

**Addendum to CONVEX Network  
File Reference Set**  
Document No. 710-001699-201

---

---

November 1988

**CONVEX Computer Corporation**  
Richardson, Texas

© 1987, 1988 CONVEX Computer Corporation  
All rights reserved.

This document is copyrighted. This document may not, in whole or part, be copied, duplicated, reproduced, translated, stored electronically, or reduced to machine-readable form without prior written consent from CONVEX Computer Corporation.

Although the material contained herein has been carefully reviewed, CONVEX Computer Corporation (CONVEX) does not warrant it to be free of errors or omissions. CONVEX reserves the right to make corrections, updates, revisions or changes to the information contained herein. CONVEX does not warrant the material described herein to be free of patent infringement.

UNLESS PROVIDED OTHERWISE IN WRITING WITH CONVEX COMPUTER CORPORATION (CONVEX), THE PROGRAM DESCRIBED HEREIN IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. SOME STATES DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES. THE ABOVE EXCLUSION MAY NOT BE APPLICABLE TO ALL PURCHASERS BECAUSE WARRANTY RIGHTS CAN VARY FROM STATE TO STATE. IN NO EVENT WILL CONVEX BE LIABLE TO ANYONE FOR SPECIAL, COLLATERAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, INCLUDING ANY LOST PROFITS OR LOST SAVINGS, ARISING OUT OF THE USE OR INABILITY TO USE THIS PROGRAM. CONVEX WILL NOT BE LIABLE EVEN IF IT HAS BEEN NOTIFIED OF THE POSSIBILITY OF SUCH DAMAGE BY THE PURCHASER OR ANY THIRD PARTY.

© 1986 Sun Microsystems, Inc.  
© 1979, 1980, Bell Telephone Laboratories, Incorporated.

The Regents of the University of California and the Electrical Engineering and Computer Sciences Department at the Berkeley Campus of the University of California are given credit for their roles in the development of the UNIX Operating System.

CONVEX and the CONVEX logo ("C") are registered trademarks of CONVEX Computer Corporation.

UNIX is a trademark of AT&T Bell Laboratories.

Ethernet is a trademark of Xerox Corporation.

NFS is a trademark of Sun Microsystems, Inc.

Printed in the United States of America

## Replacement Instructions for *CONVEX Network File System* Reference Set

This addendum updates the *CONVEX Network File System Reference Set*, Document No. 710-001730-202, Second Edition, Rev. 1, as follows:

- This documentation supports CONVEX UNIX V7.0.
- This documentation corrects errors in the previous manual.
- The *CONVEX Network File System Reference Set* document number changes to Document No. 710-001730-203.
- The *CONVEX Network File System Reference Set* changes from Second Edition, Rev. 1 to Second Edition, Rev. 2.

Other changes are described in the Revision Information page. Please update your document as indicated in the following table.

Remove Pages	Insert Pages
title and copyright pages Revision Information iii through vi	title and copyright pages Revision Information iii through vi
<i>NFS Overview</i>	
title and copyright pages over-1-5 through over-1-6	title and copyright pages over-1-5 through over-1-6
<i>RPC Programming Guide</i>	
title and copyright pages iii through iv Chapter 2 Index (pages 1 through 3)	title and copyright pages iii through iv Chapter 2 Index (pages 1 through 3)
Master Index (pages 1 through 11) Reader's Forum	Master Index (pages 1 through 11) Reader's Forum

Place this sheet after the title and copyright pages for future reference.



**CONVEX Network File System**  
**Reference Set**  
Document No. 710-001730-203

---

---

Second Edition, Rev. 2  
November 1988

**CONVEX Computer Corporation**  
Richardson, Texas

*CONVEX Network File System*  
*Reference Set*  
Order No. DSW-111  
Second Edition, Rev. 2

© 1987, 1988 CONVEX Computer Corporation  
All rights reserved.

This document is copyrighted. This document may not, in whole or part, be copied, duplicated, reproduced, translated, stored electronically, or reduced to machine-readable form without prior written consent from CONVEX Computer Corporation.

Although the material contained herein has been carefully reviewed, CONVEX Computer Corporation (CONVEX) does not warrant it to be free of errors or omissions. CONVEX reserves the right to make corrections, updates, revisions or changes to the information contained herein. CONVEX does not warrant the material described herein to be free of patent infringement.

UNLESS PROVIDED OTHERWISE IN WRITING WITH CONVEX COMPUTER CORPORATION (CONVEX), THE PROGRAM DESCRIBED HEREIN IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. SOME STATES DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES. THE ABOVE EXCLUSION MAY NOT BE APPLICABLE TO ALL PURCHASERS BECAUSE WARRANTY RIGHTS CAN VARY FROM STATE TO STATE. IN NO EVENT WILL CONVEX BE LIABLE TO ANYONE FOR SPECIAL, COLLATERAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, INCLUDING ANY LOST PROFITS OR LOST SAVINGS, ARISING OUT OF THE USE OR INABILITY TO USE THIS PROGRAM. CONVEX WILL NOT BE LIABLE EVEN IF IT HAS BEEN NOTIFIED OF THE POSSIBILITY OF SUCH DAMAGE BY THE PURCHASER OR ANY THIRD PARTY.

© 1986 Sun Microsystems, Inc.  
© 1979, 1980, Bell Telephone Laboratories, Incorporated.

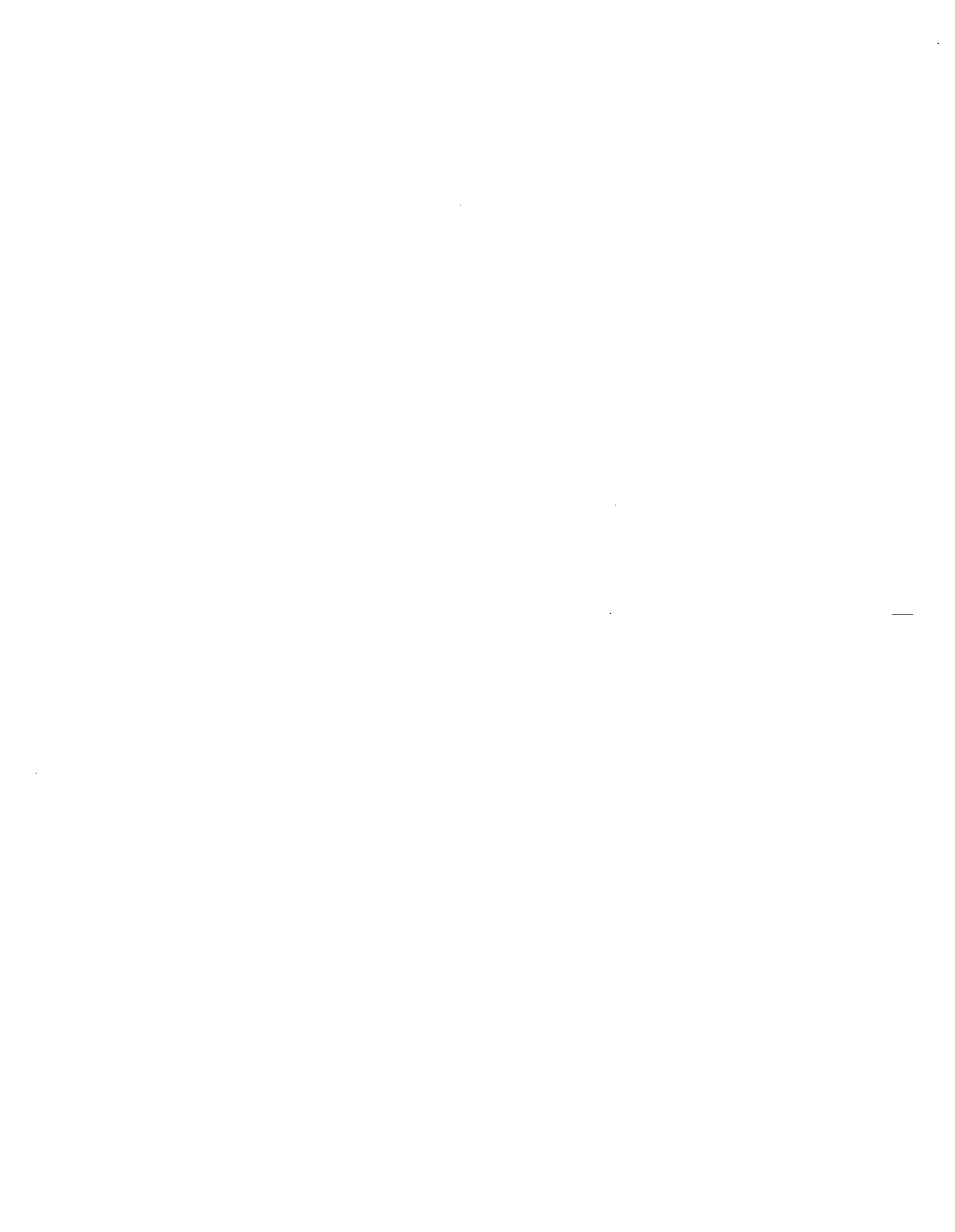
The Regents of the University of California and the Electrical Engineering and Computer Sciences Department at the Berkeley Campus of the University of California are given credit for their roles in the development of the UNIX Operating System.

CONVEX and the CONVEX logo ("C") are registered trademarks of CONVEX Computer Corporation.  
UNIX is a trademark of AT&T Bell Laboratories.  
Ethernet is a trademark of Xerox Corporation.  
NFS is a trademark of Sun Microsystems, Inc.

Printed in the United States of America

**Revision Information for  
CONVEX Network File System  
Reference Set**

Edition	Document No.	Description
Second Rev. 2	710-001730-203	Released with CONVEX UNIX V7.0, November 1988. Includes the following changes: <i>NFS Overview</i> Chapter 1, corrected sample entries to the <i>/etc/exports</i> file <i>RPC Programming Guide</i> Chapter 2, corrected sample program using <i>callrpc()</i>
Second Rev. 1	710-001730-202	Released with CONVEX UNIX V6.2, April 1988.
2.0	710-001730-201	Released with CONVEX UNIX V6.1, October 1987.
1.0	710-000630-000	Initial release with CONVEX UNIX V6.0, April 1987.



# NFS Overview

<b>1 Basics</b>	
Introduction .....	over-1-1
Computing Environments .....	over-1-2
Terms and Concepts .....	over-1-3
Comparison With Predecessors .....	over-1-3
Examples of How It Works .....	over-1-4
<b>2 Architecture of <i>nfs</i></b>	
Design Goals .....	over-2-1
<i>nfs</i> Implementation .....	over-2-2
<i>nfs</i> Interface .....	over-2-4
<b>3 Yellow Pages Database</b>	
Introduction .....	over-3-1
What Are the Yellow Pages? .....	over-3-1
<i>yp</i> Map .....	over-3-1
<i>yp</i> Domain .....	over-3-1
Servers and Clients .....	over-3-2
Masters and Slaves .....	over-3-2
<b>4 Overview of the Yellow Pages</b>	
Introduction .....	over-4-1
<i>yp</i> Network Service .....	over-4-1
Default <i>yp</i> Files .....	over-4-2

## List of Figures

1-1 Mounting Directories .....	over-1-5
2-1 System Call Request Flow .....	over-2-3

# NFS Protocol Spec.

<b>1 Introduction</b>	
Remote Procedure Call .....	nfs-1-1
External Data Representation .....	nfs-1-1
<b>2 <i>nfs</i> Protocol Definition</b>	
Introduction .....	nfs-2-1
Version 2 .....	nfs-2-1
<b>3 Mount Protocol Definition</b>	
Introduction .....	nfs-3-1
Version 1 .....	nfs-3-1

# YP Protocol Spec.

<b>1 Introduction and Terminology</b>	
Introduction .....	yp-1-1
<i>rpc</i> —Remote Procedure Call .....	yp-1-1
<i>xdr</i> —External Data Representation .....	yp-1-2
<b>2 <i>yp</i> Database Servers</b>	
Maps and Map Operations .....	yp-2-1
Master and Slave <i>yp</i> Database Servers .....	yp-2-2

Map Propagation and Consistency .....	yp-2-2
Domains .....	yp-2-2
Restrictions .....	yp-2-2
<i>yp</i> Database Server Protocol Definition .....	yp-2-3
<b>3 <i>yp</i> Binders</b>	
Introduction .....	yp-3-1
<i>yp</i> Binder Protocol Definition .....	yp-3-1

## RPC Protocol Spec.

<b>1 Introduction</b>	
Terminology .....	rpc-1-1
<i>rpc</i> Model .....	rpc-1-1
Transports and Semantics .....	rpc-1-2
Binding and Rendezvous Independence .....	rpc-1-2
Message Authentication .....	rpc-1-2
<b>2 <i>rpc</i> Protocol Requirements</b>	
Introduction .....	rpc-2-1
Remote Programs and Procedures .....	rpc-2-1
Authentication .....	rpc-2-2
Program Number Assignment .....	rpc-2-2
Other Uses of the <i>rpc</i> Protocol .....	rpc-2-3
<b>3 <i>rpc</i> Message Protocol</b>	
Protocol Definition .....	rpc-3-1
Authentication Parameter Specification .....	rpc-3-4
Record-Marking Standard .....	rpc-3-5

### Appendices

<b>A Port Mapper Program Protocol</b> .....	rpc-A-1
Introduction .....	rpc-A-1
<i>rpc</i> Protocol .....	rpc-A-1

### List of Figures

3-1 <i>rpc</i> Message Protocol Definition .....	rpc-3-1
--	---------

## XDR Protocol Spec.

<b>1 Basics</b>	
Introduction .....	xdr-1-1
Justification .....	xdr-1-1
<i>xdr</i> Library .....	xdr-1-4
<b>2 <i>xdr</i> Library Primitives</b>	
Introduction .....	xdr-2-1
Number Filters .....	xdr-2-1
Floating-Point Filters .....	xdr-2-2
Enumeration Filters .....	xdr-2-2
No Data Routines .....	xdr-2-3
Constructed Data Type Filters .....	xdr-2-3
Non-Filter Primitives .....	xdr-2-10
<i>xdr</i> Operation Directions .....	xdr-2-10

