RPC/XDR Errors

This appendix provides reference information on Remote Procedure Call (RPC) error handling. It includes these sections:

- RPC Library Error Messages

  Lists error messages generated by the RPC library.

- Message Formatting

  Describes the error message formatting routines clnt_spcreateerror(), clnt_sperrno(), and clnt_sperror().RPC Log Interface

When an error is detected by the RPC library, it calls an externally defined function called rpclog(). The default rpclog shipped with the RPC library simply formats the information passed it and then print it to stderr.

RPC Library Error Messages

This section lists error messages generated by the RPC library. Messages are listed in alphabetical order. For each message, the error number and csect string (csectp) are also given.

Note Error Number 34 is not used.

AUTHUNIX_CREATE OUT OF MEMORY

Explanation Error Number 1 – Could not malloc().

csectp – AUTHUNIX_CREATE

CACHE_SET COULD NOT ALLOCATE NEW RPC_BUFFER

Explanation Error Number 33 – The malloc() function for RPC buffer failed.

csectp – SVCUDP_BUFCREATE

CACHE_SET VICTIM NOT FOUND

Explanation Error Number 31 – Corrupted cache caused search to be aborted.

csectp – SVCUDP_BUFCREATE

CACHE_SET VICTIM ALLOC FAILED

Explanation Error Number 32 – The malloc() function for cache element failed.

csectp – SVCUDP_BUFCREATE

CLNT_BROADCAST BROADCAST DESERIALIZATION PROBLEM

Explanation Error Number 9 – Received a bad RPC reply.

csectp – CLNT_BROADCAST

CLNT_BROADCAST BROADCAST RPC NOT SUPPORTED

Explanation Error Number 35 – Broadcast RPC is not supported.

csectp – CLNT_BROADCAST

CLNT_BROADCAST BROADCAST SELECT PROBLEM

Explanation Error Number 13 – The select() function failed when broadcasting.

csectp – CLNT_BROADCAST

CLNT_BROADCAST CANNOT CREATE SOCKET FOR BROADCAST RPC

Explanation Error Number 10 – The socket() routine failed.

csectp – CLNT_BROADCAST

CLNT_BROADCAST CANNOT RECEIVE REPLY TO BROADCAST

Explanation Error Number 14 – Client did not receive a reply to a broadcast request.

csectp – CLNT_BROADCAST

CLNT_BROADCAST CANNOT SEND BROADCAST PACKET

Explanation Error Number 12 – The sendto() routine using broadcast address failed.

csectp – CLNT_BROADCAST

CLNT_BROADCAST CANNOT SET SOCKET OPTION SO_BROADCAST

Explanation Error Number 11 – The setsockopt() routine for SO_BROADCAST failed.

csectp – CLNT_BROADCAST

CLNT_PCREATEERROR string

Explanation Error Number 3 – The string is replaced with a message generated by clnt_spcreateerror(). See “Message Formatting” on page -8 for the formatting of string.

csectp – CLNT_PCREATEERROR

CLNT_PERRNO string

Explanation Error Number 5 – The string is replaced with a message generated by clnt_sperrno(). See “Message Formatting” on page -9 for the formatting of string.

csectp – CLNT_PERRNO

CLNT_PERROR string

Explanation Error Number 4 – The string is replaced with a message generated by clnt_sperror(). See “Message Formatting” on page -10 for the formatting of string.

csectp – CLNT_PERROR

CLNTRAW_CREATE FATAL HEADER SERIALIZATION ERROR

Explanation Error Number 6 – Could not XDR RPC call header.

csectp – CLNTRAW_CREATE

CLNTTCP_CREATE OUT OF MEMORY

Explanation Error Number 1 – Could not malloc().

csectp – CLNTTCP_CREATE

CLNTUDP_CREATE OUT OF MEMORY

Explanation Error Number 1 – Could not malloc().

csectp – CLNTUDP_CREATE

GET_MYADDRESS GETHOSTNAME

Explanation Error Number 7 – The gethostname() function failed.

