Documentation of the SCTP-Implementation
Release: sctplib-1.0

Andreas Jungmaier
ajung@exp-math.uni-essen.de
Michael Tüxen
Michael.Tuexen@icn.siemens.de

August 21, 2001

1 Nomenclature

Throughout this document,

callbacks for SCTP events are passed in a **SCTP_ulpCallbacks** (described more closely in sections 6.2.1 and 6.5).

Then the application either calls the possibly blocking function **sctp_eventLoop**() or the nonblocking function **sctp_getEvents**(). While calling the former, it will react to a previously scheduled timer or any file descriptor event (by executing the registered callback functions). In case a timer is scheduled at a very late point in time, and no events h-26en on registered file descriptors (e.g. sockets), the program will sleep (because the system call **poll**() is used. In this case, the control flow is handled by the library, and the user **must** register -26ropriate callbacks for events and timers before handing control over to the **sctp_eventLoop**() function. The 6roper use of the **sctp_eventLoop**() is explained in some simple example programs in section 7.

On the other hand, **sctp_getEvents**() which may be used to check for timer or file descriptor events. If -229(the60(a9(the60(none,)-263(it)-260(returns)-259(immediately)65(.)-340(While)-260(scheduling)-260(

6.2.2SCTP _InstanceParameters

This strucd contains variables that may be set per SCTP instance (as defaults), and can subsequently set or retrieved after an instance has been registered. All associations that will be created from this instance will then inherit these parameters.

Definition:

```
strucd SCTP_Instance_Parameters {
```

numberOfAddresses returns the number of destination addresses this association has, i.e. the number of possible paths.

primaryDestinationAddress returns a string containing the primary destination addrhs.

mented, although the API already contains the function **sctp_setPathStatus**(), that may be used for this purpose in a later release.

This struct also contains values from the flow control module, and may thus be used to check the status of the congestion control mechanisms.

Definition:

```
struct SCTP_Path_Status
{
  unsigned char destinationAddress[SCTP_MAX_IP_LEN];
  /**  SCTP_PATH_ACTIVE  0, SCTP_PATH_INACTIVE   1    */
  short state;
  /** smoothed round trip time in msecs */
  unsigned int srtt;
  /** current rto value in msecs */
  unsigned int rto;
  /** round trip time variation, in msecs */
  unsigned int rttvar;
  /** defines the rate at which heartbeats are sent */
  unsigned int heartbeatIntervall;
  /**  congestion window size */
  unsigned int cwnd;
  /**  congestion window size 2  */
  unsigned int cwnd2;
  /**  Partial Bytes Acked  */
  unsigned int partialBytesAcked;
  /**  Slow Start Threshold  */
  unsigned int ssthresh;
  unsigned int outstandingBytesPerAddress;
  /**  Current MTU (flowcontrol) */
  unsigned int mtu;
  /** per path ? per instance ? for the IP type of service field. */
  uns[...]us)TJ 0 -; 9.963 Tf 5.978 0  Td26.526(che26.2.5ur
```

7etPaotheatuetiC0bathe9.963 Tf 58.011 0 53.02Td28.488nition:

void*

6.3.4 sctp_registerStdinCallback()

6.3.8 sctp_getTime()

This helper function returns a 32 bit value representing the current time in milliseconds. Beware, this counter wraps about once per 20 days. Keep that in mind when calculating time differences ! This function may be useful, or may not be useful.
Definition:

```
unsigned int sctp_getTime(void);
```

6.4 ULP-to-SCTP

6.4.1 sctp_initLibrary()

This function will open raw sockets for capturing SCTP packets (IPv4 and if possible, IPv6, too) from the network and initialize the timer list.

6.4.7 sctp_send()

sctp_send() is used by the ULP to send data as data chunks. There are quite a few parameters that can be or must be passed along:

```
associationID
```

`length` length of chunk data.

`flags` `SCTP_MSG_PEEK` or `SCTP_MSG_DEFAULT`

It returns 1 if association does not exist, 0 if okay.
Definition:

```
unsigned short sctp_receive(unsigned int associationID, unsigned short streamID,
                            unsigned char *buffer, unsigned int *length, unsigned int flags)
```

6.4.10 sctp_getAssocDefaults()

This function returns all the default values of an SCTP instance, i.e. it fills the **SCTP_InstanceParameters** structure. Values that are not supported yet, but already integrated in this API are set 0 by default (here: `maxSendQueue`, `maxRecvQueue`).
The function takes the following parameters:

`SCTP_InstanceName` instance name

6.4.13 sctp_setAssocStatus()

This function may be used to set a number of values or parameters that belong to a certain (and already existing) association. Some values are not supported yet, but already integrated in this API (i.e. `maxSendQueue`, `maxRecvQueue`).
The function takes the following parameters:

`associationID` ID of association.

`status` pointer to the structure to be filled

It returns -1 if the association does not exist, 0 if okay.
Definition:

```
int sctp_setAssocStatus(unsigned int associationID, SCTP_AssociationStatus* new_status);
```

6.4.14 sctp_getPathStatus()

This function may be used to retrieve a number of path specific values or parameters within an existing

6.4.19 sctp_changeHeartBeat()

sctp_changeHeartBeat

`streamSN` pointer to stream sequence number of the data chunk that was not sent.

`protocolId` pointer to the protocol ID of the unsent chunk

Function currently not implemented, so it returns -1, always.
Definition:

```
int sctp_receiveUnsent(unsigned int associationID, unsigned char *buffer,
                       unsigned int *length, unsigned short *streamID,
                       unsigned short *streamSN,unsigned int* protocolId);
```

6.4.23 sctp_receiveUnacked()

sctp_receiveUnacked() is currently NOT implemented ! Will return messages that were sent, but have not

6.5.1 DataArrive Notification

Indicates that new data has arrived from peer (chapter 10.2.A). The parameters passed with this callback are (in this order):

unsigned int association ID

unsigned int stream ID

unsigned int length of data

unsigned int protocol ID

unsigned int unordered flag (uses constants SCTP_UNORDERED_DELIVERY==1,or SCTP_ORDERED_DELIVERY==1,or SCTP

6.5.4 CommunicationUp Notification

Indicates that an association has been successfully established (chapter 10.2.D). The parameters passed with this callback are (in this order):

unsigned int association ID

unsigned short status, type of event; the following events are defined:

6.5.6 CommunicationError Notification

Indicates that communication had an error (chapter 10.2.F). Currently not implemented !

unsigned int association ID

7.2 An Echo Server

The echo server is structured similarly to the discard server (see section 7.1). The main difference of their functionalities is, of course, in the callback functions. The general control flow is as follows:

the **main**