NFSv4
Internet-Draft
Intended status: Informational
Expires: April 3, 2017

J. Fields
A. Gruenbacher
Red Hat

September 30, 2016

Allowing inheritable NFSv4 ACLs to override the umask
draft-ietf-nfsv4-umask-01

Abstract

In some environments, inheritable NFSv4 ACLs can be rendered ineffective by the application of the per-process umask. This is easily worked around by transmitting the umask and create mode separately, to allow servers to make more intelligent decisions about the new mode of a file.

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 http://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 April 3, 2017.

Table of Contents

1. Conventions Used in This Document 2
2. Problem Statement 2
3. mode_umask Attribute 3
4. Security Considerations 4
5. Normative References 4
Appendix A. Acknowledgments 5
Authors' Addresses 5

1. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Problem Statement

On Unix-like systems, each process is associated with a file mode creation mask (umask). In the absence of inheritable permissions, the umask specifies which permissions must be turned off when creating new file system objects. With "POSIX" Access Control Lists [POSIX-1003.1e], in the presence of inheritable permissions, the umask must be ignored. Other Access Control List implementations on Unix-like systems may ignore the umask in a similar way.

The NFSv4 protocol currently does not include the umask concept; applying the umask is left to clients. Unfortunately, clients have no way of atomically checking for inheritable permissions and applying the umask only when necessary. Instead, they err on the safe side and always apply the umask. Thus the mode the server receives in an OPEN already has the umask applied.

When applying the mode, section 6.4.1.1 of [RFC7530] recommends that servers SHOULD restrict permissions granted to any user or group named in the ACL to be no more than the permissions granted by the MODE4_RGRP, MODE4_WGRP, and MODE4_XGRP bits. Servers aiming to provide clients with Unix-like chmod behavior may also be motivated by the same requirements in [SUSv4]. (See the discussion of additional and alternate access control mechanisms in section "4.4 File Permissions".)

On many existing installations, all ordinary users by default use the same effective group ID. To prevent granting all users full access to each other's files, such installations usually default to a umask with very restrictive permissions. Thus the named users and groups in an inherited ACL end up being mostly ignored.

This leads to file permissions which are more restrictive than they should be in common cases; permission inheritance over NFSv4 is broken.

To address this problem, a new attribute is proposed which allows the server to apply the umask only when there are no inheritable permissions.

3. mode_umask Attribute

```
struct mode_umask4 {
  mode4  mu_mode;
  mode4  mu_umask;
};
```

Name	Id	Data Type	Acc	Defined in
mode_umask	81	mode_umask4	W	Section 3

Table 1

The NFSv4.2 mode_umask attribute is based on the open mode and umask that together determine the mode of a newly created UNIX file. Only the nine low-order mode4 bits of mu_umask are defined. A server MUST return NFS4ERR_INVAL if bits other than those nine are set.

The mode_umask attribute is only meaningful for operations that create objects (CREATE and OPEN); the server SHOULD reject it for other operations that take fattr4 arguments.

The server MUST ignore any mode attribute set in the same operation as mode_umask.

When the server supports the mode_umask attribute, a client creating a file should use mode_umask in place of mode, with mu_mode set to the unmodified mode provided by the user, and mu_umask set to the umask of the requesting process.

The server then uses mode_umask as follows:

- On a server that supports ACL attributes, if an object inherits any ACEs from its parent directory, mu_mode SHOULD be used, and mu_umask ignored.

- Otherwise, mu_umask MUST be used to limit the mode: all bits in the mode MUST be turned off which are set in the umask; the mode

to use for creating the object becomes (mu_mode & ~mu_umask) instead.

4. Security Considerations

The mode_umask attribute shifts to the server the decision about when to apply the umask. Because the server MUST apply the umask if there are no inheritable permissions, the traditional semantics are preserved in the absence of a permission inheritance mechanism. The only relaxation of permissions comes in the case servers follow the recommendation that they SHOULD ignore the umask in the presence of inheritable permissions.

The practice of ignoring the umask when there are inheritable permissions in the form of a "POSIX" default ACL is common practice; there are no known security concerns. The "POSIX" default ACL mechanism and the mechanism of inheriting permissions in NFSv4 is equivalent for this purpose.

5. Normative References

[LEGAL] IETF Trust, "Legal Provisions Relating to IETF Documents", November 2008, <http://trustee.ietf.org/docs/IETF-Trust-License-Policy.pdf>.

[POSIX-1003.1e] Portable Applications Standards Committee of the IEEE Compute Society, "POSIX 1003.1e Withdrawn Draft 17", October 1997.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", March 1997.

[RFC4506] Eisler, M., "XDR: External Data Representation Standard", STD 67, RFC 4506, May 2006.

[RFC5661] Shepler, S., Ed., Eisler, M., Ed., and D. Noveck, Ed., "Network File System (NFS) Version 4 Minor Version 1 Protocol", RFC 5661, January 2010.

[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, January 2010.

[RFC7530] Haynes, T. and D. Noveck, "Network File System (NFS) version 4 Protocol", RFC 7530, March 2015.

[SUSv4] The Open Group, "Single UNIX Specification Version 4", 2013.

Appendix A. Acknowledgments

Thanks to Dave Noveck and Trond Myklebust for review.

Authors' Addresses

J. Bruce Fields
Red Hat, Inc.

Email: bfields@redhat.com

Andreas Gruenbacher
Red Hat, Inc.

Email: agruenba@redhat.com