



**TR-140**

TR-069 Data Model for Storage Service Enabled Devices

**Issue Number: 1.00**  
**Issue Date: August 2007**

**Notice:**

The DSL Forum is a non-profit corporation organized to create guidelines for DSL network system development and deployment. This Technical Report has been approved by members of the Forum. This document is not binding on the DSL Forum, any of its members, or any developer or service provider. This document is subject to change, but only with approval of members of the Forum.

This publication may incorporate intellectual property. The DSL Forum encourages but does not require declaration of such intellectual property. For a list of declarations made by DSL Forum member companies, please see [www.dslforum.org](http://www.dslforum.org).

©2007 Digital Subscriber Line Forum. All Rights Reserved.

DSL Forum Technical Reports may be copied, downloaded, stored on a server or otherwise re-distributed in their entirety only. The text of this notice must be included in all copies.

Notwithstanding anything to the contrary, the DSL Forum makes no representation or warranty, expressed or implied, concerning this publication, its contents or the completeness, accuracy, or applicability of any information contained in this publication. No liability of any kind shall be assumed by the DSL Forum as a result of reliance upon any information contained in this publication. The DSL Forum does not assume any responsibility to update or correct any information in this publication.

**Version History**

<b>Version umber</b>	<b>Version Date</b>	<b>Version Editor</b>	<b>Changes</b>
Issue 1.0	August 2007	Tim Spets, Westell John Blackford, 2Wire	Original

Technical comments or questions about this document should be directed to:

**Editors:**                      Tim Spets                      Westell                      [tspets@westell.com](mailto:tspets@westell.com)  
   John Blackford                      2Wire                      [jblackford@2wire.com](mailto:jblackford@2wire.com)

**DSLHome WG Chairs**      Greg Bathrick                      PMC Sierra  
   Heather Kirksey                      Motive

**Table of Contents**

<b>1</b>	<b>PURPOSE</b> .....	<b>7</b>
<b>2</b>	<b>SCOPE</b> .....	<b>7</b>
2.1	DEFINITIONS .....	8
2.2	ABBREVIATIONS .....	9
2.3	CONVENTIONS .....	9
<b>3</b>	<b>REFERENCES</b> .....	<b>9</b>
<b>4</b>	<b>PARAMETER DEFINITIONS</b> .....	<b>10</b>
<b>5</b>	<b>NOTIFICATION REQUIREMENTS</b> .....	<b>25</b>
<b>6</b>	<b>PROFILE DEFINITIONS</b> .....	<b>25</b>
6.1	NOTATION.....	25
6.2	BASELINE PROFILE.....	25
6.3	USER ACCESS PROFILE .....	26
6.4	GROUP ACCESS PROFILE.....	27
6.5	FTP SERVER PROFILE .....	28
6.6	SFTP SERVER PROFILE .....	28
6.7	HTTP SERVER PROFILE .....	28
6.8	HTTPS SERVER PROFILE .....	29
6.9	VOLUME CONFIG PROFILE .....	29
6.10	RAID PROFILE.....	30
6.11	FOLDER QUOTA PROFILE .....	30
6.12	VOLUME THRESHOLD PROFILE .....	30
6.13	NETWORK SERVER PROFILE.....	31
<b>7</b>	<b>USE CASES</b> .....	<b>31</b>
7.1	BASIC MANAGED STORAGE SERVICE.....	31
7.2	REMOTE STORAGE BACK-UP SERVICE .....	32
7.3	REMOTE ACCESS OF STORAGE SERVICE CONTENT FROM A REMOTE LOCATION ...	32
<b>ANNEX A: THEORY OF OPERATIONS</b> .....		<b>33</b>
A.1	PHYSICAL STORAGE THEORY OF OPERATIONS.....	33
A.1.1	<i>Physical Medium Discovery</i> .....	34
A.1.2	<i>Logical Volume Discovery</i> .....	34
A.1.3	<i>Folder Discovery</i> .....	34
A.1.4	<i>Access Permissions</i> .....	34
A.1.5	<i>Data Operations</i> .....	34
A.2	FILE STRUCTURE MANAGEMENT THEORY OF OPERATIONS .....	34
A.2.1	<i>File Structure Management</i> .....	35
A.2.2	<i>FTP Root File Directory</i> .....	35
A.3	ACCESS AND SECURITY THEORY OF OPERATION .....	36
A.3.1	<i>Security through UserAccess and GroupAccess profiles</i> .....	36
A.3.2	<i>Levels of Security</i> .....	36

