a minimal POSIX ACL for posix_access_acl, if that attribute is supported for the file object's file system. A server that is configured for a acl_trueform_scope other than ACL_SCOPE_FILE_OBJECT MUST NOT support the posix_default_acl and posix_access_acl unless the true form for the file system is ACL_MODEL_POSIX_DRAFT. If a client does a SETATTR of a zero length posixace4 array for posix_access_acl and the acl_trueform_scope is anything other than ACL_SCOPE_FILE_OBJECT, the server MUST reply NFS4ERR_INVALID.

For all acl_trueform_scope configurations, if the acl/dacl attribute(s) are specified in the same settable attributes bitmap as the posix_default_acl/posix_access_acl attribute(s), the server MUST reply NFS4ERR_INVALID.

7. ACL Related Attribute Atomicity

For servers that choose to implement the extensions described in this document, the following semantics with respect to ACL related attributes SHOULD be implemented.

To ensure a reply to operations that return attributes (GETATTR/READDIR) provide coherent results when multiple ACL related attributes (acl/dacl/posix_access_acl/posix_default_acl/acl_trueform/mode/mode_set_masked) are acquired, the server SHOULD acquire the reply values for these attributes atomically with respect to any SETATTR of any of these attributes. For this purpose, atomically means that no SETATTR of any of the above ACL related attributes can be performed during acquisition of the attribute values for a given file object.

8. OPTIONAL New Attributes - List and Definition References

The list of New OPTIONAL attributes appears in Table 1. The meaning of the columns of the table are:

Name: The name of the attribute.

Id: The number assigned to the attribute.

Data Type: The XDR data type of the attribute.

Acc: Access allowed to the attribute. R means read-only. RW means read/write.

Defined in: The section of this specification that describes the attribute.

Name	Id	Data Type	Acc	Defined in:
acl_trueform	89	aclmodel4	R	Section 9.1
acl_trueform_scope	90	aclscope4	R	Section 9.2
posix_default_acl	91	posixace4<>	RW	Section 9.3
posix_access_acl	92	posixace4<>	RW	Section 9.4

Table 1

9. Definitions of new optional attributes

9.1. Attribute 89: `acl_trueform`

This attribute is a read-only attribute that describes which ACL model (NFSv4 vs POSIX) is used for ACL(s) stored on the file object (its true form) on the NFSv4.2 server. The value of this attribute can also be `ACL_MODEL_NONE` to indicate that no true form ACL is stored for this file object.

It is a per-file system object attribute.

9.1.1. Rationale

For a server that returns `ACL_SCOPE_FILE_OBJECT` for `acl_trueform_scope`, this attribute can be acquired in the same `GETATTR` as other ACL related attributes (`acl/dacl/posix_access_acl/posix_default_acl`), so that the client knows which ACL attribute(s) are for the true form. For a server that returns a value other than `ACL_SCOPE_FILE_OBJECT` for `acl_trueform_scope`, this attribute need only be acquired once per file system (or for the entire mount). The reply value can then be used by the client to determine whether to `GETATTR/READDIR` of the `acl/dacl` attribute(s) or the `posix_default_acl/posix_access_acl` attribute(s) to avoid the server doing mapping between the true form and the requested form for the attribute.

9.2. Attribute 90: `acl_trueform_scope`

Although the `acl_trueform` attribute is defined as a per-file system object attribute, some servers will be configured to provide the same `acl_trueform` reply for all file objects on either a file system or all file objects that share the same server owner (ie. per server). This attribute is a read-only attribute that describes what the actual scope of the `acl_trueform` attribute is for the file server. It MUST be the narrowest scope of all the file systems exported by the file server. Note that `ACL_SCOPE_FILE_OBJECT` is narrower than `ACL_SCOPE_FILE_SYSTEM` and `ACL_SCOPE_FILE_SYSTEM` is narrower than `ACL_SCOPE_FILE_SERVER`.

It is a per server attribute.

9.2.1. Rationale

For a value of `ACL_SCOPE_FILE_OBJECT`, the `acl_trueform` attribute can be acquired in the same `GETATTR` as other ACL related attributes (`acl/dacl/posix_access_acl/posix_default_acl`), so that the client knows which ACL attribute(s) are for the true form. For a value other than `ACL_SCOPE_FILE_OBJECT`, the `acl_trueform` attribute allows the client to choose whether to use the `acl/dacl` attribute or the

posix_access_acl/posix_default_acl attributes when doing GETATTR/SETATTR/REaddir/OPEN/CREATE for servers that choose to support both the acl/dacl and posix_default_acl/posix_access_acl attributes for a file system.

9.3. Attribute 91: posix_default_acl

This attribute specifies the POSIX default ACL for a directory.