<b>3</b>	<b><i>xdr</i> Stream Access</b>	
	Introduction .....	xdr-3-1
	Standard I/O Streams .....	xdr-3-1
	Memory Streams .....	xdr-3-1
	Record (TCP/IP) Streams .....	xdr-3-2
<b>4</b>	<b><i>xdr</i> Stream Implementation</b>	
	Introduction .....	xdr-4-1
	The <i>xdr</i> Object .....	xdr-4-1
<b>5</b>	<b><i>xdr</i> Standard</b>	
	Introduction .....	xdr-5-1
	Basic Block Size .....	xdr-5-1
	Integer .....	xdr-5-1
	Unsigned Integer .....	xdr-5-1
	Enumerations .....	xdr-5-1
	Booleans .....	xdr-5-2
	Hyper Integer and Hyper Unsigned .....	xdr-5-2
	Floating Point and Double Precision .....	xdr-5-2
	Opaque Data .....	xdr-5-3
	Counted Byte Strings .....	xdr-5-3
	Fixed Arrays .....	xdr-5-3
	Counted Arrays .....	xdr-5-4
	Structures .....	xdr-5-4
	Discriminated Unions .....	xdr-5-4
	Missing Specifications .....	xdr-5-4
	Library Primitive / <i>xdr</i> Standard Cross-Reference .....	xdr-5-5
<b>6</b>	<b>Advanced Topics</b>	
	Linked Lists .....	xdr-6-1
	Record-Marking Standard .....	xdr-6-5

## Appendices

<b>A</b>	<b>Synopsis of <i>xdr</i> Routines</b> .....	xdr-A-1
----------	--	---------

## List of Tables

5-1	Primitives and Data Types .....	xdr-5-5
-----	---------------------------------	---------

# RPC Programming Guide

<b>1</b>	<b>Introduction to <i>rpc</i></b>	
	Overview .....	prog-1-1
	Layers of <i>rpc</i> .....	prog-1-1
	<i>rpc</i> Paradigm .....	prog-1-2
<b>2</b>	<b>Higher Layers of <i>rpc</i></b>	
	Highest Layer .....	prog-2-1
	Intermediate Layer .....	prog-2-2
	Assigning Program Numbers .....	prog-2-4
	Passing Arbitrary Data Types .....	prog-2-4
<b>3</b>	<b>Lowest Layer of <i>rpc</i></b>	
	Introduction .....	prog-3-1
	More on the Server Side .....	prog-3-1
	Memory Allocation With <i>xdr</i> .....	prog-3-3
	Calling Side .....	prog-3-5

<b>4 Other <i>rpc</i> Features</b>	
Select on the Server Side .....	prog-4-1
Broadcast <i>rpc</i> .....	prog-4-1
Batching .....	prog-4-2
Authentication .....	prog-4-6
Using <i>inetd</i> .....	prog-4-9
<b>5 More Examples</b>	
Versions .....	prog-5-1
TCP .....	prog-5-2
Callback Procedures .....	prog-5-5

## Appendices

<b>A Synopsis of <i>rpc</i> Routines</b> .....	prog-A-1
--	----------

### List of Tables

2-1 <i>rpc</i> Service Library Routines .....	prog-2-2
A-1 <i>req</i> Values for UDP and TCP .....	prog-A-3
A-2 <i>req</i> Values for UDP .....	prog-A-3

### List of Figures

1-1 <i>rpc</i> Paradigm .....	prog-1-2
3-1 <i>nusers</i> Program .....	prog-3-1
3-2 Calling <i>nusers</i> Service .....	prog-3-5
4-1 String Rendering Service .....	prog-4-3
4-2 Rendering Strings via Batching .....	prog-4-5
4-3 Extended Remote Users Service Example .....	prog-4-8
4-4 Sample <i>/etc/inetd.conf</i> File .....	prog-4-10
5-1 TCP Example .....	prog-5-2
5-2 <i>rpc</i> Callback Example .....	prog-5-5
5-3 Using <i>gettransient</i> Routine, Example 1 .....	prog-5-6
5-4 Using <i>gettransient</i> Routine, Example 2 .....	prog-5-8

# **CONVEX Network File System Overview**

---

---

November 1988

**CONVEX Computer Corporation**  
Richardson, Texas

© 1987, 1988 CONVEX Computer Corporation  
All rights reserved.

This document is copyrighted. This document may not, in whole or part, be copied, duplicated, reproduced, translated, stored electronically, or reduced to machine-readable form without prior written consent from CONVEX Computer Corporation.

Although the material contained herein has been carefully reviewed, CONVEX Computer Corporation (CONVEX) does not warrant it to be free of errors or omissions. CONVEX reserves the right to make corrections, updates, revisions or changes to the information contained herein. CONVEX does not warrant the material described herein to be free of patent infringement.

UNLESS PROVIDED OTHERWISE IN WRITING WITH CONVEX COMPUTER CORPORATION (CONVEX), THE PROGRAM DESCRIBED HEREIN IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. SOME STATES DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES. THE ABOVE EXCLUSION MAY NOT BE APPLICABLE TO ALL PURCHASERS BECAUSE WARRANTY RIGHTS CAN VARY FROM STATE TO STATE. IN NO EVENT WILL CONVEX BE LIABLE TO ANYONE FOR SPECIAL, COLLATERAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, INCLUDING ANY LOST PROFITS OR LOST SAVINGS, ARISING OUT OF THE USE OR INABILITY TO USE THIS PROGRAM. CONVEX WILL NOT BE LIABLE EVEN IF IT HAS BEEN NOTIFIED OF THE POSSIBILITY OF SUCH DAMAGE BY THE PURCHASER OR ANY THIRD PARTY.

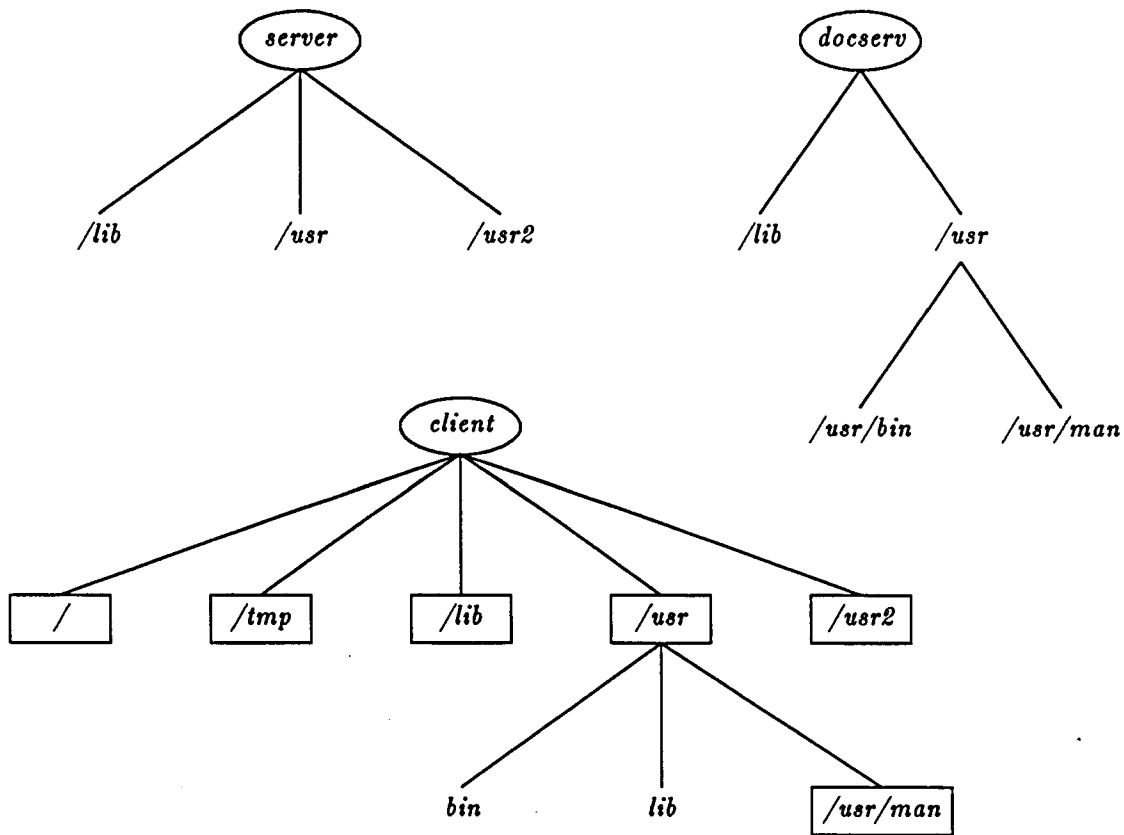
© 1986 Sun Microsystems, Inc.  
© 1979, 1980, Bell Telephone Laboratories, Incorporated.

The Regents of the University of California and the Electrical Engineering and Computer Sciences Department at the Berkeley Campus of the University of California are given credit for their roles in the development of the UNIX Operating System.

CONVEX and the CONVEX logo ("C") are registered trademarks of CONVEX Computer Corporation.  
UNIX is a trademark of AT&T Bell Laboratories.  
Ethernet is a trademark of Xerox Corporation.  
NFS is a trademark of Sun Microsystems, Inc.

Printed in the United States of America

Figure 1-1: Mounting Directories



## Exporting a Filesystem

Suppose that you and a colleague need to work together on a programming project. The source code is on your machine, in the directory `/usr/proj`. It does not matter whether your workstation is a diskless node, or has local disk. Suppose that after creating the proper directory, your colleague tried to remotely mount your directory. Unless you have explicitly exported the directory, your colleague's remote mount will fail with a *permission denied* message.

To export a directory, become superuser, and edit the file `/etc/exports`. If your colleague is on a machine named *cohort*, then you need to put this one line in `/etc/exports`:

```
/usr/proj -access=cohort
```

Without the keyword *cohort*, anybody on the network could remotely mount your directory `/usr/proj`. *nfs* mount request server `mountd(8c)` reads the `/etc/exports` file if necessary whenever it receives a request for a remote mount. Now your colleague can remotely mount the source directory by issuing this command:

```
cohort# /etc/mount client:/usr/proj /usr/proj
```

Because both you and your colleague can change files on */usr/proj*, it is best to use the *rcs(1)* source code control system for concurrency control.

Note that you can enable or disable over-the-net superuser privileges for client machines on either a machine-by-machine basis, or a filesystem-by-filesystem basis. For details, see "Superuser Access to Remote Files" in the *CONVEX Network File System System Manager's Guide*.

## Administering a Server Machine

System administrators must know how to set up the *nfs* server machine so that client workstations can mount all the necessary filesystems. You export filesystems (that is, make them available) by placing appropriate lines in the */etc/exports* file. Here is a sample */etc/exports* file for a typical server machine:

```
/
/usr
/usr2
/usr/src    -access=staff
```

The pathnames specified in */etc/exports* must be real filesystems—that is, directory mount points for disk devices. A netgroup, such as *staff*, may be specified after the filesystem, in which case remote mounts are limited to machines that are a member of this netgroup. At any one time, the system administrator can see which filesystems have been remotely mounted, by executing the *showmount(8)* command.

# CONVEX Remote Procedure Call Programming Guide

---

---

November 1988

**CONVEX Computer Corporation**  
Richardson, Texas

*CONVEX Remote Procedure Call  
Programming Guide*

© 1987, 1988 CONVEX Computer Corporation  
All rights reserved.

This document is copyrighted. This document may not, in whole or part, be copied, duplicated, reproduced, translated, stored electronically, or reduced to machine-readable form without prior written consent from CONVEX Computer Corporation.

Although the material contained herein has been carefully reviewed, CONVEX Computer Corporation (CONVEX) does not warrant it to be free of errors or omissions. CONVEX reserves the right to make corrections, updates, revisions or changes to the information contained herein. CONVEX does not warrant the material described herein to be free of patent infringement.

UNLESS PROVIDED OTHERWISE IN WRITING WITH CONVEX COMPUTER CORPORATION (CONVEX), THE PROGRAM DESCRIBED HEREIN IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. SOME STATES DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES. THE ABOVE EXCLUSION MAY NOT BE APPLICABLE TO ALL PURCHASERS BECAUSE WARRANTY RIGHTS CAN VARY FROM STATE TO STATE. IN NO EVENT WILL CONVEX BE LIABLE TO ANYONE FOR SPECIAL, COLLATERAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, INCLUDING ANY LOST PROFITS OR LOST SAVINGS, ARISING OUT OF THE USE OR INABILITY TO USE THIS PROGRAM. CONVEX WILL NOT BE LIABLE EVEN IF IT HAS BEEN NOTIFIED OF THE POSSIBILITY OF SUCH DAMAGE BY THE PURCHASER OR ANY THIRD PARTY.

© 1986 Sun Microsystems, Inc.  
© 1979, 1980, Bell Telephone Laboratories, Incorporated.

The Regents of the University of California and the Electrical Engineering and Computer Sciences Department at the Berkeley Campus of the University of California are given credit for their roles in the development of the UNIX Operating System.

CONVEX and the CONVEX logo ("C") are registered trademarks of CONVEX Computer Corporation.

C1 is a trademark of CONVEX Computer Corporation.

UNIX is a trademark of AT&T Bell Laboratories.

Ethernet is a trademark of Xerox Corporation.

NFS is a trademark of Sun Microsystems, Inc.