*A.3.3 Recommended Object Creation Pattern:* ..... 39

A.4 DATA MODEL INTEGRITY THEORY OF OPERATIONS ..... 39

*A.4.1 Physical Medium* ..... 39

*A.4.2 Storage Array*..... 39

*A.4.3 Logical Volume*..... 40

*A.4.4 Folder*..... 40

*A.4.5 UserAccount and UserGroup*..... 41

A.5 FILE RETRIEVAL THEORY OF OPERATIONS ..... 42

A.6 THRESHOLDING THEORY OF OPERATIONS..... 42

**ANNEX B: RAID TYPE DESCRIPTIONS ..... 43**

B.1 BASIC RAID LEVELS ..... 43

*B.1.1 Linear RAID*..... 43

*B.1.2 Level 0*..... 43

*B.1.3 Level 1*..... 43

*B.1.4 Level 2*..... 43

*B.1.5 Level 3*..... 44

*B.1.6 Level 4*..... 44

*B.1.7 Level 5*..... 44

*B.1.8 Level 6*..... 44

B.2 COMBINATION RAID LEVELS ..... 45

*B.2.1 Level 10*..... 45

*B.2.2 Level 01*..... 45

*B.2.3 Level 30*..... 45

*B.2.4 Level 50*..... 46

*B.2.5 Level 60*..... 46

**Summary**

This document, TR-069 Data Model for Storage Service Devices, permits the remote management of Storage Service devices via CWMP as defined in TR-069 and TR-106. It covers the data model for describing a Storage Service device as well as rules regarding notifications on parameter value change. General use cases are also described including standard data model profiles that would typically be seen while remotely managing a device of this nature.

## DSL Forum Technical Report TR-140

### TR-069 Data Model for Storage Service Devices

#### 1 Purpose

This document, TR-069 Data Model for Storage Service Devices, permits the remote management of Storage Service devices via CWMP as defined in [2] and [3]. It covers the data model for describing a Storage Service device as well as rules regarding notifications on parameter value change. General use cases are also described including standard data model profiles that would typically be seen while remotely managing a device of this nature.