csectp – GET_MYADDRESS

GET_MYADDRESS GETHOSBYTNAME

Explanation Error Number 8 – The gethostbyname() function failed.

csectp – GET_MYADDRESS

MAKEFD_XPRT OUT OF MEMORY

Explanation Error Number 1 – Could not malloc().

csectp – MAKEFD_XPRT

MARSHAL_NEW_AUTH FATAL MARSHALLING PROBLEM

Explanation Error Number 2 – Could not XDR authentication structures.

csectp – AUTHUNIX_CREATE

MVS_SVC_RUN SELECT FAILED

Explanation Error Number 22 – The select() function failed.

csectp – MVS_SVC_RUN

REGISTERRPC CAN'T REASSIGN PROCEDURE NUMBER 0

Explanation Error Number 16 – Procedure number is 0 on call to registerrpc().

csectp – REGISTERRPC

REGISTERRPC COULDN'T CREATE AN RPC SERVER

Explanation Error Number 17 – Couldn’t create an RPC server.

csectp – REGISTERRPC

REGISTERRPC COULDN'T REGISTER PROG *d_value1* VERS *d_value2*

Explanation Error Number 18 – Could not register RPC server with portmapper. d_value1 is the RPC program number and d_value2 is the RPC program version. number.

csectp – REGISTERRPC

SVC_RUN SELECT FAILED

Explanation Error Number 22 – The select() function failed.

csectp – SVC_RUN

SVCAUTH_UNIX BAD AUTH_LEN GID *d_value1* STR *d_value2* AUTH *d_value3*

Explanation Error Number 15 – UNIX credentials are invalid. d_value1 is the UNIX group id, d_value2 is the authentication string length and d_value3 is the authentication length.

csectp – SVCAUTH_UNIX

SVCTCP_CREATE CANNOT GETSOCKNAME OR LISTEN

Explanation Error Number 24 – The getsockname() or listen() function failed.

csectp – SVCTCP_CREATE

SVCTCP_CREATE OUT OF MEMORY

Explanation Error Number 1 – Could not malloc().

csectp – SVCTCP_CREATE

SVCTCP_CREATE TCP SOCKET CREATION PROBLEM

Explanation Error Number 23 – Could not create TCP socket.

csectp – SVCTCP_CREATE

SVCUDP_BUFCREATE CANNOT GETSOCKNAME

Explanation Error Number 26 – The getsockname() function failed.

csectp – SVCUDP_BUFCREATE

SVCUDP_BUFCREATE OUT OF MEMORY

Explanation Error Number 1 – Could not malloc().

csectp – SVCUDP_BUFCREATE

SVCUDP_BUFCREATE UDP SOCKET CREATION PROBLEM

Explanation Error Number 25 – Could not create UDP socket.

csectp – SVCUDP_BUFCREATE

SVCUDP_CREATE OUT OF MEMORY

Explanation Error Number 1 – Could not malloc().

csectp – SVCUDP_CREATE

SVCUDP_ENABLECACHE CACHE ALREADY ENABLED

Explanation Error Number 27 – Cache enable request when cache already enabled

csectp – SVCUDP_BUFCREATE

SVCUDP_ENABLECACHE COULD NOT ALLOCATE CACHE

Explanation Error Number 28 – The malloc() function for cache control space failed.

csectp – SVCUDP_BUFCREATE

SVCUDP_ENABLECACHE COULD NOT ALLOCATE CACHE DATA

Explanation Error Number 29 – The malloc() function for cache data space failed.

csectp – SVCUDP_BUFCREATE

SVCUDP_ENABLECACHE COULD NOT ALLOCATE CACHE FIFO

Explanation Error Number 30 – The malloc() function for cache FIFO failed.

csectp – SVCUDP_BUFCREATE

UNIVERSAL COULD NOT SEND REPLY

Explanation Error Number 19 – Could not send reply.