Printed in the United States of America

# Table of Contents

<b>1 Introduction to <i>rpc</i></b>	
Overview .....	prog-1-1
Layers of <i>rpc</i> .....	prog-1-1
<i>rpc</i> Paradigm .....	prog-1-2
<b>2 Higher Layers of <i>rpc</i></b>	
Highest Layer .....	prog-2-1
Intermediate Layer .....	prog-2-2
Assigning Program Numbers .....	prog-2-4
Passing Arbitrary Data Types .....	prog-2-4
<b>3 Lowest Layer of <i>rpc</i></b>	
Introduction .....	prog-3-1
More on the Server Side .....	prog-3-1
Memory Allocation With <i>xdr</i> .....	prog-3-3
Calling Side .....	prog-3-5
<b>4 Other <i>rpc</i> Features</b>	
Select on the Server Side .....	prog-4-1
Broadcast <i>rpc</i> .....	prog-4-1
Batching .....	prog-4-2
Authentication .....	prog-4-6
Using <i>inetd</i> .....	prog-4-9
<b>5 More Examples</b>	
Versions .....	prog-5-1
TCP .....	prog-5-2
Callback Procedures .....	prog-5-5

## Appendices

<b>A Synopsis of <i>rpc</i> Routines</b> .....	A-1
--	-----

## List of Tables

2-1 <i>rpc</i> Service Library Routines .....	prog-2-2
A-1 <i>req</i> Values for UDP and TCP .....	prog-A-3
A-2 <i>req</i> Values for UDP .....	prog-A-3

## List of Figures

1-1 <i>rpc</i> Paradigm .....	prog-1-2
3-1 <i>nusers</i> Program .....	prog-3-1
3-2 Calling <i>nusers</i> Service .....	prog-3-5
4-1 String Rendering Service .....	prog-4-3
4-2 Rendering Strings via Batching .....	prog-4-5
4-3 Extended Remote Users Service Example .....	prog-4-8
4-4 Sample <i>/etc/inetd.conf</i> File .....	prog-4-10
5-1 TCP Example .....	prog-5-2
5-2 <i>rpc</i> Callback Example .....	prog-5-5

5-3 Using <i>gettransient</i> Routine, Example 1 .....	prog-5-6
5-4 Using <i>gettransient</i> Routine, Example 2 .....	prog-5-8

## Higher Layers of *rpc*

### Highest Layer

Imagine you're writing a program that needs to know how many users are logged in to a remote machine. You can do this by calling the library routine *rnusers()*, as illustrated below:

```
#include <stdio.h>

main(argc, argv)
    int argc;
    char **argv;
{
    unsigned num;
    if (argc < 2) {
        fprintf(stderr, "usage: rnusers hostname\n");
        exit(1);
    }
    if ((num = rnusers(argv[1])) < 0) {
        fprintf(stderr, "error: rnusers\n");
        exit(-1);
    }
    printf("%d users on %s\n", num, argv[1]);
    exit(0);
}
```

*rpc* library routines such as *rnusers()* are in the *rpc* services library *librpcsvc.a*. Thus, the program above should be compiled with:

```
% cc program.c -lrpcsvc
```

This routine, and other *rpc* library routines, are documented in Section 3 of the *CONVEX UNIX Programmer's Manual*. Table 2-1 shows the *rpc* service library routines available to the C programmer.

Table 2-1: *rpc* Service Library Routines

<i>Routine</i>	<i>Description</i>
<i>rnusers()</i>	Return number of users on remote machine
<i>rusers()</i>	Return information about users on remote machine
<i>havedisk()</i>	Determine if remote machine has disk
<i>rstat()</i>	Get performance data from remote kernel
<i>rwall()</i>	Write to specified remote machines
<i>getrpcport()</i>	Get <i>rpc</i> port number
<i>yppasswd()</i>	Update user password in yellow pages

The other *rpc* services—*ether*, *mount*, *rquota*, and *spray*—are not available to the C programmer as library routines. They do, however, have *rpc* program numbers so they can be invoked with *callrpc()*, discussed in the next section.

## Intermediate Layer

The simplest interface that explicitly makes *rpc* calls, uses the functions *callrpc()* and *registerrpc()*. Using this method, another way to get the number of remote users is:

```
#include <stdio.h>
#include <utmp.h>
#include <rpc/types.h>
#include <rpc/xdr.h>
#include <rpcsvc/rusers.h>

main(argc, argv)
    int argc;
    char **argv;
{
    unsigned long nusers;

    if (argc < 2) {
        fprintf(stderr, "usage: nusers hostname\n");
        exit(-1);
    }
    if (callrpc(argv[1],
        RUSERSPROG, RUSERSVERS, RUSERSPROC_NUM,
        xdr_void, 0, xdr_u_long, &nusers) != 0) {
        fprintf(stderr, "error: callrpc\n");
        exit(1);
    }
    printf("%d users on %s\n", nusers, argv[1]);
    exit(0);
}
```

A program number, version number, and procedure number defines each *rpc* procedure. The program number defines a group of related remote procedures, each of which has a different procedure number. Each program also has a version number, so when a minor change is made to a remote service (adding a new procedure, for example), a new program number doesn't have to be assigned. When you want to call a procedure to find the number of remote users, you look up the appropriate program, version, and procedure numbers in a manual, similar to when you look up the name of memory allocator when you want to allocate memory. The simplest routine in the *rpc* library used to make remote procedure calls is *callrpc()*. It has eight parameters. The first is the name of the remote machine. The next three parameters are the program, version, and

procedure numbers. The following two parameters define the argument of the *rpc* call, and the final two parameters are for the return value of the call. If it completes successfully, *callrpc()* returns zero, but nonzero otherwise. The exact meaning of the return codes is found in `<rpc/clnt.h>`, and is in fact an *enum clnt\_stat* cast into an integer.

Since data types may be represented differently on different machines, *callrpc()* needs both the type of the *rpc* argument, as well as a pointer to the argument itself (and similarly for the result). For *RUSERSPROC\_NUM*, the return value is an *unsigned long*, so *callrpc()* has *xdr\_u\_long* as its first return parameter, which says that the result is of type *unsigned long*, and *Snusers* as its second return parameter, which is a pointer to where the long result is placed. Since *RUSERSPROC\_NUM* takes no argument, the argument parameter of *callrpc()* is *xdr\_void*.

After trying several times to deliver a message, if *callrpc()* gets no answer, it returns with an error code. The delivery mechanism is UDP, which stands for User Datagram Protocol. Methods for adjusting the number of retries or for using a different protocol require you to use the lower layer of the *rpc* library, discussed later in this document. The remote server procedure corresponding to the above might look like this:

```
char *
nuser(indata)
    char *indata;
{
    static int nusers;

    /*
     * code here to compute the number of users
     * and place result in variable nusers
     */
    return((char *)&nusers);
}
```

It takes one argument, which is a pointer to the input of the remote procedure call (ignored in our example), and it returns a pointer to the result. In the current version of C, character pointers are the generic pointers, so both the input argument and the return value are cast to *char \**.

Normally, a server registers all of the *rpc* calls it plans to handle, and then goes into an infinite loop waiting to service requests. In this example, there is only a single procedure to register, so the main body of the server would look like this:

```
#include <stdio.h>
#include <rpcsvc/rusers.h>

char *nuser();

main()
{
    registerrpc(RUSERSPROC, RUSERSVERS, RUSERSPROC_NUM,
               nuser, xdr_void, xdr_u_long);
    svc_run(); /* never returns */
    fprintf(stderr, "Error: svc_run returned!\n");
    exit(1);
}
```

The *registerrpc()* routine establishes what C procedure corresponds to each *rpc* procedure number.

The first three parameters, *RUSERPROG*, *RUSERSVERS*, and *RUSERSPROC\_NUM*, are the program, version, and procedure numbers of the remote procedure to be registered; *nuser()* is the name of the C procedure implementing it; and *xdr\_void* and *xdr\_u\_long* are the types of the input to and output from the procedure.

Only the UDP transport mechanism can use *registerrpc()*; thus, it is always safe in conjunction with calls generated by *callrpc()*.

Warning: the UDP transport mechanism can only deal with arguments and results fewer than 8 Kbytes in length.

## Assigning Program Numbers

Program numbers are assigned in groups of 0x20000000 (536870912) according to the following chart:

0 - 1fffffff	defined by sun
20000000 - 3fffffff	defined by user
40000000 - 5fffffff	transient
60000000 - 7fffffff	reserved
80000000 - 9fffffff	reserved
a0000000 - bfffffff	reserved
c0000000 - dfffffff	reserved
e0000000 - ffffffff	reserved

Sun Microsystems administers the first group of numbers, which should be identical for all customers. If a customer develops an application that might be of general interest, that application should be given an assigned number in the first range. The second group of numbers is reserved for specific customer applications. This range is intended primarily for debugging new programs. The third group is reserved for applications that generate program numbers dynamically. The final groups are reserved for future use, and should not be used.

To register a protocol specification, send a request by network mail to *sun!rpc*, or write to:

RPC Administrator  
Sun Microsystems  
2550 Garcia Ave.  
Mountain View, CA 94043

Please include a complete protocol specification similar to those in this manual for *nfs* and *yp*. You will be given a unique program number in return.

## Passing Arbitrary Data Types

In the previous example, the *rpc* call passes a single *unsigned long*. *rpc* can handle arbitrary data structures, regardless of different machines' byte orders or structure layout conventions, by always converting them to a network standard called *eXternal Data Representation (xdr)* before sending them over the wire. The process of converting from a particular machine representation to *xdr* format is called *serializing*, and the reverse process is called *deserializing*. The type field parameters of *callrpc()* and *registerrpc()* can be a built-in procedure like *xdr\_u\_long()* in the previous example, or a user-supplied one. *xdr* has these built-in type routines:

```

xdr_int()      xdr_u_int()    xdr_enum()
xdr_long()    xdr_u_long()   xdr_bool()
xdr_short()   xdr_u_short()  xdr_string()

```

As an example of a user-defined type routine, to send the structure:

```

struct simple {
    int a;
    short b;
} simple;

```

call *callrpc()* as:

```

callrpc(hostname, PROGNUM, VERSNUM, PROCNUM,
         xdr_simple, &simple ...);

```

where *xdr\_simple()* is written as:

```

#include <rpc/rpc.h>
xdr_simple(xdresp, simplep)
    XDR *xdresp;
    struct simple *simplep;
{
    if (!xdr_int(xdresp, &simplep->a))
        return (0);
    if (!xdr_short(xdresp, &simplep->b))
        return (0);
    return (1);
}

```

An *xdr* routine returns nonzero (true in the sense of C) if it completes successfully, and zero otherwise. A complete description of *xdr* is in the *XDR Protocol Specification*, so this section only gives a few examples of *xdr* implementation.

In addition to the built-in primitives, there are also the prefabricated building blocks:

```

xdr_array()    xdr_bytes()
xdr_reference() xdr_union()

```

To send a variable array of integers, you might package them as a structure like this:

```

struct varintarr {
    int *data;
    int arrlen;
} arr;

```

and make an *rpc* call such as:

```

callrpc(hostname, PROGNUM, VERSNUM, PROCNUM,
         xdr_varintarr, &arr...);

```

with *xdr\_varintarr()* defined as:

```

xdr_varintarr(xdrsp, arrp)
    XDR *xdrsp;
    struct varintarr *arrp;
{
    xdr_array(xdrsp, &arrp->data, &arrp->arrlenh, MAXLEN,
              sizeof(int), xdr_int);
}

```

This routine takes as parameters the *xdr* handle, a pointer to the array, a pointer to the size of the array, the maximum allowable array size, the size of each array element, and an *xdr* routine for handling each array element.

If the size of the array is known in advance, then the following could also be used to send an array of length *SIZE*:

```

int intarr[SIZE];
xdr_intarr(xdrsp, intarr)
    XDR *xdrsp;
    int intarr[];
{
    int i;
    for (i = 0; i < SIZE; i++) {
        if (!xdr_int(xdrsp, &intarr[i]))
            return (0);
    }
    return (1);
}

```

*xdr* always converts quantities to 4-byte multiples when deserializing. Thus, if either of the examples above involved characters instead of integers, each character would occupy 32 bits. That is the reason for the *xdr* routine *xdr\_bytes()*, which is like *xdr\_array()* except that it packs characters; *xdr\_bytes()* has four parameters, similar to the first four parameters of *xdr\_array()*. For null-terminated strings, there is also the *xdr\_string()* routine, which is the same as *xdr\_bytes()* without the length parameter. On serializing it gets the string length from *strlen()*, and on deserializing it creates a null-terminated string.

Here is a final example that calls the previously written *xdr\_simple()* as well as the built-in functions *xdr\_string()* and *xdr\_reference()*, which chase pointers:

```
struct finalexample {
    char *string;
    struct simple *simplep;
} finalexample;

xdr_finalexample(xdrsp, finalp)
    XDR *xdrsp;
    struct finalexample *finalp;
{
    int i;
    if (!xdr_string(xdrsp, &finalp->string, MAXSTRLEN))
        return (0);
    if (!xdr_reference(xdrsp, &finalp->simplep,
        sizeof(struct simple), xdr_simple);
        return (0);
    return (1);
}
```



# Index

*clnt\_create*, *rpc* routines prog-A-3

## A

allocating memory with *xdr*, example  
prog-3-4  
*auth\_destroy* routine, *rpc* prog-A-1  
authentication credentials structure, *rpc*  
prog-4-6  
authentication handle, *rpc* prog-4-6  
authentication of *rpc* calls prog-4-6  
authentication of *rpc* calls, calling side  
prog-4-6  
authentication of *rpc* calls, server side  
prog-4-7  
*authnone\_create* routine, *rpc* prog-A-1  
*authunix\_create* routine, *rpc* prog-A-1  
*authunix\_create\_default* routine, *rpc* prog-A-1

## B

batch call attributes, with *rpc* prog-4-4  
batching, *rpc* prog-4-2  
batching, *rpc*, example prog-4-3  
broadcast *rpc* prog-4-1  
broadcast *rpc*, synopsis prog-4-2  
built-in *xdr* procedures, use in serializing data  
structures prog-2-4

## C

callback procedures, via *rpc* prog-5-5  
callback procedures, via *rpc*, example  
prog-5-5  
*callrpc* prog-1-1, prog-2-2, prog-3-4  
*callrpc*, example procedure prog-2-3  
*callrpc*, parameters used with prog-2-2  
*callrpc*, return codes prog-2-3  
*callrpc* routine, *rpc* prog-A-2  
*callrpc*, use with built-in procedures prog-2-4  
client handles, *rpc*, example prog-4-6  
*clnt\_broadcast* routine, *rpc* prog-A-2  
*clnt\_call* routine, *rpc* prog-A-2  
*clnt\_control*, *rpc* routines prog-A-3  
*clnt\_destroy* routine, *rpc* prog-A-4  
*clnt\_freeres* routine, *rpc* prog-A-4  
*clnt\_geterr* routine, *rpc* prog-A-4  
*clnt\_pcreateerror* routine, *rpc* prog-A-4  
*clnt\_perrno* routine, *rpc* prog-A-4  
*clnt\_perror* routine, *rpc* prog-A-5  
*clntraw\_create* routine, *rpc* prog-A-6  
*clnt\_screateerror* routine, *rpc* prog-A-5  
*clnt\_sperrno* routine, *rpc* prog-A-5  
*clnt\_sperror* routine, *rpc* prog-A-5  
*clnt\_syslog* routine, *rpc* prog-A-5  
*clnttcp\_create* routine, *rpc* prog-A-6  
*clntudp\_create* routine, *rpc* prog-A-6

## D

debugging, remote, via *rpc* prog-5-5  
deserialization prog-3-4, prog-5-2  
deserializing data structures prog-2-4,  
prog-2-6, prog-3-1

## E

*/etc/inetd.conf*, entry format for *rpc* services  
prog-4-9  
*/etc/termcap* prog-4-6  
*ether* service prog-2-2

## F

*fscanf(3s)* prog-4-6

## G

*get\_myaddress* routine, *rpc* prog-A-7  
*gettransient* routine, using, example prog-5-6