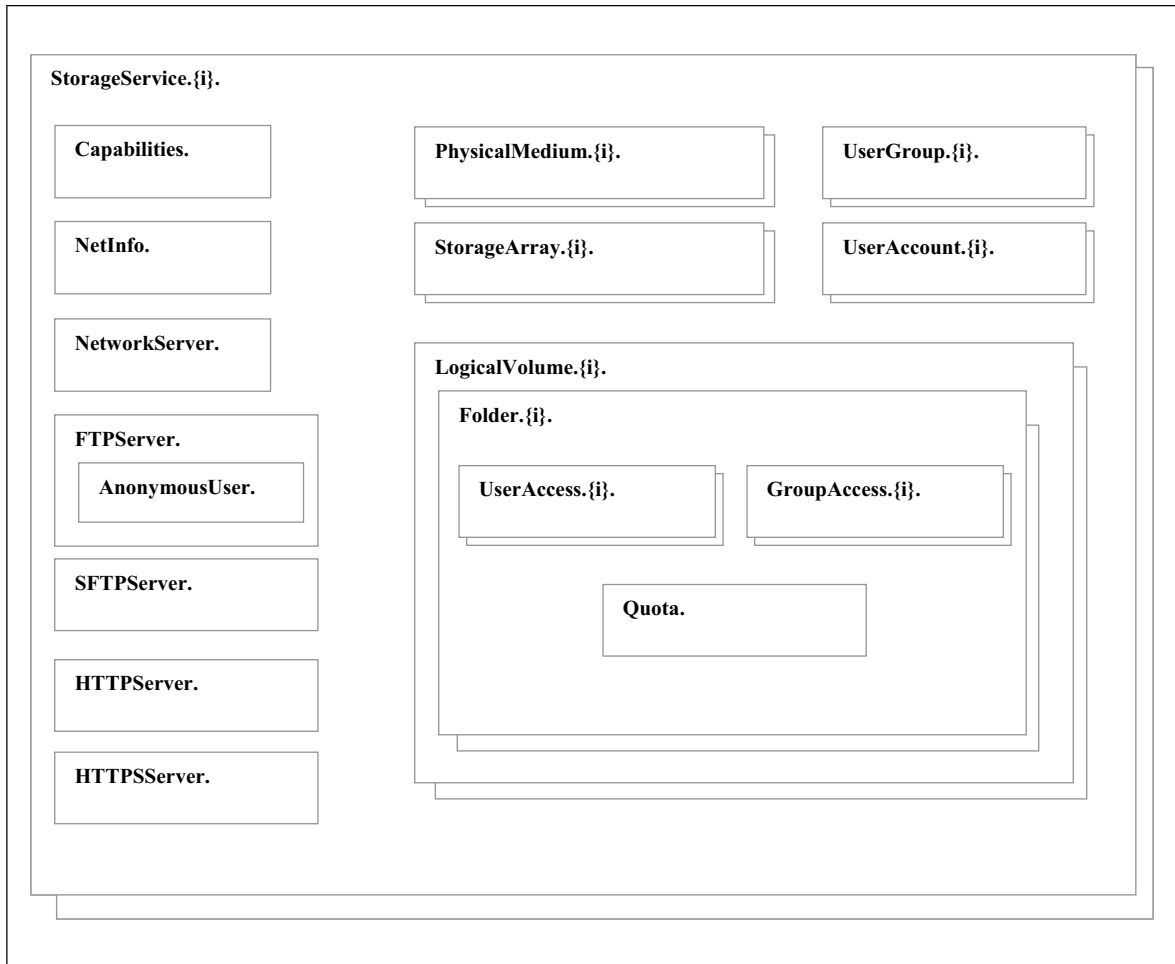
This document defines StorageService as the container associated with the provisioning of objects for Storage Service devices. CPE making use of a StorageService object MUST adhere to all of the data-hierarchy requirements defined in [3]. In the context of [3], the StorageService object is a service object. As such, individual CPE devices can contain one or more of these objects within their Services object alongside the generic data objects defined in [3]. The presence of more than one StorageService object would be appropriate primarily where a CPE device serves as a management proxy for other non-TR-069 capable Storage Service devices. For example, an internet gateway device might serve as a management proxy for one or more non-TR-069 capable Network Attached Storage (NAS) Devices.

#### 2 Scope

This document defines the data model for provisioning a CPE device that maintains a storage service such as a NAS device by an Auto-Configuration Server (ACS) using the mechanism defined in [2].

The goals of this specification are as follows:

- Enable troubleshooting and remote configuration of Storage Service devices from an ACS.
- Accommodate Storage Service devices that are either embedded as part of an internet gateway device, as defined in [2], or a standalone device.



**Figure 1 – StorageService Object Structure**

Figure 1 depicts the StorageService object structure as seen in the Parameter Definitions section. This image provides a high-level overview of the different objects that exist in this data model and how they are associated.

## 2.1 Definitions

This document defines the following terms:

**AFP:** The Apple Filing Protocol is a network protocol used to exchange files.

**NAT:** When a NAS device resides behind a residential gateway then typically Network Address Translation (NAT) will be used to allow the NAS device to communicate to the Internet through the residential gateway while protecting the IP address known to the NAS.