For the posixacetag4 values of POSIXACE4_TAG_USER_OBJ, POSIXACE4_TAG_GROUP_OBJ, POSIXACE4_TAG_MASK and POSIXACE4_TAG_OTHER the who field SHOULD be of zero length and MUST be ignored by the receiver. For the posxiacetag4 values of POSIXACE4_TAG_USER and POSIXACE4_TAG_GROUP, the who field must be in the same format as would be used for the owner or owner_group attribute, respectively. If the server cannot translate the who string into a valid user or group, the server MUST reply NFS4ERR_BADOWNER. There MUST only be one POSIXACE4_TAG_USER ACE in the ACL for each user as represented by the who value. Similarly, there MUST be only one POSIXACE4_TAG_GROUP in the ACL for each group as represented by the who value.

Since a POSIX default ACL only applies to directories, a SETATTR/OPEN/CREATE of the posix_default_acl for a non-directory object MUST reply NFS4ERR_INVALID. If SETATTR of a POSIX ACL for a non-zero length posixace4 array does not have one ACE for each of POSIXACE4_TAG_USER_OBJ, POSIXACE4_TAG_GROUP_OBJ and POSIXACE4_TAG_OTHER, the SETATTR of the ACL MUST reply NFS4ERR_INVALID. If SETATTR of a POSIX extended ACL does not have a POSIXACE4_TAG_MASK ACE, the SETATTR of the ACL MUST reply NFS4ERR_INVALID.

For file servers that reply ACL_SCOPE_FILE_OBJECT for the acl_trueform_scope attribute, a successful setting of this attribute sets the value of acl_trueform to ACL_MODEL_POSIX_DRAFT. In addition, if the object's acl model had been ACL_MODEL_NFS4, any dacl (all ALLOW/DENY ACES) will be deleted from the stored ACL. As noted above, any AUDIT/ALARM ACES stored for the file object are not affected.

Doing a SETATTR for a posix_default_acl of a zero length posixace4 array deletes the POSIX default ACL, if one exists. For file servers that reply ACL_SCOPE_FILE_OBJECT for the acl_trueform attribute and no POSIX access ACL exists for the directory, this SETATTR might set the file object's acl_trueform to ACL_MODEL_NONE.

If a directory does not have a POSIX default ACL, a GETATTR/REaddir for the posix_default_acl attribute will reply with a posixace4 array of zero length.

This attribute is a per-file system object attribute.

9.3.1. Rationale

Without this optional attribute, for a file object whose true form is `ACL_MODEL_POSIX_DRAFT`, the server must map a NFSv4 ACL into a POSIX default ACL. It also must somehow combine the POSIX default ACL used for inheritance with the POSIX access ACL used for access control to the directory itself during the mapping, since a directory file object can only have, at most, one NFSv4 ACL.

9.4. Attribute 92: `posix_access_acl`

This attribute specifies the POSIX access ACL for a file object.

For a `GETATTR/REaddir`, if the `acl_trueform` is `ACL_MODEL_NONE`, the server MUST return a zero length `posixace4` array.

For the `posixacetag4` values of `POSIXACE4_TAG_USER_OBJ`, `POSIXACE4_TAG_GROUP_OBJ`, `POSIXACE4_TAG_MASK` and `POSIXACE4_TAG_OTHER` the who field SHOULD be of zero length and MUST be ignored by the receiver. For the `posxiacetag4` values of `POSIXACE4_TAG_USER` and `POSIXACE4_TAG_GROUP`, the who field must be in the same format as would be used for the owner or owner_group attribute, respectively. If the server cannot translate the who string into a valid user or group, the server MUST reply `NFS4ERR_BADOWNER`. There MUST only be one `POSIXACE4_TAG_USER` ACE in the ACL for each user as represented by the who value. Similarly, there MUST be only one `POSIXACE4_TAG_GROUP` in the ACL for each group as represented by the who value.

For file servers that reply `ACL_SCOPE_FILE_OBJECT` for the `acl_trueform_scope` attribute, a successful setting of this attribute sets the value of `acl_trueform` to `ACL_MODEL_POSIX_DRAFT`. In addition, if the object's acl model had been `ACL_MODEL_NFS4`, any `dacl` (all `ALLOW/DENY` ACEs) will be deleted from the stored ACL. As noted above, any `AUDIT/ALARM` ACEs stored for the file object are not affected.

For file servers that reply `ACL_SCOPE_FILE_OBJECT` for the `acl_trueform_scope` attribute, a successful `SETATTR` of a `posix_access_acl` with a `posixace4` array of zero length deletes any POSIX access ACL stored on the file object. The deletion results in the file object having a `acl_trueform` value of `ACL_MODEL_NONE`. It also deletes any POSIX default ACL stored for the object, if it is a directory.