## I

*inetd*, using to start *rpc* servers prog-4-9

## L

*librpcsvc.a* prog-2-1

## M

*malloc(3)* prog-1-1  
memory allocation with *xdr* prog-3-3  
memory allocation with *xdr*, example routine  
prog-3-3, prog-3-4  
*mount* service prog-2-2  
multiple program versions, C procedures used  
to implement prog-5-1  
multiple program versions, supporting  
prog-5-1

## P

*pmap\_getmaps* routine, *rpc* prog-A-7  
*pmap\_getport* routine, *rpc* prog-A-7  
*pmap\_rmtcall* routine, *rpc* prog-A-7  
*pmap\_set* routine, *rpc* prog-A-8  
*pmap\_unset* routine, *rpc* prog-A-8  
port numbers, determination under *rpc*  
prog-3-2  
*portmap(8)* prog-4-1  
*portmapper* prog-4-1  
procedure numbers, example of use prog-3-3  
procedure numbers used to define *rpc* pro-  
cedures prog-2-2  
program numbers, assigning to *rpc* highest  
layer prog-2-4  
program numbers used to define *rpc* pro-  
cedures prog-2-2  
program numbers used with callback pro-  
cedures prog-5-5  
protocol specifications, registering prog-2-4

## R

registering protocol specifications prog-2-4  
*registerrpc* prog-1-1, prog-2-2, prog-2-3,  
prog-3-2  
*registerrpc* routine, *rpc* prog-A-8  
*registerrpc*, use with built-in procedures  
prog-2-4  
remote debugging, via *rpc* prog-5-5  
remote users service, example prog-4-8

- rendering strings, via *rpc* batching prog-4-4
- request handle, *rpc* prog-4-7
- rnusers* prog-1-1, prog-2-1
- rpc* and remote debugging prog-5-5
- rpc* authentication credentials structure prog-4-6
- rpc* authentication handle prog-4-6
- rpc*, batching prog-4-2
- rpc*, batching, example prog-4-3
- rpc*, broadcast prog-4-1
- rpc*, broadcast, synopsis prog-4-2
- rpc* callback procedure, example prog-5-5
- rpc* callback remotes prog-5-5
- rpc* calls, authentication of prog-4-6
- rpc* calls, authentication of, on calling side prog-4-6
- rpc* calls, authentication of, server side prog-4-7
- rpc* calls, *callrpc* prog-1-1
- rpc* calls, *registerrpc* prog-1-1
- rpc* calls, *rnusers* prog-1-1
- rpc* client handle, example prog-4-6
- rpc*, determination of port numbers prog-3-2
- rpc*, differences between broadcast and normal prog-4-1
- rpc*, example prog-3-5
- rpc*, highest layer prog-1-1, prog-2-1
- rpc*, highest layer, assigning program numbers prog-2-4
- rpc*, layers prog-1-1
- rpc*, lowest layer prog-1-1, prog-3-1
- rpc*, lowest layers, example prog-3-1, prog-3-3
- rpc*, lowest layers, occasions for use prog-3-1
- rpc*, middle layer prog-1-1, prog-2-2
- rpc*, middle layer, example of use prog-2-2
- rpc*, overview prog-1-1
- rpc*, paradigm prog-1-2
- rpc*, passing arbitrary data structures prog-2-4
- rpc* remote users service example prog-4-8
- rpc* request handle prog-4-7
- rpc* routines, *auth\_destroy* prog-A-1
- rpc* routines, *authnone\_create* prog-A-1
- rpc* routines, *authunix\_create* prog-A-1
- rpc* routines, *authunix\_create\_default* prog-A-1
- rpc* routines, *callrpc* prog-A-2
- rpc* routines, *clnt\_broadcast* prog-A-2
- rpc* routines, *clnt\_call* prog-A-2
- rpc* routines, *clnt\_control* prog-A-3
- rpc* routines, *clnt\_create* prog-A-3
- rpc* routines, *clnt\_destroy* prog-A-4
- rpc* routines, *clnt\_freeres* prog-A-4
- rpc* routines, *clnt\_geterr* prog-A-4
- rpc* routines, *clnt\_pcreateerror* prog-A-4
- rpc* routines, *clnt\_perrno* prog-A-4
- rpc* routines, *clnt\_perror* prog-A-5
- rpc* routines, *clntraw\_create* prog-A-6
- rpc* routines, *clnt\_spccreateerror* prog-A-5
- rpc* routines, *clnt\_sperrno* prog-A-5
- rpc* routines, *clnt\_spcerror* prog-A-5
- rpc* routines, *clnt\_syslog* prog-A-5
- rpc* routines, *clnttcp\_create* prog-A-6
- rpc* routines, *clntudp\_create* prog-A-6
- rpc* routines, *get\_myaddress* prog-A-7
- rpc* routines, *pmap\_getmaps* prog-A-7
- rpc* routines, *pmap\_getport* prog-A-7
- rpc* routines, *pmap\_rmtcall* prog-A-7
- rpc* routines, *pmap\_set* prog-A-8
- rpc* routines, *pmap\_unset* prog-A-8
- rpc* routines, *registerrpc* prog-A-8
- rpc* routines, *rpc\_createerr* prog-A-8
- rpc* routines, *svc\_destroy* prog-A-8
- rpc* routines, *svcerr\_auth* prog-A-11
- rpc* routines, *svcerr\_decode* prog-A-11
- rpc* routines, *svcerr\_noproc* prog-A-11
- rpc* routines, *svcerr\_noprogram* prog-A-11
- rpc* routines, *svcerr\_progvers* prog-A-12
- rpc* routines, *svcerr\_systemerr* prog-A-12
- rpc* routines, *svcerr\_weakauth* prog-A-12
- rpc* routines, *svcsd\_create* prog-A-13
- rpc* routines, *svc\_fds* prog-A-9
- rpc* routines, *svc\_fdset* prog-A-9
- rpc* routines, *svc\_freeargs* prog-A-9
- rpc* routines, *svc\_getargs* prog-A-9
- rpc* routines, *svc\_getcaller* prog-A-9
- rpc* routines, *svc\_getreq* prog-A-9
- rpc* routines, *svc\_getreqset* prog-A-10
- rpc* routines, *svccraw\_create* prog-A-12
- rpc* routines, *svc\_register* prog-A-10
- rpc* routines, *svc\_run* prog-A-10
- rpc* routines, *svc\_sendreply* prog-A-10
- rpc* routines, *svctcp\_create* prog-A-12
- rpc* routines, *svcurdp\_create* prog-A-13
- rpc* routines, *svc\_unregister* prog-A-11
- rpc* routines, *xdr\_accepted\_reply* prog-A-13
- rpc* routines, *xdr\_array* prog-A-13
- rpc* routines, *xdr\_authunix\_parms* prog-A-14
- rpc* routines, *xdr\_bool* prog-A-14
- rpc* routines, *xdr\_bytes* prog-A-14
- rpc* routines, *xdr\_callhdr* prog-A-14
- rpc* routines, *xdr\_callmsg* prog-A-14
- rpc* routines, *xdr\_char* prog-A-15
- rpc* routines, *xdr\_double* prog-A-15
- rpc* routines, *xdr\_enum* prog-A-15
- rpc* routines, *xdr\_float* prog-A-15
- rpc* routines, *xdr\_hyper* prog-A-15
- rpc* routines, *xdr\_inline* prog-A-16
- rpc* routines, *xdr\_int* prog-A-16
- rpc* routines, *xdr\_long* prog-A-16
- rpc* routines, *xdr\_opaque* prog-A-16
- rpc* routines, *xdr\_opaque\_auth* prog-A-16
- rpc* routines, *xdr\_pmap* prog-A-17
- rpc* routines, *xdr\_pmaplist* prog-A-17
- rpc* routines, *xdr\_pointer* prog-A-17
- rpc* routines, *xdr\_reference* prog-A-17
- rpc* routines, *xdr\_rejected\_reply* prog-A-17
- rpc* routines, *xdr\_replymsg* prog-A-18
- rpc* routines, *xdr\_short* prog-A-18
- rpc* routines, *xdr\_string* prog-A-18
- rpc* routines, *xdr\_u\_char* prog-A-18
- rpc* routines, *xdr\_u\_hyper* prog-A-18

*rpc* routines, *xdr\_u\_int* prog-A-19  
*rpc* routines, *xdr\_u\_long* prog-A-19  
*rpc* routines, *xdr\_union* prog-A-19  
*rpc* routines, *xdr\_u\_short* prog-A-19  
*rpc* routines, *xdr\_vector* prog-A-19  
*rpc* routines, *xdr\_void* prog-A-20  
*rpc* routines, *xdr\_wrapstring* prog-A-20  
*rpc* routines, *xprt\_register* prog-A-20  
*rpc* routines, *xprt\_unregister* prog-A-20  
*rpc* servers, starting with *inetd* prog-4-9  
*rpc*, service library routines available  
 prog-2-1  
*rpc* services, */etc/inetd.conf* entry format  
 prog-4-9  
*rpc* services library, *librpcsvc.a* prog-2-1  
*rpc*, string rendering service, example  
 prog-4-3  
*rpc*, use with TCP, example prog-5-2  
*rpc*, use with *xdr* prog-2-4  
*rpc\_createerr* routine, *rpc* prog-A-8  
*rquota* service prog-2-2

## S

*select*, use with *rpc*, example prog-4-1  
 serialization prog-3-4, prog-5-2  
 serializing data structures prog-2-4, prog-2-6,  
 prog-3-1, prog-3-3  
 service dispatch routines, guaranteed *rpc* func-  
 tionality prog-4-7  
*spray* service prog-2-2  
 string rendering service, *rpc* example  
 prog-4-3  
 string rendering, via *rpc* batching, perfor-  
 mance results prog-4-6  
 strings, rendering via *rpc* batching prog-4-4  
*svc\_destroy* routine, *rpc* prog-A-8  
*svcerr\_auth* routine, *rpc* prog-A-11  
*svcerr\_decode* routine, *rpc* prog-A-11  
*svcerr\_noproc* routine, *rpc* prog-A-11  
*svcerr\_noprogram* routine, *rpc* prog-A-11  
*svcerr\_progmisc* routine, *rpc* prog-A-12  
*svcerr\_systemerr* routine, *rpc* prog-A-12  
*svcerr\_weakauth* routine, *rpc* prog-A-12  
*svcsd\_create* routine, *rpc* prog-A-13  
*svcsd\_fds* routine, *rpc* prog-A-9  
*svcsd\_fdset* routine, *rpc* prog-A-9  
*svcsd\_freeargs* routine, *rpc* prog-A-9  
*svcsd\_getargs* routine, *rpc* prog-A-9  
*svcsd\_getcaller* routine, *rpc* prog-A-9  
*svcsd\_getreq* routine, *rpc* prog-A-9  
*svcsd\_getreqset* routine, *rpc* prog-A-10  
*svcsd\_create* routine, *rpc* prog-A-12  
*svcsd\_register* routine, *rpc* prog-A-10  
*svcsd\_run* routine, *rpc* prog-A-10  
*svcsd\_sendreply* routine, *rpc* prog-A-10  
*svcsd\_tcp\_create* routine, *rpc* prog-A-12  
*svcsd\_udp\_create* routine, *rpc* prog-A-13  
*svcsd\_unregister* routine, *rpc* prog-A-11

## T

TCP use, with *rpc*, example prog-5-2

## U

user-defined type routines, example prog-2-5

## V

version numbers used to define *rpc* procedures  
 prog-2-2

## X

*xdr*, built-in procedures prog-2-4  
*xdr*, example routine prog-3-3  
*xdr*, use in allocating memory, summary  
 prog-3-4  
*xdr*, use in deserializing memory prog-3-4  
*xdr*, use in memory allocation prog-3-3  
*xdr*, use in passing arbitrary data structures  
 prog-2-4  
*xdr*, use in serializing memory prog-3-4  
*xdr*, use with *rpc* prog-2-4  
*xdr\_accepted\_reply*, *rpc* routine prog-A-13  
*xdr\_array* routine, *rpc* prog-A-13  
*xdr\_authunix\_parms*, *rpc* routine prog-A-14  
*xdr\_bool* routine, *rpc* prog-A-14  
*xdr\_bytes* routine, *rpc* prog-A-14  
*xdr\_callhdr* routine, *rpc* prog-A-14  
*xdr\_callmsg* routine, *rpc* prog-A-14  
*xdr\_char* routine, *rpc* prog-A-15  
*xdr\_double* routine, *rpc* prog-A-15  
*xdr\_enum* routine, *rpc* prog-A-15  
*xdr\_float* routine, *rpc* prog-A-15  
*xdr\_hyper* routine, *rpc* prog-A-15  
*xdr\_inline* routine, *rpc* prog-A-16  
*xdr\_int* routine, *rpc* prog-A-16  
*xdr\_long* routine, *rpc* prog-A-16  
*xdr\_opaque* routine, *rpc* prog-A-16  
*xdr\_opaque\_auth*, *rpc* routine prog-A-16  
*xdr\_pmap* routine, *rpc* prog-A-17  
*xdr\_pmaplist* routine, *rpc* prog-A-17  
*xdr\_pointer* routine, *rpc* prog-A-17  
*xdr\_reference* routine, *rpc* prog-A-17  
*xdr\_rejected\_reply*, *rpc* routine prog-A-17  
*xdr\_replymsg* routine, *rpc* prog-A-18  
*xdr\_short* routine, *rpc* prog-A-18  
*xdr\_string* routine, *rpc* prog-A-18  
*xdr\_u\_char*, *rpc* routine prog-A-18  
*xdr\_u\_hyper*, *rpc* routine prog-A-18  
*xdr\_u\_int*, *rpc* routine prog-A-19  
*xdr\_u\_long*, *rpc* routine prog-A-19  
*xdr\_union* routine, *rpc* prog-A-19  
*xdr\_u\_short*, *rpc* routine prog-A-19  
*xdr\_vector* routine, *rpc* prog-A-19  
*xdr\_void* routine, *rpc* prog-A-20  
*xdr\_wrapstring* routine, *rpc* prog-A-20  
*xprt\_register* routine, *rpc* prog-A-20  
*xprt\_unregister* routine, *rpc* prog-A-20



