NAME

rpcb_getmaps, **rpcb_getaddr**, **rpcb_gettime**, **rpcb_rmtcall**, **rpcb_set**, **rpcb_unset** — library routines for RPC bind service

SYNOPSIS

```
#include <rpc/rpc.h>

rpcblist *
rpcb_getmaps(const struct netconfig *netconf, const char *host);

bool_t
rpcb_getaddr(const rpcprog_t prognum, const rpcvers_t versnum,
    const struct netconfig *netconf, struct netbuf *svcaddr,
    const char *host);

bool_t
rpcb_gettime(const char *host, time_t * timep);

enum clnt_stat
rpcb_rmtcall(const struct netconfig *netconf, const char *host,
    const rpcprog_t prognum, const rpcvers_t versnum,
    const rpcproc_t procnum, const xdrproc_t inproc, const caddr_t in,
    const xdrproc_t outproc, const caddr_t out,
    const struct timeval tout, const struct netbuf *svcaddr);

bool_t
rpcb_set(const rpcprog_t prognum, const rpcvers_t versnum,
    const struct netconfig *netconf, const struct netbuf *svcaddr);

bool_t
rpcb_unset(const rpcprog_t prognum, const rpcvers_t versnum,
    const struct netconfig *netconf);
```

DESCRIPTION

These routines allow client C programs to make procedure calls to the RPC binder service. (see `rpcbind`(8)) maintains a list of mappings between programs and their universal addresses.

Routines

rpcb_getmaps()

An interface to the rpcbind service, which returns a list of the current RPC program-to-address mappings on *host*. It uses the transport specified through *netconf* to contact the remote rpcbind service on *host*. This routine will return `NULL`, if the remote rpcbind could not be contacted.

rpcb_getaddr()

An interface to the rpcbind service, which finds the address of the service on *host* that is registered with program number *prognum*, version *versnum*, and speaks the transport protocol associated with *netconf*. The address found is returned in *svcaddr*. The *svcaddr* argument should be preallocated. This routine returns `TRUE` if it succeeds. A return value of `FALSE` means that the mapping does not exist or that the RPC system failed to contact the remote rpcbind service. In the latter case, the global variable *rpc_createerr* (see `rpc_clnt_create`(3)) contains the RPC status.

rpcb_gettime()

This routine returns the time on *host* in *timep*. If *host* is `NULL`, **rpcb_gettime**() returns the time on its own machine. This routine returns `TRUE` if it succeeds, `FALSE` if it fails. The **rpcb_gettime**() function can be used to synchronize the time between the client and the remote server.

rpcb_rmtcall()
An interface to the rpcbind service, which instructs rpcbind on *host* to make an RPC call on your behalf to a procedure on that host. The **netconfig**() structure should correspond to a connectionless transport. The *svcaddr* argument will be modified to the server's address if the procedure succeeds (see **rpc_call**() and **clnt_call**() in rpc_clnt_calls(3) for the definitions of other arguments).

This procedure should normally be used for a "ping" and nothing else. This routine allows programs to do lookup and call, all in one step.

Note: Even if the server is not running **rpcb_rmtcall**() does not return any error messages to the caller. In such a case, the caller times out.

Note: **rpcb_rmtcall**() is only available for connectionless transports.

rpcb_set()
An interface to the rpcbind service, which establishes a mapping between the triple [*prognum*, *versnum*, *netconf->nc_netid*] and *svcaddr* on the machine's rpcbind service. The value of *nc_netid* must correspond to a network identifier that is defined by the netconfig database. This routine returns TRUE if it succeeds, FALSE otherwise. (See also **svc_reg**() in rpc_svc_calls(3).) If there already exists such an entry with rpcbind, **rpcb_set**() will fail.

rpcb_unset()
An interface to the rpcbind service, which destroys the mapping between the triple [*prognum*, *versnum*, *netconf->nc_netid*] and the address on the machine's rpcbind service. If *netconf* is NULL, **rpcb_unset**() destroys all mapping between the triple [*prognum*, *versnum*, all-transports] and the addresses on the machine's rpcbind service. This routine returns TRUE if it succeeds, FALSE otherwise. Only the owner of the service or the super-user can destroy the mapping. (See also **svc_unreg**() in rpc_svc_calls(3).)

AVAILABILITY
These functions are part of libtirpc.

SEE ALSO
rpc_clnt_calls(3), rpc_svc_calls(3), rpcbind(8), rpcinfo(8)