csectp – REGISTERRPC

UNIVERSAL NEVER REGISTERED PROG *d_value*

Explanation Error Number 21 – Program *d_value* was never registered. *d_value* is the RPC program number.

csectp – REGISTERRPC

UNIVERSAL TROUBLE REPLYING TO PROG *d_value*

Explanation Error Number 20 – Could not send reply. *d_value* is the RPC program number.

csectp – REGISTERRPC

XDR_ARRAY OUT OF MEMORY

Explanation Error Number 1 – Could not malloc().

csectp – XDR_ARRAY

XDR_BYTES OUT OF MEMORY

Explanation Error Number 1 – Could not malloc().

csectp – XDR_BYTES

XDR_RECORD OUT OF MEMORY

Explanation Error Number 1 – Could not malloc().

csectp – XDR_RECORD

XDR_REFERENCE OUT OF MEMORY

Explanation Error Number 1 – Could not malloc().

csectp – XDR_REFERENCE

XDR_STRING OUT OF MEMORY

Explanation Error Number 1 – Could not malloc().

csectp – XDR_STRING

Message Formatting

These error message formatting routines are described in this section:

- clnt_spcreateerror()
- clnt_sperrno()
- clnt_sperror()

clnt_spcreateerror()

Explanation This routine formats a message about errors related to the creation of a client handle. It should be passed a pointer to user message text. It formats the message this way and then returns a pointer to the formatted message:

***%s*: *%ns*\n**

%s User-supplied message text.

%ns Message text generated by clnt_sperrno() acting on the client status.

If the creation error is neither RPC_PMAPFAILURE nor RPC_SYSTEMERROR, a newline is appended. Otherwise these errors are additionally added:

- For RPC_PMAPFAILURE:

  ***%s*: *%ns* - *%es*\n**

%s User supplied message text.

%ns Message text generated by clnt_sperrno() acting on the client status.

%es Message text generated by lnt_sperrno() acting on the RPC error status.

- For RPC_SYSTEMERROR:

  ***%s*: *%ns* - *%ss*\n**

%s User supplied message text

%ns Message text generated by clnt_sperrno() acting on the client status.

%ss Message text generated by indexing into the perror() socket library error table using the RPC err. If the error is not in the socket library perror() error table %ss is replaced with ERROR %d where %d is the RPC library errno value.

clnt_sperrno()

Explanation This routine generates a message about an RPC library error. This routine should be passed an enum clnt_stat argument. A pointer to the message text relating to the clnt_stat argument is returned. It can be one of the following:

RPC: SUCCESS	Successful completion
RPC: CAN'T ENCODE ARGUMENTS	Client could not XDR the arguments it is to pass to the remote procedure.
RPC: CAN'T DECODE RESULT	Client could not XDR the result returned from the remote procedure.
RPC: UNABLE TO SEND	Client could not send an RPC CALL to the remote procedure.
RPC: UNABLE TO RECEIVE	Client could not receive the RPC REPLY from the remote procedure.
RPC: TIMED OUT	Client did not get a response from the server within an allowable amount of time.
RPC: INCOMPATIBLE VERSIONS OF RPC	The versions of the RPC protocol used by the server and the client are not equal.
RPC: AUTHENTICATION ERROR	Authentication check failed on remote system.
RPC: PROGRAM UNAVAILABLE	Program is not available on remote system.
RPC: PROGRAM/VERSION MISMATCH	Program and version are not available on the remote system.
RPC: PROCEDURE UNAVAILABLE	Requested procedure of the selected program on the remote system is not available for use.
RPC: SERVER CAN'T DECODE ARGUMENTS	Remote system could not understand arguments passed to the selected program.
RPC: REMOTE SYSTEM ERROR	Remote system had a major failure trying to execute the selected program.
RPC: UNKNOWN HOST	User selected a remote host system that is unknown to the DNR.
RPC: UNKNOWN PROTOCOL	User specified an unsupported protocol to be used for transport.
RPC: PORT MAPPER FAILURE	Remote host's portmapper could not be communicated with properly.
RPC: PROGRAM NOT REGISTERED	Remote program is not registered with the remote host's portmapper.
RPC: FAILED (UNSPECIFIED ERROR)	Error was not specific enough to justify its own error code.
RPC: (UNKNOWN ERROR CODE)	Error is not decipherable.

clnt_sperror()

Explanation This error message formatting routine requires two arguments:

- A pointer to a client handle
- A pointer to user message text

It formats a message and returns a pointer to the formatted message text.