# Index

*clnt\_create*, *rpc* routines prog-A-3

## A

abstract data types xdr-4-1  
*accept* procedure over-1-3  
access control, and *yp* yp-2-3  
*Add Mount Entry* mount server procedure  
nfs-3-3  
administering server machines over-1-6  
advanced topics xdr-6-1  
allocating memory with *xdr*, example  
prog-3-4  
*Answer Only If You Serve This Domain* pro-  
cedure, *yp* yp-2-8  
array handling functions, *xdr* xdr-2-4  
array handling primitives, *xdr* xdr-2-3  
arrays, counted xdr-5-4  
arrays, fixed-size xdr-5-3  
*attrstat* structure, version 2 nfs-2-7  
*auth\_destroy* routine, *rpc* prog-A-1  
authentication, caller to service, *rpc* rpc-2-2  
authentication credentials structure, *rpc*  
prog-4-6  
authentication handle, *rpc* prog-4-6  
authentication, message, and *rpc* rpc-1-2  
authentication, null, *rpc* rpc-3-4  
authentication of *rpc* calls prog-4-6  
authentication of *rpc* calls, calling side  
prog-4-6  
authentication of *rpc* calls, server side  
prog-4-7  
authentication, *rpc*, types rpc-3-4  
authentication, UNIX, *rpc* rpc-3-4  
*authnone\_create* routine, *rpc* prog-A-1  
*authunix\_create* routine, *rpc* prog-A-1  
*authunix\_create\_default* routine, *rpc* prog-A-1

## B

basic block size, *xdr* standard xdr-5-1  
basic data types, version 2 nfs-2-3  
batch call attributes, with *rpc* prog-4-4  
batching, *rpc* prog-4-2  
batching, *rpc*, example prog-4-3  
batching, using *rpc* rpc-2-3  
*bind* procedure over-1-3  
binders, *yp*, data structures used yp-3-2  
binders, *yp*, design assumptions yp-3-1  
binders, *yp*, overview yp-3-1  
binders, *yp*, protocol definition yp-3-1  
binders, *yp*, remote procedures yp-3-3  
binding, client, and *rpc* rpc-1-2  
binding, dynamic, with *rpc* rpc-A-1  
bit fields, specifications xdr-5-4  
bit maps, specifications xdr-5-4  
boolean, data definition yp-1-2  
Boolean data type, data definition xdr-5-2  
Boolean data type definition, *xdr* standard  
xdr-5-2  
boolean definition, *xdr* standard yp-1-2  
*bool\_t xdr\_array* library primitives xdr-2-4  
*bool\_t xdr\_bool* library primitives xdr-2-2  
*bool\_t xdr\_char* library primitives xdr-2-1

*bool\_t xdr\_discrim* library primitives xdr-2-7  
*bool\_t xdr\_double* library primitives xdr-2-2  
*bool\_t xdr\_enum* library primitives xdr-2-2  
*bool\_t xdr\_float* library primitives xdr-2-2  
*bool\_t xdr\_hyper* library primitives xdr-2-1  
*bool\_t xdr\_int* library primitives xdr-2-1  
*bool\_t xdr\_long* library primitives xdr-2-1  
*bool\_t xdr\_opaque* library primitives xdr-2-6  
*bool\_t xdr\_reference* library primitives  
xdr-2-9  
*bool\_t xdr\_setpos* library primitives xdr-2-10  
*bool\_t xdr\_short* library primitives xdr-2-1  
*bool\_t xdr\_string* library primitives xdr-2-3  
*bool\_t xdr\_u\_char* library primitives xdr-2-1  
*bool\_t xdr\_u\_hyper* library primitives xdr-2-2  
*bool\_t xdr\_u\_int* library primitives xdr-2-1,  
xdr-2-4  
*bool\_t xdr\_u\_long* library primitives xdr-2-1  
*bool\_t xdr\_u\_short* library primitives xdr-2-1  
*bool\_t xdr\_void* library primitives xdr-2-3  
broadcast *rpc* prog-4-1, rpc-2-3  
broadcast *rpc*, synopsis prog-4-2  
built-in *xdr* procedures, use in serializing data  
structures prog-2-4  
byte array handling routines, *xdr* xdr-2-4  
byte arrays, vs. strings xdr-2-4

## C

C library primitives vs. *xdr* standard data  
types xdr-5-5  
callback procedures, via *rpc* prog-5-5  
callback procedures, via *rpc*, example  
prog-5-5  
caller to service authentication, *rpc* rpc-2-2  
*callrpc* prog-1-1, prog-2-2, prog-3-4  
*callrpc*, example procedure prog-2-3  
*callrpc*, parameters used with prog-2-2  
*callrpc*, return codes prog-2-3  
*callrpc* routine, *rpc* prog-A-2  
*callrpc*, use with built-in procedures prog-2-4  
client binding, and *rpc* rpc-1-2  
client, defined over-1-3, rpc-1-1  
client handles, *rpc*, example prog-4-6  
clients, *yp* over-3-2  
clients, *yp*, operation over-4-2  
*clnt\_broadcast* routine, *rpc* prog-A-2  
*clnt\_call* routine, *rpc* prog-A-2  
*clnt\_control*, *rpc* routines prog-A-3  
*clnt\_destroy* routine, *rpc* prog-A-4  
*clnt\_freeres* routine, *rpc* prog-A-4  
*clnt\_geterr* routine, *rpc* prog-A-4  
*clnt\_pcreateerror* routine, *rpc* prog-A-4  
*clnt\_perrno* routine, *rpc* prog-A-4  
*clnt\_perror* routine, *rpc* prog-A-5  
*clntraw\_create* routine, *rpc* prog-A-6  
*clnt\_screateerror* routine, *rpc* prog-A-5  
*clnt\_sperrno* routine, *rpc* prog-A-5  
*clnt\_sperror* routine, *rpc* prog-A-5  
*clnt\_syslog* routine, *rpc* prog-A-5  
*clnttcp\_create* routine, *rpc* prog-A-6  
*clntudp\_create* routine, *rpc* prog-A-6  
compiling *xdr* routines xdr-1-1

computing environment, network, illustrated  
     over-1-2  
 computing environment, traditional, illus-  
     trated over-1-2  
 computing environments, pros and cons  
     over-1-1, over-1-2  
 configuration trade-offs, *nfs* over-2-2  
 constants, manifest *yp*-3-2  
 constants, manifest, used with *yp* *yp*-2-3  
 constants, *rpc* *yp*-3-1  
 constants, *rpc*, needed to call *nfs* *nfs*-2-3  
 constants, *rpc*, used with *yp* *yp*-2-3  
 constructed data type filters, *xdr* *xdr*-2-3  
 constructed data types, *xdr*, examples *xdr*-2-4  
*COOKIESIZE* structure, version 2 *nfs*-2-3  
 counted arrays, data definition *xdr*-5-4  
 counted byte string definition, *xdr* standard  
     *xdr*-5-3  
 counted byte strings, data definition *xdr*-5-3  
*Create Directory* server procedure *nfs*-2-13  
*Create File* server procedure *nfs*-2-11  
*Create Link to File* server procedure *nfs*-2-12  
*Create Symbolic Link* server procedure  
     *nfs*-2-12

## D

data definition language, *xdr* *rpc*-3-1, *xdr*-5-1,  
     *yp*-1-2, *yp*-3-3  
 data representations, *xdr* *yp*-1-2  
 data structures, and *yp* remote procedures  
     *yp*-2-4  
 data structures, arbitrary, and portability  
     problems *xdr*-1-3  
 data structures, used with *rpc* *yp*-1-2  
 data structures, used with *yp* binders *yp*-3-2  
 data types, abstract *xdr*-4-1  
 data types, basic, version 2 *nfs*-2-3  
 data types, *xdr*, vs. C library primitives  
     *xdr*-5-5  
*dbm(3)* over-4-2  
*dbm(3)*, and the implementation of *yp* maps  
     over-3-1  
 debugging, remote, via *rpc* *prog*-5-5  
 default *yp* files over-4-2  
 delimiting records *xdr*-3-2  
 deserialization *prog*-3-4, *prog*-5-2, *xdr*-1-4  
 deserialization of data to or from standard  
     I/O *xdr*-3-1  
 deserializing arrays *xdr*-2-4  
 deserializing byte areas *xdr*-2-4  
 deserializing data structures *prog*-2-4,  
     *prog*-2-6, *prog*-3-1  
 deserializing discriminated unions *xdr*-2-8  
 deserializing linked lists *xdr*-6-2  
 deserializing linked lists, side effects *xdr*-6-3  
 deserializing strings *xdr*-2-3  
*df(1)* over-1-4  
*diopargs* data structure, version 2 *nfs*-2-7  
 direction independence, and *xdr* routines  
     *xdr*-1-4  
*diopres* data structure, version 2 *nfs*-2-7

*dirpath* data type *nfs*-3-2  
 discriminated union construct *yp*-1-2  
 discriminated union, example *yp*-1-2  
 discriminated unions, data definition *xdr*-5-4  
 discriminated unions, *xdr* *xdr*-2-7, *xdr*-2-9  
 discriminated unions, *xdr*, deserializing  
     *xdr*-2-8  
 discriminated unions, *xdr*, example *xdr*-2-8  
 distributed filesystem, benefits over-1-1  
*Do Nothing* mount server procedure *nfs*-3-3  
*Do Nothing* procedure, *rpc* *rpc*-A-1  
*Do Nothing* procedure, *yp* *yp*-2-7  
*Do Nothing* procedure, *yp* binder remote  
     *yp*-3-3  
*Do Nothing* server procedure *nfs*-2-8  
*Do You Serve This Domain?* procedure, *yp*  
     *yp*-2-8  
*domainname* data structure *yp*-2-5  
*domainname* data structure, used with *yp*  
     binders *yp*-3-2  
*domainname(1)* over-3-1, over-4-1  
 domains and *yp*, relevance over-4-1  
 domains, defined *yp*-1-1  
 domains, overview *yp*-2-2  
 domains, *yp*, example over-3-1  
 double precision floating-point definition, *xdr*  
     standard *xdr*-5-2  
*Dumping the Mappings* procedure, *rpc*  
     *rpc*-A-3  
 dynamic binding with *rpc* *rpc*-A-1

## E

*enum xdr\_op* structure *xdr*-4-1  
 enumeration definition, *xdr* definition *xdr*-5-1  
 enumeration filters, *xdr* *xdr*-2-2  
 enumerations, data definition *xdr*-5-1  
 error conditions, *rpc* *rpc*-2-1  
*/etc/ethers* over-4-2, over-4-3  
*/etc/exports* over-1-5, over-1-6, over-2-4  
*/etc/exports*, modifying to export filesystems  
     over-1-5  
*/etc/group* over-4-2  
*/etc/group*, and use with *yp* maps over-3-1  
*/etc/hosts* over-4-1, over-4-2  
*/etc/hosts*, and use with *yp* maps over-3-1  
*/etc/inetd.conf*, entry format for *rpc* services  
     *prog*-4-9  
*/etc/netgroup* over-4-3  
*/etc/networks* over-4-1, over-4-2, over-4-3  
*/etc/networks*, and use with *yp* maps  
     over-3-1  
*/etc/passwd* over-2-1, over-4-2  
*/etc/passwd*, and use with *yp* maps over-3-1  
*/etc/protocols* over-4-2, over-4-3  
*/etc/pwrestrict* over-4-2  
*/etc/rc.local* over-3-1  
*/etc/services* over-4-2, over-4-3  
*/etc/termcap* *prog*-4-6  
*/etc/yp* over-3-1  
*ether* service *prog*-2-2  
 exporting a filesystem, procedure over-1-5

## F

*fattr* data structure, version 2 nfs-2-5  
*fattr* data type, version 2, *mode* bit positions  
 nfs-2-5  
*fhandle* data type, used with mount protocol  
 nfs-3-1  
*fhandle* data type, version 2 nfs-2-5, nfs-3-2  
*FHSIZE 32* structure, version 2 nfs-2-3  
*fhstatus* data type nfs-3-2  
 file protection, in *nfs* implementation  
 over-2-5  
*filename* data type, version 2 nfs-2-7  
 filesystem data, defined over-1-3  
 filesystem data vs. filesystem operations  
 over-1-3  
 filesystem location transparency, with *nfs*  
 over-2-4  
 filesystem operations, defined over-1-3  
 filesystem operations *not* supported, in *nfs*  
 implementation over-2-5  
 filesystem transparency, with *nfs* over-2-3  
 fixed-size arrays, data definition xdr-5-3  
 fixed-sized arrays xdr-2-7  
 floating library primitives, *xdr* xdr-2-2  
 floating-point definition, *xdr* standard xdr-5-2  
 floating-point exponent field, *xdr* standard  
 xdr-5-2  
 floating-point mantissa field, *xdr* standard  
 xdr-5-2  
 floating-point sign field, *xdr* standard xdr-5-2  
*fscanf(3c)* prog-4-6  
*ftp(1)*, overview over-1-3  
*ftp(1)*, shortcomings over-1-4  
*ftp(1)*, vs. *nfs* over-1-4  
*ftype* data type, version 2 nfs-2-4

## G

*Get All Key-Value Pairs in Map* procedure, *yp*  
 yp-2-11  
*Get All Maps in Domain* procedure, *yp*  
 yp-2-11  
*Get Current Binding for a Domain* procedure,  
*yp* binder remote yp-3-4  
*Get File Attributes* server procedure nfs-2-8  
*Get Filesystem Attributes* server procedure  
 nfs-2-14  
*Get Filesystem Root* server procedure nfs-2-9  
*Get First Key-Value Pair in Map* procedure, *yp*  
 yp-2-9  
*Get Map Master Name* procedure, *yp* yp-2-11  
*Get Map Order Number* procedure, *yp*  
 yp-2-11  
*Get Next Key-Value Pair in Map* procedure, *yp*  
 yp-2-9  
*getdomainname(2)* over-4-1  
*getgrent(3)* over-4-2  
*gethostbyaddr(3)* over-4-3  
*gethostbyname(3)* over-4-3  
*gethostent(3)* over-4-2  
*gethostname(2)* over-4-1  
*get\_myaddress* routine, *rpc* prog-A-7

*getpwent(3)* over-4-2  
*getpwrestent(3)* over-4-2  
*gettransient* routine, using, example prog-5-6  
*gid*, and protocol permission issues nfs-2-2

## H

hard links nfs-2-12  
*hostname(1)* over-4-1  
*hosts* file, *yp* over-4-3  
*hosts.byadr* over-4-3  
*hosts.byname* over-4-3  
 hyper integer definition, *xdr* standard xdr-5-2  
 hyper unsigned definition, *xdr* standard  
 xdr-5-2

