MD5 Logical Unit Identifier

(T10/02-035r1)
Dave Peterson, Cisco Systems, Inc.

It is highly desirable to identify logical units using a unique identifier. Legacy and non-compliant devices that do not support VPD page 0x83, specifically identifier type 3h or 2h, cause problems for applications using commands such as Extended Copy. Without the VPD page 0x83 type 3h or 2h unique identifier, applications are forced to find and use, alternate, hopefully unique identifiers, for the logical unit. Most likely, this alternate identifier will be a VPD page 0x83 type 0h or 1h value or the unit serial number obtained via VPD page 0x80.

So, for Extended Copy, the E4 target descriptor is much preferred since it provides the capability to uniquely identify the logical unit (i.e., not the address or port where the logical unit resides). But, currently there is no guaranteed way to completely pass these alternate identifiers via the E4 target descriptor since the E4 target descriptor allows for a maximum of 20 bytes for the identifier field. As a result, the 3PC application must pick one of the other target descriptor formats.

Table 1: Current identifiers

Type	Description	Notes
VPD page 0x80	Unit serial number page	Provides no guarantee that the identifier is unique, *n* bytes.
VPD page 0x83, type 0h	Vendor specific	Vendor specific data, *n* bytes.
VPD page 0x83, type 1h	T10 vendor identification	Vendor ID plus vendor specific data, *n* bytes.
VPD page 0x83, type 2h	EUI-64	Wouldn’t have to do this proposal if the vendor supported this.
VPD page 0x83, type 3h	NAA	Wouldn’t have to do this proposal if the vendor supported this and is the recommended flavor.

Proposal: add a new Identifier Type and specify a method (MD5) for generating a unique identifier for a logical unit. This unique identifier is intended to be generated by a bridge device that front ends a legacy or non-compliant device.

The MD5 algorithm (see RFC 1321) generates a 128 bit (16 byte) message digest of the supplied message input and the message input may be of an arbitrary length. The MD5 algorithm is not an encryption algorithm. As such, there is no feasible way to determine the input, given the output.

Additional Identifier type (table 257):

07h = MD5 Logical Unit Identifier

x.x MD5 Logical Unit Identifier format

If the identifier type is MD5 Logical Unit Identifier (7h), the IDENTIFIER field has the format shown in table 2. The MD5 Logical Unit Identifier is intended for use in configurations where a logical unit provides no unique identification (e.g., if a logical unit does not support Device Identification page 0x83, type 2h or 3h, a bridge device may return a MD5 Logical Unit Identifier type for that logical unit).

Table 2 — MD5 LOGICAL UNIT IDENTIFIER field format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1								
2								
3								
4								
5								
6								
7								
8				MD5 LOGICAL UNIT IDENTIFIER				
9								
10								
11								
12								
13								
14								
15								(LSB)

The MD5 LOGICAL UNIT IDENTIFIER field contains the message digest of the supplied message input. The message digest shall be generated using the MD5 message-digest algorithm as specified in [RFC1321] with the following information as message input:

1. standard INQUIRY data VENDOR IDENTIFICATION field (8 bytes);
2. standard INQUIRY data PRODUCT IDENTIFICATION field (8 bytes);
3. standard INQUIRY data PRODUCT REVISION LEVEL field (8 bytes);
4. Vital Product Data page 0x80 (n bytes);
5. Vital Product Data page 0x83, type 0h, vendor specific IDENTIFIER field (n bytes); and
6. Vital Product Data page 0x83, type 1h, T10 vendor IDENTIFIER field (n bytes).

Message input shall be in network byte order (i.e., big endian).

If a field or page is not available, the message input shall be 8 bytes of ASCII blank characters (0x20).

The uniqueness of the MD5 Logical Unit Identifier is dependent upon the relative degree of randomness (i.e., the entropy) of the message input. If it is found that two or more logical units have the same MD5 Logical Unit Identifier, the implementation should determine, in a vendor specific manner, whether or not the logical units are the same entities.