**NFS:** The Network File System is a network protocol used to exchange files.

**RAID:** A Redundant Array of Inexpensive Drives allows multiple physical media to be combined into a larger (in either size or number) storage concept.

**SMB:** Server Message Block is a network protocol used to exchange files.

**WebDAV:** Web-based Distributed Authoring and Versioning is used to enhance HTTP (and HTTPS) for publishing files.

## 2.2 Abbreviations

This document defines the following abbreviations:

ACS	Auto-Configuration Server
CPE	Customer Premises Equipment
CWMP	CPE WAN Management Protocol
FTP	File Transfer Protocol
HTTP	Hyper-Text Transfer Protocol
HTTPS	HTTP over SSL or TLS
NAS	Network Attached Storage
SMART	Self-Monitoring, Analysis, and Reporting Technology
SFTP	SSH FTP
SSH	Secure Shell
UPnP	Universal Plug and Play

## 2.3 Conventions

In this document, several words are used to signify the requirements of the specification. These words are often capitalized.

<b>MUST</b>	This word, or the adjective “REQUIRED”, means that the definition is an absolute requirement of the specification.
<b>MUST NOT</b>	This phrase means that the definition is an absolute prohibition of the specification.
<b>SHOULD</b>	This word, or the adjective “RECOMMENDED”, means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications must be understood and carefully weighted before choosing a different course.
<b>MAY</b>	This word, or the adjective “OPTIONAL”, means that this item is one of an allowed set of alternatives. An implementation that does not include this option <b>MUST</b> be prepared to inter-operate with another implementation that does include the option.

## 3 References

The following DSL Forum Technical Reports and other references contain provisions, which, through reference in this text, constitute provisions of this document. At the time

of publication, the editions indicated each of the listed documents were valid. All Technical Reports and other references are subject to revision; users of this document are therefore encouraged to investigate the possibility of applying the most recent edition of the Technical Report and other references listed below. A list of the currently valid DSL Forum Technical Reports is published at [www.dslforum.org](http://www.dslforum.org).

NOTE – The reference to a document within this document does not give it, as a standalone document, the status of a Technical Report.

- [1] RFC 2119, *Key words for use in RFCs to Indicate Requirement Levels*, <http://www.ietf.org/rfc/rfc2119.txt>
- [2] TR-069 Amendment 1, *CPE WAN Management Protocol*, DSL Forum Technical Report
- [3] TR-106 Amendment 1, *Data Model Template for TR-069-Enabled Devices*, DSL Forum Technical Report

## 4 Parameter Definitions

**Table 1** lists the objects associated with a Storage Service device and its parameters. The notation used to indicate the data type of each parameter, and the notation associating with multi-instance objects, follows the notation defined in [3].

**Table 1 – Parameter list for a Storage Service device**

Name <sup>1</sup>	Type	Write <sup>2</sup>	Description	Object Default <sup>3</sup>
.StorageService.{i}.	object	-	The Service Object for a Storage Service device.	-
Enable	boolean	W	Enables or Disables the entire Storage mechanism.	-
PhysicalMediumNumberOfEntries	unsignedInt	-	The number of instances of PhysicalMedium.	-
StorageArrayNumberOfEntries	unsignedInt	-	The number of instances of StorageArray.	-
LogicalVolumeNumberOfEntries	unsignedInt	-	The number of instances of LogicalVolume.	-
UserAccountNumberOfEntries	unsignedInt	-	The number of instances of UserAccount.	-
UserGroupNumberOfEntries	unsignedInt	-	The number of instances of UserGroup.	-
.StorageService.{i}.Capabilities.	object	-	The overall capabilities of a Storage Service device. This is a constant read-only object, meaning that only a firmware upgrade will cause these values to be altered.	-
FTPCapable	boolean	-	Does this device contain an FTP server allowing clients to access the data via an FTP client?	-

<sup>1</sup> The name of a Parameter is formed from the concatenation of the base path, (refer to [3] section 2.1), the object name shown in the yellow header, and the individual Parameter name.