## I

*Indirect Call* procedure, *rpc* rpc-A-3  
*inetd*, using to start *rpc* servers prog-4-9  
 inode, defined over-1-3  
 integer definition, *xdr* standard xdr-5-1  
 integers, data definition xdr-5-1  
 integers, unsigned xdr-5-1

## K

*keydat* data structure yp-2-5  
 keys, defined yp-2-1

## L

library primitives, *xdr*, synopsis xdr-2-1  
 library routines, rewritten for *yp* over-4-2  
 library, *xdr* xdr-1-4  
*librpcsvc.a* prog-2-1  
 linked list, data description xdr-6-1  
 linked lists, data structures used to implement,  
 example xdr-6-1  
 linked lists, (de)serializing xdr-6-2  
 linked lists, (de)serializing, side effects  
 xdr-6-3  
 linked lists, passing, example xdr-6-1  
 links, hard nfs-2-12  
 links, symbolic nfs-2-12  
 locking maps yp-2-3  
*Look Up a Mapping* procedure, *rpc* rpc-A-2  
*Look Up File Name* server procedure nfs-2-9  
*LOOKUP* procedure, *nfs* protocol nfs-2-1

## M

machine type transparency, with *nfs* over-2-4  
*makedbm(3)* over-3-2  
*makedbm(8)*, operation explained over-4-2  
*malloc(3)* prog-1-1  
 manifest constants yp-3-2  
 manifest constants used with *yp* yp-2-3  
 map entry enumeration yp-2-1  
 map propagation yp-2-2, yp-2-3  
 map propagation, hints yp-2-2  
 map retrieval yp-2-1  
 map structure yp-2-1  
 map updates yp-2-1  
*mapname* data structure yp-2-5

mapname, *yp*, defined over-3-1  
 maps, defined yp-2-1  
 maps, locking yp-2-3  
 maps, relationship to *yp* yp-2-1  
 maps, typical applications yp-2-1  
 maps, updating yp-2-2, yp-2-3  
 maps, *yp*, defined over-3-1  
 master servers, *yp*, defined over-3-2, yp-2-2  
 match operation, *yp* yp-2-1  
 MAXDATA structure, version 2 nfs-2-3  
 MAXNAMLEN structure, version 2 nfs-2-3  
 MAXPATHLEN structure, version 2 nfs-2-3  
 memory allocation with *xdr* prog-3-3  
 memory allocation with *xdr*, example routine  
 prog-3-3, prog-3-4  
 memory streams, creating xdr-3-1  
 message authentication, and *rpc* rpc-1-2  
 mount protocol, and *rpc* nfs-3-1  
 mount protocol, overview nfs-3-1  
 mount protocol, *rpc* constants needed to call  
 nfs-3-1  
 mount protocol, version 1 attributes nfs-3-1  
 mount protocol, *xdr* structure sizes used  
 nfs-3-1  
 mount server procedures, *Add Mount Entry*  
 nfs-3-3  
 mount server procedures, *Do Nothing* nfs-3-3  
 mount server procedures, *Remove All Mount*  
*Entries* nfs-3-4  
 mount server procedures, *Remove Mount Entry*  
 nfs-3-4  
 mount server procedures, *Return Export List*  
 nfs-3-4  
 mount server procedures, *Return Mount*  
*Entries* nfs-3-3  
 mount servers, *rpc* procedures used nfs-3-2  
 mount service prog-2-2  
 mount(8) over-2-4  
 mountd(8c) over-1-5  
 mounting remote filesystems over-1-4  
 multiple program versions, C procedures used  
 to implement prog-5-1  
 multiple program versions, supporting  
 prog-5-1  
 multi-threading, and *rpc* rpc-1-1

## N

name data type nfs-3-2  
 network administration and *yp* over-2-1  
 network paging, permission problems associ-  
 ated nfs-2-2  
 network "pipes" xdr-1-2  
 network transparency, with *nfs* over-2-4  
*nfs*, configuration trade-offs over-2-2  
*nfs*, design and implementation over-2-1  
*nfs*, design features, ease of administration  
 over-2-1  
*nfs*, design features, extensibility over-2-1  
*nfs*, design features, open system approach  
 over-2-1  
*nfs*, design features, performance over-2-2

*nfs*, design features, reliability over-2-2  
*nfs*, design features, transparent information  
 access over-2-1  
*nfs*, examples of operation over-1-4  
*nfs*, file system interface over-2-2  
*nfs* implementation over-2-2  
*nfs* implementation, file protection in  
 over-2-5  
*nfs* implementation, filesystem operations *not*  
 supported over-2-5  
*nfs* implementation, types of transparency  
 over-2-3  
*nfs* interface, filesystem operations defined  
 over-2-4  
*nfs*, operating system interface over-2-2  
*nfs*, overview nfs-1-1, over-1-1, over-1-3  
*nfs*, performance goals over-2-2  
*nfs*, performance improvements over-2-2  
*nfs* protocol, characteristics nfs-2-1  
*nfs* protocol definition, overview nfs-2-1  
*nfs* protocol, LOOKUP procedure nfs-2-1  
*nfs* protocol, permission issues nfs-2-2  
*nfs* protocol, READDIR procedure nfs-2-1  
*nfs* protocol, similarities with UNIX nfs-2-1  
*nfs* protocol, version 2 nfs-2-1  
*nfs* protocol, version 2, basic data types  
 nfs-2-3  
*nfs*, virtual file system interface over-2-2  
*nfs*, vs. *rcp* and *ftp* over-1-3  
 no data routines, *xdr* xdr-2-3  
 non-filter primitives, *xdr* xdr-2-10  
 null authentication, *rpc* rpc-3-4  
 number filters, *xdr* xdr-2-1

## O

opaque data definition, *xdr* standard xdr-5-3  
 opaque handles, *xdr* xdr-2-6  
*open* calls, permission problems associated  
 nfs-2-2  
 operating system transparency, with *nfs*  
 over-2-4  
 operation directions, *xdr* xdr-2-10

## P

paging, network, permission problems associ-  
 ated nfs-2-2  
*passwd.byname* over-4-3  
*passwd.byuid* over-4-3  
 password, changing with *yppasswd(1)*  
 over-4-3  
*path* data type, version 2 nfs-2-7  
*peername* data structure yp-2-5  
 permission issues, *nfs* protocol nfs-2-2  
 permission problems, with *open* calls nfs-2-2  
 pipes, network xdr-1-2  
*pmap\_getmaps* routine, *rpc* prog-A-7  
*pmap\_getport* routine, *rpc* prog-A-7  
*pmap\_rmtcall* routine, *rpc* prog-A-7  
*pmap\_set* routine, *rpc* prog-A-8  
*pmap\_unset* routine, *rpc* prog-A-8  
 pointer handling routines, *xdr* xdr-2-3

- pointer semantics, and *xdr* xdr-2-9
  - pointers, *xdr* xdr-2-9
  - pointers, *xdr*, example xdr-2-9
  - port mapper program protocol, *rpc* rpc-A-1
  - port numbers, determination under *rpc* prog-3-2
  - port numbers, reserved rpc-A-1
  - portability, and arbitrary data structures xdr-1-3
  - portability, *xdr* library as solution xdr-1-4
  - portable data, and *xdr* xdr-1-2
  - portmap*(8) prog-4-1
  - portmapper* prog-4-1
  - primitives, C library, vs. *xdr* standard data types xdr-5-5
  - procedure number, remote, and *rpc* rpc-2-1
  - procedure numbers yp-1-1
  - procedure numbers, example of use prog-3-3
  - procedure numbers used to define *rpc* procedures prog-2-2
  - procedures, defined rpc-1-1
  - program number, remote, and *rpc* rpc-2-1
  - program numbers yp-1-1
  - program numbers, and *rpc* rpc-2-2
  - program numbers, assigning to *rpc* highest layer prog-2-4
  - program numbers used to define *rpc* procedures prog-2-2
  - program numbers used with callback procedures prog-5-5
  - program version number, remote, and *rpc* rpc-2-1
  - programs, defined rpc-1-1
  - propagating maps yp-2-2, yp-2-3
  - protocol definition, *rpc* rpc-3-1
  - protocol definition, *yp* binders yp-3-1
  - protocol definition, *yp* database server yp-2-3
  - protocol requirements, *rpc* rpc-2-1
  - protocol specifications, registering prog-2-4
- R**
- rcp*(1), overview over-1-3
  - rcp*(1), shortcomings over-1-3
  - rcp*(1), vs. *nfs* over-1-3
  - rcs*(1), using with exported filesystems over-1-6
  - Read From Directory* server procedure nfs-2-13
  - Read From File* server procedure nfs-2-10
  - Read From Symbolic Link* server procedure nfs-2-10
  - READDIR* procedure, *nfs* protocol nfs-2-1
  - record fragment, defined xdr-6-5
  - record fragments, *rpc* rpc-3-5
  - record marking, *rpc* rpc-3-5
  - record streams, creating xdr-3-2
  - record-marking standard xdr-6-5
  - records, defined xdr-6-5
  - records, delimiting xdr-3-2
  - records, *rpc* rpc-3-5
  - registering protocol specifications prog-2-4
  - registerrpc* prog-1-1, prog-2-2, prog-2-3, prog-3-2
  - registerrpc* routine, *rpc* prog-A-8
  - registerrpc*, use with built-in procedures prog-2-4
  - Reinitialize Internal State* procedure, *yp* yp-2-10
  - remote debugging, via *rpc* prog-5-5
  - remote procedure number, and *rpc* rpc-2-1
  - remote procedures, *yp* binder yp-3-3
  - remote program number, and *rpc* rpc-2-1
  - remote program version number, and *rpc* rpc-2-1
  - remote users service, example prog-4-8
  - Remove All Mount Entries* mount server procedure nfs-3-4
  - Remove Directory* server procedure nfs-2-13
  - Remove File* server procedure nfs-2-11
  - Remove Mount Entry* mount server procedure nfs-3-4
  - Rename File* server procedures nfs-2-12
  - rendering strings, via *rpc* batching prog-4-4
  - request handle, *rpc* prog-4-7
  - reserved port numbers rpc-A-1
  - Return Export List* mount server procedure nfs-3-4
  - Return Mount Entries* mount server procedure nfs-3-3
  - return status values, *yp* remote procedures yp-2-4
  - Return Value of a Key* procedure, *yp* yp-2-8
  - rnusers* prog-1-1, prog-2-1
  - rpc* over-2-3, over-2-4, rpc-1-1
  - rpc*, and client binding rpc-1-2
  - rpc*, and message authentication rpc-1-2
  - rpc*, and mount protocol nfs-3-1
  - rpc*, and multi-threading rpc-1-1
  - rpc* and remote debugging prog-5-5
  - rpc*, and *vfs* interface over-2-3
  - rpc* authentication credentials structure prog-4-6
  - rpc* authentication handle prog-4-6
  - rpc*, authentication parameters nfs-1-1
  - rpc* authentication, types rpc-3-4
  - rpc*, batching prog-4-2
  - rpc*, batching, example prog-4-3
  - rpc*, broadcast prog-4-1, rpc-2-3
  - rpc*, broadcast, synopsis prog-4-2
  - rpc* callback procedure, example prog-5-5
  - rpc* callback remotes prog-5-5
  - rpc* calls, authentication of prog-4-6
  - rpc* calls, authentication of, on calling side prog-4-6
  - rpc* calls, authentication of, server side prog-4-7
  - rpc* calls, *callrpc* prog-1-1
  - rpc* calls, *registerrpc* prog-1-1
  - rpc* calls, *rnusers* prog-1-1
  - rpc* client handle, example prog-4-6
  - rpc* constants yp-3-1
  - rpc* constants, needed to call mount protocol