If SETATTR of a POSIX ACL does not have one ACE for each of POSIXACE4_TAG_USER_OBJ, POSIXACE4_TAG_GROUP_OBJ and POSIXACE4_TAG_OTHER, the SETATTR of the ACL MUST reply NFS4ERR_INVAL. If SETATTR of a POSIX extended ACL does not have a POSIXACE4_TAG_MASK ACE, the SETATTR of the ACL MUST reply NFS4ERR_INVAL.

This attribute is a per-file system object attribute.

9.4.1. Rationale

Without this optional attribute, for a file object whose true form is ACL_MODEL_POSIX_DRAFT, the server must map a NFSv4 ACL into a POSIX access ACL.

10. Security Considerations

ACLs are used to provide finer grained access control for file objects, which can be used to improve file system security. If the acl_trueform for a file object is ACL_MODEL_POSIX_DRAFT, these new attributes allow a client to set ACLs accurately, which should improve file system security when ACLs are used.

11. IANA Considerations

This document has no IANA actions.

12. Implementation Status

This section is to be removed before publishing as an RFC.

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs.

Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

12.1. FreeBSD NFS server and client

This section is to be removed before publishing as an RFC.

Organization: FreeBSD Project

URL: <https://www.freebsd.org>

Maturity: Experimental software based on the current document.

Coverage: The `posix_default_acl` and `posix_access_acl` attributes described in this document were implemented for a UFS file system configured to store POSIX ACLs as its true form. This experimental implementation does not explore the case of `ACL_SCOPE_FILE_OBJECT`.

Licensing: BSD

Implementation experience: The `setfacl` and `getfacl` commands appeared to function correctly.

12.2. Linux kernel patch for the NFSv4 client and server

This section is to be removed before publishing as an RFC.

URL: <https://people.freebsd.org/~rmacklem/linux-posixacl.patch>

Maturity: Experimental software based on the current document.

Coverage: The patch provides the Linux internal inode operations for `get_acl` and `set_acl`. This allows the `getfacl(1)` and `setfacl(1)` commands to work against a server that implements these extensions. It also provides the internal modifications that allow the `knfsd` to handle the attributes described in this extension.

Licensing: GPL

Implementation experience: The `setfacl` and `getfacl` commands appeared to function correctly, when tested against the experimental FreeBSD server noted above. The same tests also worked against a patched Linux `knfsd`. ACLs with many ACEs have been tested and work.

13. References

13.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

- [RFC4506] Eisler, M., Ed., "XDR: External Data Representation Standard", STD 67, RFC 4506, DOI 10.17487/RFC4506, May 2006, <<https://www.rfc-editor.org/info/rfc4506>>.
- [RFC5662] Shepler, S., Ed., Eisler, M., Ed., and D. Noveck, Ed., "Network File System (NFS) Version 4 Minor Version 1 External Data Representation Standard (XDR) Description", RFC 5662, DOI 10.17487/RFC5662, January 2010, <<https://www.rfc-editor.org/info/rfc5662>>.
- [RFC7862] Haynes, T., "Network File System (NFS) Version 4 Minor Version 2 Protocol", RFC 7862, DOI 10.17487/RFC7862, November 2016, <<https://www.rfc-editor.org/info/rfc7862>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8178] Noveck, D., "Rules for NFSv4 Extensions and Minor Versions", RFC 8178, DOI 10.17487/RFC8178, July 2017, <<https://www.rfc-editor.org/info/rfc8178>>.
- [RFC8275] Fields, J. and A. Gruenbacher, "Allowing Inheritable NFSv4 Access Control Entries to Override the Umask", RFC 8275, DOI 10.17487/RFC8275, December 2017, <<https://www.rfc-editor.org/info/rfc8275>>.
- [RFC8881] Noveck, D., Ed. and C. Lever, "Network File System (NFS) Version 4 Minor Version 1 Protocol", RFC 8881, DOI 10.17487/RFC8881, August 2020, <<https://www.rfc-editor.org/info/rfc8881>>.

13.2. Informational References

- [Eriksen] Eriksen, M. and J. Fields, "Mapping Between NFSv4 and Posix Draft ACLs", August 2006.
- [Gruenbacher] Gruenbacher, A., "POSIX Access Control Lists on Linux", USENIX 2003 Annual Technical Conference Proceedings, June 2003.
- [IEEE] Institute of Electrical and Electronics Engineers, "IEEE 1003.1e and 1003.2c: Draft Standard for Information Technology--Portable Operating System Interface (POSIX)--Part 1: System Application Program Interface (API) and Part 2: Shell and Utilities, draft 17", October 1997.

Acknowledgments

Thanks go to David Noveck for pointing out that a per-file object scope for `acl_trueform_scope` was useful and to Frank Filz for noting that IBM's GPFS already does this. David Noveck also helped greatly with editorial corrections. Thanks also goes to Pali Rohrer for assorted corrections, particularly with respect to the setting of a zero length NFSv4 ACL.

Author's Address

Rick Macklem
FreeBSD Project
Canada
Email: rmacklem@uoguelph.ca