<sup>2</sup> "W" indicates the parameter MAY be writable (if "W" is not present, the parameter is defined as read-only). For an object, "W" indicates object instances can be Added or Deleted.

<sup>3</sup> The default value of the parameter on creation of an object instance via TR-069. If the default value is an empty string, this is represented by the symbol <Empty>.

Name <sup>1</sup>	Type	Write <sup>2</sup>	Description	Object Default <sup>3</sup>
SFTPCapable	boolean	-	Does this device contain an SSH FTP server allowing clients to access the data via an SFTP client?	-
HTTPCapable	boolean	-	Does this device contain an HTTP server allowing clients to access the data via an HTTP client?	-
HTTPS capable	boolean	-	Does this device contain an HTTPS server allowing clients to access the data via an HTTPS client?	-
HTTPWritable	boolean	-	Does this device contain an HTTP server that supports creating files via an HTTP PUT/POST mechanism that would allow an HTTP client to upload files via HTTP? This is also sometimes referred to as "WebDAV" support.	-
SupportedNetworkProtocols	string	-	Comma-separated list of supported application-level network protocols. The possible set of supported network protocols is an enumeration <sup>4</sup> of the following strings:  "SMB" "NFS" "AFP"	-
SupportedFileSystemTypes	string	-	Comma-separated list of supported FileSystems Types. The possible set of supported file system types is an enumeration <sup>4</sup> of the following strings:  "FAT16" "FAT32" "NTFS" "HFS" "HFS+ " "HSFJ" "ext2" "ext3" "XFS" "REISER"	-
SupportedRaidTypes	string	-	Comma-separated list of supported RAID types. The possible set of supported RAID types is an enumeration <sup>4</sup> of the following strings:  "RAID0" "RAID1" "RAID2" "RAID3" "RAID4" "RAID5" "RAID6" "RAID10" "RAID0+1" "RAID30" "RAID50" "RAID60"	-

<sup>4</sup> A vendor MAY also extend the set of values of an enumeration. If this is done, the vendor-specified values MUST be in the form "X\_<VENDOR>\_VendorSpecificValue". The total length of such a string MUST NOT exceed 31 characters.

Name <sup>1</sup>	Type	Write <sup>2</sup>	Description	Object Default <sup>3</sup>
VolumeEncryptionCapable	boolean	-	Does this device have the ability to encrypt and decrypt Logical Volumes as they are stored and retrieved?	-
.StorageService.{i}.NetInfo.	object	-	This object provides general LAN network information about this device.	-
HostName	string(64)	W	Logical name which identifies the device on the local network. This is the first segment of a fully qualified domain name. Combining this parameter, a "." and the DomainName parameter will result in a fully qualified domain name.	-
DomainName	string(255)	W	Domain name for the device on the local network. Combining the HostName, a "." and this parameter will result in a fully qualified domain name (FQDN). For example, if the HostName contains "myLaptop" and the DomainName contains "dsl.sp1.com", then the FQDN would be "myLaptop.dsl.sp1.com".	-
.StorageService.{i}.UserGroup.{i}.	object	W	This object provides information about each user group configured on this device, which allows the grouping of user accounts for easier maintenance of permissions.  The unique <sup>5</sup> key for this object is GroupName	-
Enable	boolean	W	Enables or disables this group.	False
GroupName	string(64)	W	The unique name of the group.	<Empty>
.StorageService.{i}.UserAccount.{i}.	object	W	This object provides information about each user configured on this device, which provides a means for controlling access to the device.  The unique <sup>5</sup> key for this object is GroupName	-
Enable	boolean	W	Enables or disables this user.	False
Username	string(64)	W	The unique name of the user. Also used for authentication.	<Empty>
Password	string(64)	W	Password used to authenticate the user when connecting to the Storage Service Device.  When read, this parameter returns an empty string, regardless of the actual value.	-
UserGroupParticipation	string(1024)	W	This is a comma-separated list of UserGroup references. Each User Group referenced by this parameter MUST exist within the same StorageService instance. Each reference can be either in the form of ".UserGroup.{i}" or a fully qualified object name. For example:  .UserGroup.3  This indicates that this User Account is associated with UserGroup instance #3 on this StorageService device.	-