- nfs-3-1
- rpc constants needed to call *nfs* nfs-2-3
- rpc constants, used with *yp* yp-2-3
- rpc, data structures used yp-1-2
- rpc, determination of port numbers prog-3-2
- rpc, differences between broadcast and normal prog-4-1
- rpc, error conditions rpc-2-1
- rpc, example prog-3-5
- rpc, features not supported rpc-2-1
- rpc, highest layer prog-1-1, prog-2-1
- rpc, highest layer, assigning program numbers prog-2-4
- rpc information, version 2 nfs-2-2
- rpc, layers prog-1-1
- rpc, lowest layer prog-1-1, prog-3-1
- rpc, lowest layers, example prog-3-1, prog-3-3
- rpc, lowest layers, occasions for use prog-3-1
- rpc, middle layer prog-1-1, prog-2-2
- rpc, middle layer, example of use prog-2-2
- rpc, overview nfs-1-1, over-1-3, prog-1-1
- rpc, paradigm prog-1-2, yp-1-1
- rpc, passing arbitrary data structures prog-2-4
- rpc, port mapper program protocol rpc-A-1
- rpc procedures, *Do Nothing* rpc-A-1
- rpc procedures, *Dumping the Mappings* rpc-A-3
- rpc procedures, *Indirect Call* rpc-A-3
- rpc procedures, listed rpc-A-1
- rpc procedures, *Look Up a Mapping* rpc-A-2
- rpc procedures, *Set a Mapping* rpc-A-2
- rpc procedures, *Unset a Mapping* rpc-A-2
- rpc, procedures used with mount servers nfs-3-2
- rpc, program numbers used rpc-2-2
- rpc protocol definition rpc-3-1
- rpc protocol requirements rpc-2-1
- rpc remote users service example prog-4-8
- rpc request handle prog-4-7
- rpc routines, *auth\_destroy* prog-A-1
- rpc routines, *authnone\_create* prog-A-1
- rpc routines, *authunix\_create* prog-A-1
- rpc routines, *authunix\_create\_default* prog-A-1
- rpc routines, *callrpc* prog-A-2
- rpc routines, *clnt\_broadcast* prog-A-2
- rpc routines, *clnt\_call* prog-A-2
- rpc routines, *clnt\_control* prog-A-3
- rpc routines, *clnt\_create* prog-A-3
- rpc routines, *clnt\_destroy* prog-A-4
- rpc routines, *clnt\_freeres* prog-A-4
- rpc routines, *clnt\_geterr* prog-A-4
- rpc routines, *clnt\_pcreateerror* prog-A-4
- rpc routines, *clnt\_perrno* prog-A-4
- rpc routines, *clnt\_perror* prog-A-5
- rpc routines, *clntraw\_create* prog-A-6
- rpc routines, *clnt\_spcreateerror* prog-A-5
- rpc routines, *clnt\_sperrno* prog-A-5
- rpc routines, *clnt\_sperror* prog-A-5
- rpc routines, *clnt\_syslog* prog-A-5
- rpc routines, *clnttcp\_create* prog-A-6
- rpc routines, *clntudp\_create* prog-A-6
- rpc routines, *get\_myaddress* prog-A-7
- rpc routines, *pmap\_getmaps* prog-A-7
- rpc routines, *pmap\_getport* prog-A-7
- rpc routines, *pmap\_rmtcall* prog-A-7
- rpc routines, *pmap\_set* prog-A-8
- rpc routines, *pmap\_unset* prog-A-8
- rpc routines, *registerrpc* prog-A-8
- rpc routines, *rpc\_createerr* prog-A-8
- rpc routines, *svc\_destroy* prog-A-8
- rpc routines, *svcerr\_auth* prog-A-11
- rpc routines, *svcerr\_decode* prog-A-11
- rpc routines, *svcerr\_noproc* prog-A-11
- rpc routines, *svcerr\_noprogram* prog-A-11
- rpc routines, *svcerr\_progmisc* prog-A-12
- rpc routines, *svcerr\_systemerr* prog-A-12
- rpc routines, *svcerr\_weakauth* prog-A-12
- rpc routines, *svcfld\_create* prog-A-13
- rpc routines, *svc\_fds* prog-A-9
- rpc routines, *svc\_fdset* prog-A-9
- rpc routines, *svc\_freeargs* prog-A-9
- rpc routines, *svc\_getargs* prog-A-9
- rpc routines, *svc\_getcaller* prog-A-9
- rpc routines, *svc\_getreq* prog-A-9
- rpc routines, *svc\_getreqset* prog-A-10
- rpc routines, *svcrw\_create* prog-A-12
- rpc routines, *svc\_register* prog-A-10
- rpc routines, *svc\_run* prog-A-10
- rpc routines, *svc\_sendreply* prog-A-10
- rpc routines, *svctcp\_create* prog-A-12
- rpc routines, *svcudp\_create* prog-A-13
- rpc routines, *svc\_unregister* prog-A-11
- rpc routines, *xdr\_accepted\_reply* prog-A-13
- rpc routines, *xdr\_array* prog-A-13
- rpc routines, *xdr\_authunix\_parms* prog-A-14
- rpc routines, *xdr\_bool* prog-A-14
- rpc routines, *xdr\_bytes* prog-A-14
- rpc routines, *xdr\_callhdr* prog-A-14
- rpc routines, *xdr\_callmsg* prog-A-14
- rpc routines, *xdr\_char* prog-A-15
- rpc routines, *xdr\_double* prog-A-15
- rpc routines, *xdr\_enum* prog-A-15
- rpc routines, *xdr\_float* prog-A-15
- rpc routines, *xdr\_hyper* prog-A-15
- rpc routines, *xdr\_inline* prog-A-16
- rpc routines, *xdr\_int* prog-A-16
- rpc routines, *xdr\_long* prog-A-16
- rpc routines, *xdr\_opaque* prog-A-16
- rpc routines, *xdr\_opaque\_auth* prog-A-16
- rpc routines, *xdr\_pmap* prog-A-17
- rpc routines, *xdr\_pmaplist* prog-A-17
- rpc routines, *xdr\_pointer* prog-A-17
- rpc routines, *xdr\_reference* prog-A-17
- rpc routines, *xdr\_rejected\_reply* prog-A-17
- rpc routines, *xdr\_replymsg* prog-A-18
- rpc routines, *xdr\_short* prog-A-18
- rpc routines, *xdr\_string* prog-A-18
- rpc routines, *xdr\_u\_char* prog-A-18
- rpc routines, *xdr\_u\_hyper* prog-A-18
- rpc routines, *xdr\_u\_int* prog-A-19

*rpc* routines, *xdr\_u\_long* prog-A-19  
*rpc* routines, *xdr\_union* prog-A-19  
*rpc* routines, *xdr\_u\_short* prog-A-19  
*rpc* routines, *xdr\_vector* prog-A-19  
*rpc* routines, *xdr\_void* prog-A-20  
*rpc* routines, *xdr\_wrapstring* prog-A-20  
*rpc* routines, *xprt\_register* prog-A-20  
*rpc* routines, *xprt\_unregister* prog-A-20  
*rpc*, semantics rpc-1-2  
*rpc* servers, starting with *inetd* prog-4-9  
*rpc*, service library routines available prog-2-1  
*rpc* services, */etc/inetd.conf* entry format prog-4-9  
*rpc* services library, *librpcsvc.a* prog-2-1  
*rpc*, string rendering service, example prog-4-3  
*rpc* transport independence rpc-1-2  
*rpc*, use in batching rpc-2-3  
*rpc*, use in dynamic binding rpc-A-1  
*rpc*, use with TCP, example prog-5-2  
*rpc*, use with *xdr* prog-2-4  
*rpc*, *yp* interface yp-1-2  
*rpc\_createerr* routine, *rpc* prog-A-8  
*rpc/rpc.h* xdr-1-1  
*quota* service prog-2-2

## S

*sattr* data structure, version 2 nfs-2-6  
*select* procedure over-1-3  
*select*, use with *rpc*, example prog-4-1  
 semantics, *rpc* rpc-1-2  
 serialization prog-3-4, prog-5-2, xdr-1-4  
 serialization of data to or from standard I/O xdr-3-1  
 serializing arrays xdr-2-4  
 serializing byte areas xdr-2-4  
 serializing data structures prog-2-4, prog-2-6, prog-3-1, prog-3-3  
 serializing linked lists xdr-6-2  
 serializing linked lists, side effects xdr-6-3  
 serializing null-value pointers xdr-2-10  
 serializing objects xdr-6-1  
 server, defined over-1-1, over-1-3, rpc-1-1  
 server procedures, version 2, *Create Directory* nfs-2-13  
 server procedures, version 2, *Create File* nfs-2-11  
 server procedures, version 2, *Create Link to File* nfs-2-12  
 server procedures, version 2, *Create Symbolic Link* nfs-2-12  
 server procedures, version 2, defined nfs-2-8  
 server procedures, version 2, *Do Nothing* nfs-2-8  
 server procedures, version 2, *Get File Attributes* nfs-2-8  
 server procedures, version 2, *Get Filesystem Attributes* nfs-2-14  
 server procedures, version 2, *Get Filesystem Root* nfs-2-9  
 server procedures, version 2, *Look Up File Name* nfs-2-9  
 server procedures, version 2, *Read From Directory* nfs-2-13  
 server procedures, version 2, *Read From File* nfs-2-10  
 server procedures, version 2, *Read From Symbolic Link* nfs-2-10  
 server procedures, version 2, *Remove Directory* nfs-2-13  
 server procedures, version 2, *Remove File* nfs-2-11  
 server procedures, version 2, *Rename File* nfs-2-12  
 server procedures, version 2, *Set File Attributes* nfs-2-9  
 server procedures, version 2, *Write to Cache* nfs-2-10  
 server procedures, version 2, *Write to File* nfs-2-11  
 server/client relationship, *nfs* protocol, version 2 nfs-2-1  
 servers, master vs. slave yp-2-2  
 servers, *yp* over-3-2  
 service dispatch routines, guaranteed *rpc* functionality prog-4-7  
 services, defined rpc-1-1  
*Set a Mapping* procedure, *rpc* rpc-A-2  
*Set Domain Binding* procedure, *yp* binder remote yp-3-4  
*Set File Attributes* server procedure nfs-2-9  
*showmount(8)* over-1-6  
 slave servers, defined yp-2-2  
 slave servers, *yp*, defined over-3-2  
*spray* service prog-2-2  
 standard I/O streams, *xdr* xdr-3-1  
*stat* data type, version 2 nfs-2-3  
*stat* data type, version 2, error numbers nfs-2-4  
 stateless server, advantages over-2-2, over-2-5  
 stateless server, defined over-2-4  
 stream access, *xdr*, introduction xdr-3-1  
 stream creation routines, in *xdr* library xdr-1-4  
 streams, implementing xdr-4-1  
 streams, interface, structure used xdr-4-1  
 streams, interface to xdr-4-1  
 string handling primitives, *xdr* xdr-2-3  
 string handling routines, *xdr* xdr-2-3  
 string handling routines, *xdr*, effect of deserializing xdr-2-3  
 string rendering service, *rpc* example prog-4-3  
 string rendering, via *rpc* batching, performance results prog-4-6  
 strings, counted byte, data definition xdr-5-3  
 strings, counted byte, defined xdr-5-3  
 strings, data definition yp-1-2  
 strings, rendering via *rpc* batching prog-4-4  
 strings, vs. byte arrays xdr-2-4

structure sizes, version 2 nfs-2-3  
 structures, data definition xdr-5-4  
*svc\_destroy* routine, *rpc* prog-A-8  
*svcerr\_auth* routine, *rpc* prog-A-11  
*svcerr\_decode* routine, *rpc* prog-A-11  
*svcerr\_noproc* routine, *rpc* prog-A-11  
*svcerr\_noprog* routine, *rpc* prog-A-11  
*svcerr\_progvers* routine, *rpc* prog-A-12  
*svcerr\_systemerr* routine, *rpc* prog-A-12  
*svcerr\_weakauth* routine, *rpc* prog-A-12  
*svcsd\_create* routine, *rpc* prog-A-13  
*svc\_fds* routine, *rpc* prog-A-9  
*svc\_fdset* routine, *rpc* prog-A-9  
*svc\_freeargs* routine, *rpc* prog-A-9  
*svc\_getargs* routine, *rpc* prog-A-9  
*svc\_getcaller* routine, *rpc* prog-A-9  
*svc\_getreq* routine, *rpc* prog-A-9  
*svc\_getreqset* routine, *rpc* prog-A-10  
*svccraw\_create* routine, *rpc* prog-A-12  
*svc\_register* routine, *rpc* prog-A-10  
*svc\_run* routine, *rpc* prog-A-10  
*svc\_sendreply* routine, *rpc* prog-A-10  
*svtcp\_create* routine, *rpc* prog-A-12  
*svcupd\_create* routine, *rpc* prog-A-13  
*svc\_unregister* routine, *rpc* prog-A-11  
 symbolic links nfs-2-12

## T

TCP use, with *rpc*, example prog-5-2  
 terminology, basic rpc-1-1  
*timeval* data structure, version 2 nfs-2-5  
*Transfer Map* procedure, *yp* yp-2-10  
 transparency, filesystem location, with *nfs*  
     over-2-4  
 transparency, filesystem, with *nfs* over-2-3  
 transparency, in *nfs* implementation over-2-3  
 transparency, machine type, with *nfs*  
     over-2-4  
 transparency, network, with *nfs* over-2-4  
 transparency, operating system, with *nfs*  
     over-2-4  
 transport independence, *rpc* rpc-1-2  
 transport protocol numbers, used with *rpc*  
     rpc-A-1

## U

*uid*, and protocol permission issues nfs-2-2  
*uid/gid* permissions, problems associated  
     nfs-2-2  
*u\_int xdr\_getpos* library primitives xdr-2-10  
 union handling primitives, *xdr* xdr-2-3  
 UNIX authentication, *rpc* rpc-3-4  
*Unset a Mapping* procedure, *rpc* rpc-A-2  
 unsigned integer definition, *xdr* standard  
     xdr-5-1  
 unsigned integers, data definition xdr-5-1  
 updating maps yp-2-1, yp-2-2, yp-2-3  
 user-defined type routines, example prog-2-5

## V

*valdat* data structure yp-2-5  
 version 2, *nfs* protocol nfs-2-1  
 version 2, *rpc* information nfs-2-2  
 version 2, server procedures, defined nfs-2-8  
 version 2, structure sizes nfs-2-3  
 version numbers yp-1-1  
 version numbers used to define *rpc* procedures  
     prog-2-2  
 versions, defined rpc-1-1  
*vfs* interface, flow-of-request diagram  
     over-2-3  
*vfs* interface, overview over-2-3  
 virtual file system, defined over-1-3  
 vnode, defined over-2-2  
 vnode, defined over-1-3

## W

*Write to Cache* server procedure nfs-2-10

## X

*x\_destroy* xdr-4-1  
*xdr* over-1-3, over-2-3, over-2-4  
*xdr*, and pointer semantics xdr-2-9  
*xdr*, built-in procedures prog-2-4  
*xdr* constructed data types, examples xdr-2-4  
*xdr*, creating memory streams xdr-3-1  
*xdr*, creating record streams xdr-3-2  
*xdr* data definition language rpc-3-1, yp-1-2,  
     yp-3-3  
*xdr* data definition language, example nfs-1-1  
*xdr* data representations yp-1-2  
*xdr*, example routine prog-3-3  
*xdr*, examples of use xdr-1-2, xdr-1-4  
*xdr*, justification for xdr-1-2  
*xdr* library xdr-1-4  
*xdr* library, fixed-sized arrays xdr-2-7  
*xdr* library primitives, array handling func-  
     tions xdr-2-4  
*xdr* library primitives, *bool\_t xdr\_array*  
     xdr-2-4  
*xdr* library primitives, *bool\_t xdr\_bool* xdr-2-2  
*xdr* library primitives, *bool\_t xdr\_bytes*  
     xdr-2-4  
*xdr* library primitives, *bool\_t xdr\_char* xdr-2-1  
*xdr* library primitives, *bool\_t xdr\_discrim*  
     xdr-2-7  
*xdr* library primitives, *bool\_t xdr\_double*  
     xdr-2-2  
*xdr* library primitives, *bool\_t xdr\_enum*  
     xdr-2-2  
*xdr* library primitives, *bool\_t xdr\_float*  
     xdr-2-2  
*xdr* library primitives, *bool\_t xdr\_hyper*  
     xdr-2-1  
*xdr* library primitives, *bool\_t xdr\_int* xdr-2-1  
*xdr* library primitives, *bool\_t xdr\_long* xdr-2-1  
*xdr* library primitives, *bool\_t xdr\_opaque*  
     xdr-2-6  
*xdr* library primitives, *bool\_t xdr\_reference*  
     xdr-2-9