The user message is formatted followed by a colon and a space (e.g., "%s: "). A message generated by clnt_sperrno() (see "Message Formatting" on page -9) follows and then the message formatting varies based on the current error status.

If the error is any of these, a newline is attached to the user portion and the pointer to the text returned:

- RPC_SUCCES
- RPC_CANTENCODEARGS
- RPC_CANTDECODERES
- RPC_TIMEDOUT
- RPC_PROGUNAVAIL
- RPC_PROCUNAVAIL
- RPC_CANTDECODEARGS
- RPC_SYSTEMERROR
- RPC_UNKNOWNHOST
- RPC_UNKNOWNPROTO
- RPC_PMAPFAILURE
- RPC_PROGNOTREGISTERED
- RPC_FAILED

Thus the message for these looks like this:

***s*: *%ns*\n**

An example is:

```
clnt_sperror(clntp, "USER MESSAGE");
```

The clnt_sperrno routine returns

```
RPC: CAN'T DECODE RESULT t
```

and the message pointer points to

```
USER MESSAGE: RPC: CAN'T DECODE RESULT\n
```

If the error is RPC_CANTSEND or RPC_CANTRECV, the user portion of the message is followed by an entry from the perror() error message list of the socket library.

The generalized format of the message looks like this:

***%s*: *%ns* ; errno = *%ss*\n**

%s User message passed.

%ns clnt_sperrno() generated message text.

%ss Message from the socket library.

If the error status is RPC_VERSMISMATCH, the message is formatted like this:

***%s*: *%ns* ; LOW VERSION = *%lul*, HIGH VERSION = *%luh*\n**

%s User message passed.

%ns clnt_sperrno() generated message text.

%lul Lowest decimal version number of the RPC program running.

%luh Highest decimal version number of the RPC program running.

If the error status is RPC_AUTHERROR, the message is formatted like this:

***%s*: *%ns* ; why = *%sa*:**

%s User message provided.

%ns clnt_sperrno() generated message text.

%sa Authentication message generated.

These are the authentication messages:

Table 12-1 Authentication Messages

Message	Description
AUTHENTICATION OK	The authentication is OK.
INVALID CLIENT CREDENTIAL	Client authentication credentials are incorrect for authentication type.
SERVER REJECTED CREDENTIAL	Client credentials do not allow access to the procedure.
INVALID CLIENT VERIFIER	Credentials are not supported by client.
SERVER REJECTED VERIFIER	Server could not decode this type of authentication.
CLIENT CREDENTIAL TOO WEAK	Client credentials formatted properly but are of too low authentication to allow access.
INVALID SERVER VERIFIER	Credentials are not supported by server.
FAILED (UNSPECIFIED ERROR)	Authentication failed for an error that does not justify a more specific error code.
UNKNOWN AUTHENTICATION ERROR - %d	The authentication error is unknown to the RPC library. The %d is replaced with the decimal value of the authentication error.

If the error status is RPC_PROGVERSMISMATCH, the message is formatted this way:

***%s*: *%ns* ; LOW VERSION = *%lul*, HIGH VERSION = *%luh*\n**

%s User message provided.

%ns clnt_sperrno() generated message text.

%lul Lowest version of the RPC program running.

%luh Highest version of the RPC program running.

If the error status is not mentioned, the message is formatted this way:

***%s*: *%ns* ; S1 = *%lu1*, S2 = *%lu2*\n**

%s User message provided.

%ns clnt_sperrno() generated message text.

%lu1 First error argument.

%lu2 Second error argument.