<sup>5</sup> When an object is writeable and contains unique keys, the CPE will create the object with the default key and will not allow another object creation until the key is modified and unique.

Name <sup>1</sup>	Type	Write <sup>2</sup>	Description	Object Default <sup>3</sup>
AllowFTPAccess	boolean	W	Enables or disables access via FTP (including SSH FTP access) for this user.	False
AllowHTTPAccess	boolean	W	Enables or disables access via HTTP (including HTTPS access) for this user.	False
.StorageService.{i}.NetworkServer.	object	-	This object allows the control of network layer protocols authorization enforcement.	-
AFPEnable	boolean	W	Enables or disables the AFP network protocol.	-
NFSEnable	boolean	W	Enables or disables the NFS network protocol.	-
SMBEnable	boolean	W	Enables or disables the SMB network protocol.	-
NetworkProtocolAuthReq	boolean	W	If this parameter is set to False then the device MUST NOT attempt to request login credentials or authenticate access from network layer protocols such as AFP, NFS, and SMB. If this parameter is set to True then the device MUST attempt to request login credentials or authenticate access from network layer protocols such as AFP, NFS, and SMB by using the UserAccount instances found on this device.	-
.StorageService.{i}.FTPServer.	object	-	This object allows the configuration of the FTP server.	-
Enable	boolean	W	Enables or disables the FTP server.	-
Status	string	-	The current status of this FTP server. This is an enumeration of the following status strings: "Enabled" "Disabled" "Error"	-
MaxNumUsers	unsignedInt [1:32]	W	Maximum number of users allowed to log in to the device at once via FTP.	-
IdleTime	unsignedInt [0:600]	W	Maximum amount of time in seconds that the FTP socket will remain open without any activity. If set to 0 an infinite timeout will apply.	-
PortNumber	unsignedInt [0:65535]	W	The port number that the FTP server is listening on. The default FTP port is 21.	-
.StorageService.{i}.FTPServer.Anonymous User.	object	-	This object allows the configuration of anonymous FTP access.	-
Enable	boolean	W	Enables or disables support for anonymous access into the FTP server	-

Name <sup>1</sup>	Type	Write <sup>2</sup>	Description	Object Default <sup>3</sup>
StartingFolder	string(256)	W	This is a reference to an instance of a Folder object that acts as the home directory for anonymous FTP access. The Folder referenced by this parameter MUST exist within the same StorageService instance. Each reference can be either in the form of ".LogicalVolume.{i}.Folder.{i}" or a fully qualified object name. For example: .LogicalVolume.2.Folder.3  This indicates a starting folder on this StorageService enabled device of Folder instance #3 contained within Logical Volume instance #2.	-
ReadOnlyAccess	boolean	W	If this is set to True, then the anonymous user is limited to only retrieval of files from the Storage Service (no deletions, copies, creations or uploads). If this is set to False, then the anonymous user has full permissions within the folder specified in the StartingFolder parameter. This SHOULD default to True.	-
.StorageService.{i}.SFTPServer.	object	-	This object allows the configuration of the SSH FTP server.	-
Enable	boolean	W	Enables or disables the SSH FTP server.	-
Status	string	-	The current status of this SSH FTP server. This is an enumeration of the following status strings:  "Enabled" "Disabled" "Error"	-
MaxNumUsers	unsignedInt[1:32]	W	Maximum number of users allowed to log in to the device at once via SFTP.	-
IdleTime	unsignedInt[0:600]	W	Maximum amount of time in seconds that the SFTP socket will remain open without any activity. If set to 0 an infinite timeout will apply.	-
PortNumber	unsignedInt[0:65535]	W	The port number that the SSH FTP server is listening on. The default SFTP port is 115.	-
.StorageService.{i}.HTTPServer.	object	-	This object allows the configuration of the HTTP server.	-
Enable	boolean	W	Enables or disables the HTTP server.	-
Status	string	-	The current status of this HTTP server. This is an enumeration of the following status strings:  "Enabled" "Disabled" "Error"	-
MaxNumUsers	unsignedInt [1:32]	W	Maximum number of users allowed to log in to the device at once via HTTP.	-
IdleTime	unsignedInt [0:600]	W	Maximum amount of time in seconds that the HTTP socket will remain open without any activity. If set to 0 an infinite timeout will apply.	-
HTTPWritingEnabled	boolean	-	Is support for the HTTP PUT/POST mechanism (WebDAV) enabled?	-