- xdr* library primitives, *bool\_t xdr\_setpos*  
xdr-2-10
- xdr* library primitives, *bool\_t xdr\_short*  
xdr-2-1
- xdr* library primitives, *bool\_t xdr\_string*  
xdr-2-3
- xdr* library primitives, *bool\_t xdr\_u\_char*  
xdr-2-1
- xdr* library primitives, *bool\_t xdr\_u\_hyper*  
xdr-2-2
- xdr* library primitives, *bool\_t xdr\_u\_int*  
xdr-2-1
- xdr* library primitives, *bool\_t xdr\_u\_long*  
xdr-2-1
- xdr* library primitives, *bool\_t xdr\_u\_short*  
xdr-2-1
- xdr* library primitives, *bool\_t xdr\_void* xdr-2-3
- xdr* library primitives, byte array handling  
routines xdr-2-4
- xdr* library primitives, constructed data type  
filters xdr-2-3
- xdr* library primitives, discriminated unions  
xdr-2-7, xdr-2-9
- xdr* library primitives, discriminated unions,  
example xdr-2-8
- xdr* library primitives, enumeration filters  
xdr-2-2
- xdr* library primitives, floating-point filters  
xdr-2-2
- xdr* library primitives, no data routines  
xdr-2-3
- xdr* library primitives, non-filter primitives  
xdr-2-10
- xdr* library primitives, number filters xdr-2-1
- xdr* library primitives, opaque handles  
xdr-2-6
- xdr* library primitives, pointers xdr-2-9
- xdr* library primitives, pointers, example  
xdr-2-9
- xdr* library primitives, synopsis xdr-2-1
- xdr* library primitives, *u\_int xdr\_getpos*  
xdr-2-10
- xdr* library primitives, *xdr\_destroy* xdr-2-10
- xdr* library, stream creation routines xdr-1-4
- xdr* operation directions xdr-2-10
- xdr*, overview nfs-1-1
- xdr* routines, and direction independence  
xdr-1-4
- xdr* routines, compiling xdr-1-1
- xdr* routines, synopsis xdr-A-1
- xdr* routines, to interface streams to standard  
I/O xdr-3-1
- xdr* routines, *xdr\_array* xdr-A-1
- xdr* routines, *xdr\_bool* xdr-A-1
- xdr* routines, *xdr\_bytes* xdr-A-1
- xdr* routines, *xdr\_destroy* xdr-A-1, xdr-A-2
- xdr* routines, *xdr\_double* xdr-A-2
- xdr* routines, *xdr\_enum* xdr-A-2
- xdr* routines, *xdr\_float* xdr-A-2
- xdr* routines, *xdr\_free* xdr-A-2
- xdr* routines, *xdr\_getpos* xdr-A-3
- xdr* routines, *xdr\_hyper* xdr-A-3
- xdr* routines, *xdr\_inline* xdr-A-3
- xdr* routines, *xdr\_int* xdr-A-3
- xdr* routines, *xdr\_long* xdr-A-3
- xdr* routines, *xdrmem\_create* xdr-A-7
- xdr* routines, *xdr\_opaque* xdr-A-4
- xdr* routines, *xdr\_pointer* xdr-A-4
- xdr* routines, *xdrrec\_create* xdr-A-7
- xdr* routines, *xdrrec\_endofrecord* xdr-A-7
- xdr* routines, *xdrrec\_eof* xdr-A-8
- xdr* routines, *xdrrec\_skiprecord* xdr-A-8
- xdr* routines, *xdr\_reference* xdr-A-4
- xdr* routines, *xdr\_setpos* xdr-A-4
- xdr* routines, *xdr\_short* xdr-A-5
- xdr* routines, *xdrstdio\_create* xdr-A-8
- xdr* routines, *xdr\_string* xdr-A-5
- xdr* routines, *xdr\_u\_char* xdr-A-5
- xdr* routines, *xdr\_u\_hyper* xdr-A-5
- xdr* routines, *xdr\_u\_int* xdr-A-5
- xdr* routines, *xdr\_u\_long* xdr-A-6
- xdr* routines, *xdr\_union* xdr-A-6
- xdr* routines, *xdr\_u\_short* xdr-A-6
- xdr* routines, *xdr\_vector* xdr-A-6
- xdr* routines, *xdr\_void* xdr-A-6
- xdr* routines, *xdr\_wrapstring* xdr-A-7
- xdr* standard, assumptions used xdr-5-1
- xdr* standard, basic block size xdr-5-1
- xdr* standard, boolean definition xdr-5-2,  
yp-1-2
- xdr* standard, counted byte string definition  
xdr-5-3
- xdr* standard, defined xdr-5-1
- xdr* standard, double precision floating-point  
definition xdr-5-2
- xdr* standard, enumeration definition xdr-5-1
- xdr* standard, floating-point definition xdr-5-2
- xdr* standard, hyper integer definition xdr-5-2
- xdr* standard, hyper unsigned definition  
xdr-5-2
- xdr* standard, integer definition xdr-5-1
- xdr* standard, opaque data definition xdr-5-3
- xdr* standard, unsigned integer definition  
xdr-5-1
- xdr* stream access, introduction xdr-3-1
- xdr* structures used in mount protocol, sizes  
nfs-3-1
- xdr*, use in allocating memory, summary  
prog-3-4
- xdr*, use in deserializing memory prog-3-4
- xdr*, use in making data portable xdr-1-2
- xdr*, use in memory allocation prog-3-3
- xdr*, use in passing arbitrary data structures  
prog-2-4
- xdr*, use in serializing memory prog-3-4
- xdr*, use with *rpc* prog-2-4
- xdr*, *yp* interface yp-1-2
- xdr\_accepted\_reply*, *rpc* routine prog-A-13
- xdr\_array* routine, *rpc* prog-A-13
- xdr\_array* routine, *xdr* xdr-A-1
- xdr\_authunix\_parms*, *rpc* routine prog-A-14
- xdr\_bool* routine, *rpc* prog-A-14

*xdr\_bool* routine, *xdr* xdr-A-1  
*xdr\_bytes* routine, *rpc* prog-A-14  
*xdr\_bytes* routine, *xdr* xdr-A-1  
*xdr\_callhdr* routine, *rpc* prog-A-14  
*xdr\_callmsg* routine, *rpc* prog-A-14  
*xdr\_char* routine, *rpc* prog-A-15  
*xdr\_destroy* library primitives xdr-2-10  
*xdr\_destroy* routine, *xdr* xdr-A-1, xdr-A-2  
*xdr\_double* routine, *rpc* prog-A-15  
*xdr\_double* routine, *xdr* xdr-A-2  
*xdr\_enum* routine, *rpc* prog-A-15  
*xdr\_enum* routine, *xdr* xdr-A-2  
*xdr\_float* routine, *rpc* prog-A-15  
*xdr\_float* routine, *xdr* xdr-A-2  
*xdr\_free* routine, *xdr* xdr-A-2  
*xdr\_getpos* routine, *xdr* xdr-A-3  
*xdr\_hyper* routine, *rpc* prog-A-15  
*xdr\_hyper* routine, *xdr* xdr-A-3  
*xdr\_inline* routine, *rpc* prog-A-16  
*xdr\_inline* routine, *xdr* xdr-A-3  
*xdr\_int* routine, *rpc* prog-A-16  
*xdr\_int* routine, *xdr* xdr-A-3  
*xdr\_long* routine, *rpc* prog-A-16  
*xdr\_long* routine, *xdr* xdr-A-3  
*xdrmem\_create* routine xdr-3-1  
*xdrmem\_create* routine, *xdr* xdr-A-7  
*xdr\_opaque* routine, *rpc* prog-A-16  
*xdr\_opaque* routine, *xdr* xdr-A-4  
*xdr\_opaque\_auth*, *rpc* routine prog-A-16  
*xdr\_pmap* routine, *rpc* prog-A-17  
*xdr\_pmaplist* routine, *rpc* prog-A-17  
*xdr\_pointer* routine, *rpc* prog-A-17  
*xdr\_pointer* routine, *xdr* xdr-A-4  
*xdrrec\_create* routine xdr-3-2  
*xdrrec\_create* routine, *xdr* xdr-A-7  
*xdrrec\_endofrecord* routine xdr-3-3  
*xdrrec\_endofrecord* routine, *xdr* xdr-A-7  
*xdrrec\_eof* routine, *xdr* xdr-A-8  
*xdrrec\_skiprecord* routine xdr-3-3  
*xdrrec\_skiprecord* routine, *xdr* xdr-A-8  
*xdr\_reference* routine, *rpc* prog-A-17  
*xdr\_reference* routine, *xdr* xdr-A-4  
*xdr\_rejected\_reply*, *rpc* routine prog-A-17  
*xdr\_replymsg* routine, *rpc* prog-A-18  
*xdr\_setpos* routine, *xdr* xdr-A-4  
*xdr\_short* routine, *rpc* prog-A-18  
*xdr\_short* routine, *xdr* xdr-A-5  
*xdrstdio\_create* routine xdr-3-1  
*xdrstdio\_create* routine, *xdr* xdr-A-8  
*xdr\_string* routine, *rpc* prog-A-18  
*xdr\_string* routine, *xdr* xdr-A-5  
*xdr\_u\_char* routine, *xdr* xdr-A-5  
*xdr\_u\_char*, *rpc* routine prog-A-18  
*xdr\_u\_hyper* routine, *xdr* xdr-A-5  
*xdr\_u\_hyper*, *rpc* routine prog-A-18  
*xdr\_u\_int* routine, *xdr* xdr-A-5  
*xdr\_u\_int*, *rpc* routine prog-A-19  
*xdr\_u\_long* routine, *xdr* xdr-A-6  
*xdr\_u\_long*, *rpc* routine prog-A-19  
*xdr\_union* routine, *rpc* prog-A-19  
*xdr\_union* routine, *xdr* xdr-A-6

*xdr\_u\_short* routine, *xdr* xdr-A-6  
*xdr\_u\_short*, *rpc* routine prog-A-19  
*xdr\_vector* routine, *rpc* prog-A-19  
*xdr\_vector* routine, *xdr* xdr-A-6  
*xdr\_void* routine, *rpc* prog-A-20  
*xdr\_void* routine, *xdr* xdr-A-6  
*xdr\_wrapstring* routine, *rpc* prog-A-20  
*xdr\_wrapstring* routine, *xdr* xdr-A-7  
*x\_getpostn* macro xdr-4-1  
*x\_postn* macro xdr-4-1  
*xprt\_register* routine, *rpc* prog-A-20  
*xprt\_unregister* routine, *rpc* prog-A-20

## Y

*ymaplist* data structure yp-2-7  
*ymap\_parms* data structure yp-2-5  
*yp* over-3-1  
*yp* and */etc/passwd* over-2-1  
*yp* and network administration over-2-1  
*yp* binder data structures yp-3-2  
*yp* binder data structures, *domainname* yp-3-2  
*yp* binder data structures, *ypbind\_binding* yp-3-2  
*yp* binder data structures, *ypbind\_resp* yp-3-3  
*yp* binder data structures, *ypbind\_setdom* yp-3-3  
*yp* binder protocol definition yp-3-1  
*yp* binder remote procedures yp-3-3  
*yp* binder remote procedures, *Do Nothing* yp-3-3  
*yp* binder remote procedures, *Get Current Binding for a Domain* yp-3-4  
*yp* binder remote procedures, *Set Domain Binding* yp-3-4  
*yp* binders, design assumptions yp-3-1  
*yp* binders, overview yp-3-1  
*yp*, characteristics of over-3-1  
*yp* clients, operation over-4-2  
*yp* data storage over-4-2  
*yp*, database server protocol definition yp-2-3  
*yp* database server remote procedures yp-2-7  
*yp* domains, defined over-3-1  
*yp*, features not included yp-2-2  
*yp*, features not included, map update within *yp* yp-2-3  
*yp*, features not supported, access control yp-2-3  
*yp*, features not supported, guaranteed global consistency yp-2-3  
*yp*, features not supported, version commitment across multiple requests yp-2-3  
*yp* hosts file over-4-3  
*yp* map operation yp-2-1  
*yp* maps, defined over-3-1  
*yp* maps, derivation from */etc/passwd* files over-3-1  
*yp* master servers over-3-2  
*yp*, operation over-4-1  
*yp*, overview over-4-1, yp-1-1  
*yp*, overview of over-1-3, over-2-1, over-3-1

- yp passwd* file over-4-3
- yp* procedures, *Answer Only If You Serve This Domain* yp-2-8
- yp* procedures, *Do Nothing* yp-2-7
- yp* procedures, *Do You Serve This Domain?* yp-2-8
- yp* procedures, *Get All Key-Value Pairs in Map* yp-2-11
- yp* procedures, *Get All Maps in Domain* yp-2-11
- yp* procedures, *Get First Key-Value Pair in Map* yp-2-9
- yp* procedures, *Get Map Master Name* yp-2-11
- yp* procedures, *Get Map Order Number* yp-2-11
- yp* procedures, *Get Next Key-Value Pair in Map* yp-2-9
- yp* procedures, *Reinitialize Internal State* yp-2-10
- yp* procedures, *Return Value of a Key* yp-2-8
- yp* procedures, *Transfer Map* yp-2-10
- yp* remote procedures, data structures used yp-2-4
- yp* remote procedures, status values returned yp-2-4
- yp*, *rpc* interface yp-1-2
- yp* servers and clients, example over-3-2
- yp* servers, setting up over-4-2
- yp* slave servers over-3-2
- yp*, *xdr* interface yp-1-2
- ypbind*(8), operation over-4-2
- ypbind\_binding* data structure, used with *yp* binders yp-3-2
- ypbinderr* yp-3-2
- ypbind\_resp* data structure, used with *yp* binders yp-3-3
- ypbind\_resptype* yp-3-2
- ypbind\_setdom* data structure, used with *yp* binders yp-3-3
- ypcat* over-4-3
- ypcat*(1), defined over-4-2
- ypcat*(1), example over-4-2
- ypinit* over-3-2
- ypinit*(8) over-4-2
- ypmake*(8) over-4-3
- ypmatch*(1), defined over-4-2
- ypmatch*(1), example over-4-2
- yppasswd*(1) over-4-3
- yppasswd* over-4-3
- YPPROC\_ALL* procedure, *yp* yp-2-1
- YPPROC\_FIRST* procedure, *yp* yp-2-1
- YPPROC\_MATCH* procedure, *yp* yp-2-1
- YPPROC\_NEXT* procedure, *yp* yp-2-1
- YPPROC\_XFR* procedure, *yp* yp-2-2
- yppush*(8) over-4-2
- ypreq\_xfr* data structure yp-2-5
- ypresp\_all* data structure yp-2-6
- ypresp\_key\_val* data structure yp-2-6
- ypresp\_maplist* data structure yp-2-7
- ypresp\_master* data structure yp-2-6
- ypresp\_order* data structure yp-2-6
- ypresp\_val* data structure yp-2-6
- ypresp\_xfr* data structure yp-2-7
- ypserv* over-4-2
- ypstat*, status values returned yp-2-4
- ypwhich*(1) over-4-2, over-4-3
- ypxfr* over-4-2
- ypxfrstat*, status values returned yp-2-4

