NFS: Network File System Version 3 Protocol Specification

Comments to:
sun!eng!nfs3
nfs3@Eng.sun.com

February 16, 1994

ONC™ Technologies
2550 Garcia Avenue
Mountain View, CA 94043 USA

Table of Contents

Introduction

The Sun Network File System (NFS™) protocol provides transparent remote access to shared file systems across networks. The NFS protocol is designed to be machine, operating system, network architecture, and transport protocol independent. This independence is achieved through the use of Remote Procedure Call (RPC) primitives built on top of an eXternal Data Representation (XDR). Implementations of the NFS version 2 protocol exist for a variety of machines, from personal computers to supercomputers. The initial version of the NFS protocol is specified in the Network File System specification *[RFC1094]*. A description of the initial implementation can be found in *[Sandberg]*.

The supporting MOUNT protocol performs the operating system-specific functions that allow clients to attach remote directory trees to a point within the local file system. The mount process also allows the server to grant remote access privileges to a restricted set of clients via export control.

The Lock Manager provides support for file locking when used in the NFS environment. The Network Lock Manager (NLM) protocol isolates the inherently stateful aspects of file locking into a separate protocol.

A complete description of the above protocols and their implementation is to be found in *[X/OpenNFS]*.

The purpose of this document is to:

- Specify the NFS version 3 protocol
- Describe semantics of the protocol through annotation and description of intended implementation
- Specify the MOUNT version 3 protocol
- Briefly describe the changes between the NLM version 3 protocol and the NLM version 4 protocol.

The normative text is the description of the RPC procedures and arguments and results, which defines the over-the-wire protocol, and the semantics of those procedures. The material describing implementation practice aids the understanding of the protocol specification and describes some possible implementation issues and solutions. It is not possible to describe all implementations and the UNIX® operating system implementation of the NFS version 3 protocol is most often used to provide examples. Given that, the implementation discussion does not bear the authority of the description of the over-the wire protocol itself.

Scope of the NFS version 3 protocol

This revision of the NFS protocol addresses new requirements. The need to support larger files and file systems has prompted extensions to allow `64` bit file sizes and offsets. The revision enhances security by adding support for an access check to be done on the server. Performance modifications are of three types.

1. First, the number of over-the-wire packets for a given set of file operations is reduced by returning file attributes on every operation, thus decreasing the number of calls to get modified attributes.
2. Second, the write throughput bottleneck caused by the synchronous definition of write in the NFS version 2 protocol has been addressed by adding support so that the NFS server can do unsafe writes. Unsafe writes are writes which have not been committed to stable storage before the operation returns. This specification defines a method for committing these unsafe writes to stable storage in a reliable way.
3. Third, limitations on transfer sizes have been relaxed.

The ability to support multiple versions of a protocol in RPC will allow implementors of the NFS version 3 protocol to define clients and servers that provide backwards compatibility with the existing installed base of NFS version 2 protocol implementations.

The extensions described here represent an evolution of the existing NFS protocol and most of the design features of NFS described in *[Sandberg]* persist. See *Changes from the NFS version 2 protocol* on page 5 for a more detailed summary of the changes introduced by this revision.

Useful terms

In this specification, a *server* is a machine that provides resources to the network; a *client* is a machine that accesses

™ NFS is a registered trademark of Sun Microsystems, Inc.
® UNIX is a registered trademark of UNIX System Laboratories.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.