Document:	T10/98-191 revision 0
Date:	98-07-15
To:	NCITS T10 SCSI Working Group
From:	Keith W. Parker <diogenes@europa.com>
Subject:	SCSI Socket Services (SSS) Review 98-07-15

SCSI Socket Services (SSS) Review 98-07-15

Project T10/1246-D

Request for tentative assignment of SCSI command numbers

I am requesting that these SCSI command numbers be tentatively assigned to
"SCSI Socket Services (SSS) Command Set" project T10/1246-D.
Final assignment would not occur until the SSS has been discussed and voted upon.
The tentative assignment would allow people that are ready to work on developmental phase implementation to begin without requiring changing code once final command numbers are assigned.

They all need to be **OPTIONAL** for **ALL** SCSI device types.
They are all SCSI Group Code #4 (16 byte) commands.

```
/* "Socket API over SCSI RPC" command numbers */
#define SSS_RPC_MGMT_GET        0x90
#define SSS_RPC_MGMT_PUT        0x91
#define SSS_RPC_XFER_GET        0x92
#define SSS_RPC_XFER_PUT        0x93

/* "Internet IP packets over SCSI" command numbers */
#define SSS_PKT_MGMT_GET        0x94
#define SSS_PKT_MGMT_PUT        0x95
#define SSS_PKT_XFER_GET        0x96
#define SSS_PKT_XFER_PUT        0x97
```

According to "SCSI Primary Commands-2 (SPC-2) T10/1236-D rev.3 (98-05-22) Table B.2,
these command numbers are currently unassigned.

Does anyone know of any conflicts with these SCSI command numbers being assigned to the
"SCSI Socket Services (SSS) Command Set" project T10/1246-D?

All commands use the exact same format with the exception that the Pkt_count and Data_len fields specify the actual amounts to be transferred during SSS_xxx_xxxx_PUT commands and the maximum amounts that can be transferred during the SSS_xxx_xxxx_GET commands.

```
Byte  |Bits| Type   | Field Name
 0    |  8 | BE_uchar SCSI_Cmd_Num
 1    |  8 | BE_uchar Func_Code
 2-3  | 16 | BE_short Pkt_Count
 4-7  | 32 | BE_long  Data_Len
 8-11 | 32 | BE_long  Cmd_Key
12-13 | 16 | BE_short Channel_Token
14    |  8 | BE_uchar Func_Flags
15    |  8 | BE_uchar Control ( SAM r18 5.1, 5.6 ; SAM-2 r5a 5.1.2, 5.6 )
```

SSS_RPC_xxxx_xxx - These commands support the SSS Remote Procedure Call (RPC) mechanism.
This connects systems together at the "top" of the protocol stack.
This method has the highest performance/management potential since it does not use the TCP/IP protocol stack. It is the most complex to implement.

SSS_PKT_xxxx_xxx - These commands support the SSS "IP Packets over SCSI" mechanism.
This connects systems together at the "bottom" of the protocol stack.
This method is much easier to implement if a system already has a TCP/IP protocol stack.

This is an alternate method of accomplishing the same objective: providing a high performance (i.e. SCSI bandwidth) Platform/Device Independent (PDI) "Socket" Application Programming Interface (API).

This is new to the SSS project. It an extension for the sake of pragmatism, it is much easier (therefore more likely) to implement.
It also addresses the issue of situations that are intrinsically packet oriented such as using SCSI as an expansion bus for bridges, routers, hubs and switches or as an intermediate link in a data stream that has already been broken down into packets.
While the primary focus is Internet IP Packets (IPv4 & IPv6), other packets (802.3, USB, IrDA, etc.) may be transported.

SSS_xxx_MGMT_xxx - These commands support communication management functions. These are handled separately from the transfer (_XFER_) functions because they may be relatively large and infrequently used so it is desirable for them to be unloaded from memory when not needed.

SSS_xxx_XFER_xxx - These commands support the actual transfer of data.
These are more compact and often used so they need to be left in memory.

SSS_xxx_xxxx_GET & **SSS_xxx_xxxx_PUT -** These commands GET and PUT packets into the transport buffers. They allow symmetric communications even when one of the SCSI devices is "Initiator Only" capable.

I am preparing a T10 document (T10/98-183 "SSS & IP packets over SCSI") that contains all information necessary to implement the "IP packets over SCSI" portion of the SSS proposed standard. This document of excerpts from T10/1246 is to allow the implementation of "IP packets over SCSI" without the distraction of the "Socket API over SCSI RPC".

Status

The bad news is that the project is outrageously behind schedule.
As originator and Technical Editor for the project the responsibility
for this delay rests on my shoulders alone.
I don't have the discipline of being paid to follow a specific
schedule so I've been in an "academic open loop" mode of operation.

When I started the project I focused initially (and exclusively) on the "purist" perspective of connecting systems together at the "top" of the Protocol Stack by implementing the Socket API (Application Programming Interface) operating on top of a Remote Procedure Call (RPC) SCSI command set.
The "Socket API over SCSI RPC" approach offers the potential for the highest performance and most flexible management. Unfortunately, it comes at the cost of a much more extensive implementation effort.
This made the "first step" an extremely high one.
For personal reasons, this bite was too big for me to chew by myself. Live and learn, the hard way.

The good news is that I have succumbed to the "pragmatic" argument for implementing "Internet IP Packets over SCSI" (at least as a first step). This will allow connecting systems together at the "bottom" of the Protocol Stack.

Fortunately, this is MUCH simpler to implement.
Other than house-keeping, only 2 functions are necessary: getting and putting IP packets.
It also makes more sense for using SCSI as an expansion bus for bridges, routers, hubs, or switches.

Eventually, once "Internet IP Packets over SCSI" is operational, the "Socket API over SCSI RPC" can be implemented as a performance/management improving extension.
Then systems will be able to choose the optimal technique for each application.

My objective is to have the documentation in a useable form to distribute it to computer science departments in time to be used as a class development project for Fall term.

Keith W. Parker <diogenes@europa.com>
Technical Editor, NCITS T10/1246-D
SCSI Socket Services (SSS) Command Set