NAME

rpc_svc_create, svc_control, svc_create, svc_destroy, svc_dg_create, svc_fd_create, svc_raw_create, svc_tli_create, svc_tp_create, svc_vc_create – library routines for the creation of server handles

MT-LEVEL

MT-Safe

DESCRIPTION

These routines are part of the RPC library which allows C language programs to make procedure calls on servers across the network. These routines deal with the creation of service handles. Once the handle is created, the server can be invoked by calling **svc_run( )**.

Routines

See **rpc**(3N) for the definition of the **SVCXPRT** data structure.

#include <rpc/rpc.h>

bool_t
svc_control (SVCXPRT **svc*, **const u_int** *req*, **void** **info***);**

A function to change or retrieve various information about a service object. *req* indicates the type of operation and *info* is a pointer to the information. The supported values of *req*, their argument types, and what they do are:

SVCGET_VERSQUIET If a request is received for a program number served by this server but the version number is outside the range registered with the server, an RPC_PROGVERSMISMATCH error will normally be returned. *info* should be a pointer to an integer. Upon successful completion of the SVCGET_VERSQUIET request, **info* contains an integer which describes the server's current behavior: 0 indicates normal server behavior (that is, an RPC_PROGVERSMISMATCH error will be returned); 1 indicates that the out of range request will be silently ignored.

SVCSET_VERSQUIET If a request is received for a program number served by this server but the version number is outside the range registered with the server, an RPC_PROGVERSMISMATCH error will normally be returned. It is sometimes desirable to change this behavior. *info* should be a pointer to an integer which is either 0 (indicating normal server behavior – an RPC_PROGVERSMISMATCH error will be returned), or 1 (indicating that the out of range request should be silently ignored).

int svc_create(const void (**dispatch***)(const struct svc_req *, const SVCXPRT *),**
const u_long *prognum*, **const u_long** *versnum*, **const char** **nettype***);**

svc_create( ) creates server handles for all the transports belonging to the class *nettype*.

nettype defines a class of transports which can be used for a particular application. The transports are tried in left to right order in **NETPATH** variable or in top to bottom order in the netconfig database. If *nettype* is **NULL**, it defaults to **netpath**.

svc_create( ) registers itself with the **rpcbind** service (see **rpcbind**(1M)). *dispatch* is called when there is a remote procedure call for the given *prognum* and *versnum*; this requires calling **svc_run( )** (see **svc_run( )** in **rpc_svc_reg**(3N)). If **svc_create( )** succeeds, it returns the number of server handles it created, otherwise it returns **0** and an error message is logged.

void svc_destroy(SVCXPRT **xprt***);**

A function macro that destroys the RPC service handle *xprt*. Destruction usually involves deallocation of private data structures, including *xprt* itself. Use of *xprt* is undefined after calling this routine.

SVCXPRT *svc_dg_create(const int *fildes***, const u_int** *sendsz***, const u_int** *recvsz***);**

This routine creates a connectionless RPC service handle, and returns a pointer to it. This routine returns **NULL** if it fails, and an error message is logged. *sendsz* and *recvsz* are parameters used to specify the size of the buffers. If they are **0**, suitable defaults are chosen. The file descriptor *fildes* should be open and bound. The server is not registered with **rpcbind**(1M).

Warning: since connectionless-based RPC messages can only hold limited amount of encoded data, this transport cannot be used for procedures that take large arguments or return huge results.

SVCXPRT *svc_fd_create(const int *fildes***, const u_int** *sendsz***, const u_int** *recvsz***);**

This routine creates a service on top of an open and bound file descriptor, and returns the handle to it. Typically, this descriptor is a connected file descriptor for a connection-oriented transport. *sendsz* and *recvsz* indicate sizes for the send and receive buffers. If they are **0**, reasonable defaults are chosen. This routine returns **NULL** if it fails, and an error message is logged.

SVCXPRT *svc_raw_create(void);

This routine creates an RPC service handle and returns a pointer to it. The transport is really a buffer within the process's address space, so the corresponding RPC client should live in the same address space; (see **clnt_raw_create( )** in **rpc_clnt_create**(3N)). This routine allows simulation of RPC and acquisition of RPC overheads (such as round trip times), without any kernel and networking interference. This routine returns **NULL** if it fails, and an error message is logged.

Note: **svc_run( )** should not be called when the raw interface is being used.

SVCXPRT *svc_tli_create(const int *fildes***, const struct netconfig** **netconf***,**
const struct t_bind **bindaddr***, const u_int** *sendsz***, const u_int** *recvsz***);**

This routine creates an RPC server handle, and returns a pointer to it. *fildes* is the file descriptor on which the service is listening. If *fildes* is **RPC_ANYFD**, it opens a file descriptor on the transport specified by *netconf*. If the file descriptor is unbound and *bindaddr* is non-null *fildes* is bound to the address specified by *bindaddr*, otherwise *fildes* is bound to a default address chosen by the transport. In the case where the default address is chosen, the number of outstanding connect requests is set to 8 for connection-oriented transports. The user may specify the size of the send and receive buffers with the parameters *sendsz* and *recvsz* ; values of **0** choose suitable defaults. This routine returns **NULL** if it fails, and an error message is logged. The server is not registered with the **rpcbind**(1M) service.

SVCXPRT *svc_tp_create(const void (**dispatch***)(const struct svc_req *,**
const SVCXPRT *), const u_long *prognum***, const u_long** *versnum***,**
const struct netconfig **netconf***);**

svc_tp_create( ) creates a server handle for the network specified by *netconf*, and registers itself with the **rpcbind** service. *dispatch* is called when there is a remote procedure call for the given *prognum* and *versnum*; this requires calling **svc_run( )**. **svc_tp_create( )** returns the service handle if it succeeds, otherwise a **NULL** is returned and an error message is logged.

SVCXPRT *svc_vc_create(const int *fildes***, const u_int** *sendsz***, const u_int** *recvsz***);**

This routine creates a connection-oriented RPC service and returns a pointer to it. This routine returns **NULL** if it fails, and an error message is logged. The users may specify the size of the send and receive buffers with the parameters *sendsz* and *recvsz*; values of **0** choose suitable defaults. The file descriptor *fildes* should be open and bound. The server is not registered with the **rpcbind**(1M) service.

SEE ALSO

rpcbind(1M), **rpc**(3N), **rpc_svc_calls**(3N), **rpc_svc_err**(3N), **rpc_svc_reg**(3N)