RPCGEN

## NAME

rpcgen − an RPC protocol compiler

## SYNOPSIS

**rpcgen** *infile*
**rpcgen −c | −h | −l | −m** [ **−o** *outfile* ] [ *infile* ]
**rpcgen −s** *transport* [ **−o** *outfile* ] [ *infile* ]

## DESCRIPTION

**rpcgen** is a tool that generates C code to implement an RPC protocol. The input to **rpcgen** is a language similar to C known as RPC Language (Remote Procedure Call Language). Information about the syntax of RPC Language is available in the '*rpcgen*' *Programming Guide.*

**rpcgen** is normally used as in the first synopsis where it takes an input file and generates four output files. If the *infile* is named **proto.x**, then **rpcgen** will generate a header file in **proto.h**, XDR routines in **proto_xdr.c**, server-side stubs in **proto_svc.c**, and client-side stubs in **proto_clnt.c**.

The other synopses shown above are used when one does not want to generate all the output files, but only a particular one. Their usage is described in the USAGE section below.

The C-preprocessor, **cpp**(1), is run on all input files before they are actually interpreted by **rpcgen**, so all the **cpp** directives are legal within an **rpcgen** input file. For each type of output file, **rpcgen** defines a special **cpp** symbol for use by the **rpcgen** programmer:

| | |
|---|---|
| RPC_HDR | defined when compiling into header files |
| RPC_XDR | defined when compiling into XDR routines |
| RPC_SVC | defined when compiling into server-side stubs |
| RPC_CLNT | defined when compiling into client-side stubs |

In addition, **rpcgen** does a little preprocessing of its own. Any line beginning with '**%**' is passed directly into the output file, uninterpreted by **rpcgen**.

You can customize some of your XDR routines by leaving those data types undefined. For every data type that is undefined, **rpcgen** will assume that there exists a routine with the name **xdr_** prepended to the name of the undefined type.

## OPTIONS

**−c** Compile into XDR routines.

**−h** Compile into **C** data-definitions (a header file)

**−l** Compile into client-side stubs.

**−m** Compile into server-side stubs, but do not generate a "main" routine. This option is useful for doing callback-routines and for people who need to write their own "main" routine to do initialization.

**−o** *outfile*
Specify the name of the output file. If none is specified, standard output is used (**−c**, **−h**, **−l** and **−s** modes only).

**−s** *transport*
Compile into server-side stubs, using the the given transport. The supported transports are **udp** and **tcp**. This option may be invoked more than once so as to compile a server that serves multiple transports.

## SEE ALSO

**cpp**(1)

'*rpcgen*' *Programming Guide.*

## BUGS

Nesting is not supported. As a work-around, structures can be declared at top-level, and their name used inside other structures in order to achieve the same effect.

Name clashes can occur when using program definitions, since the apparent scoping does not really apply. Most of these can be avoided by giving unique names for programs, versions, procedures and types.