Name <sup>1</sup>	Type	Write <sup>2</sup>	Description	Object Default <sup>3</sup>
PortNumber	unsignedInt [0:65535]	W	The port number that the HTTP server is listening on. The default HTTP port is 80.	-
AuthenticationReq	boolean	W	If True, HTTP will require login prior to access (basic or digest authentication)	-
.StorageService.{i}.HTTPSServer.	object	-	This object allows the configuration of the HTTPS server.	-
Enable	boolean	W	Enables or disables the HTTPS server.	-
Status	string	-	The current status of this HTTPS server. This is an enumeration of the following status strings:  "Enabled" "Disabled" "Error"	-
MaxNumUsers	unsignedInt[1: 32]	W	Maximum number of users allowed to log in to the device at once via HTTPS.	-
IdleTime	unsignedInt[0: 600]	W	Maximum amount of time in seconds that the HTTPS socket will remain open without any activity. If set to 0 an infinite timeout will apply.	-
HTTPWritingEnabled	boolean	-	Is support for the HTTP PUT/POST mechanism (WebDAV) enabled?	-
PortNumber	unsignedInt[0: 65535]	W	The port number that the HTTPS server is listening on. The default HTTPS port is 443.	-
AuthenticationReq	boolean	W	If True, HTTPS will require login prior to access (basic or digest authentication)	-
.StorageService.{i}.PhysicalMedium.{i}.	object	-	This object provides information about each physical medium connected to this device.	-
Name	string(64)	W	A user-friendly name for this physical storage medium.	-
Vendor	string(64)	-	The vendor of this physical storage medium.	-
Model	string(128)	-	The model name/number of this physical storage medium.	-
SerialNumber	string(64)	-	The serial number of this physical storage medium.	-
FirmwareVersion	string(64)	-	The firmware version for firmware contained within the physical medium's controller.	-

Name <sup>1</sup>	Type	Write <sup>2</sup>	Description	Object Default <sup>3</sup>
ConnectionType	string	-	Enumerated <sup>4</sup> type indicating the method of connection to this storage device: <i>"USB 1.1"</i> <i>"USB 2.0"</i> <i>"IEEE1394"</i> <i>"IEEE1394b"</i> <i>"IDE"</i> <i>"EIDE"</i> <i>"ATA/33"</i> <i>"ATA/66"</i> <i>"ATA/100"</i> <i>"ATA/133"</i> <i>"SATA/150"</i> <i>"SATA/300"</i> <i>"SCSI-1"</i> <i>"Fast SCSI"</i> <i>"Fast-Wide SCSI"</i> <i>"Ultra SCSI"</i> <i>"Ultra Wide SCSI"</i> <i>"Ultra2 SCSI"</i> <i>"Ultra2 Wide SCSI"</i> <i>"Ultra3 SCSI"</i> <i>"Ultra-320 SCSI"</i> <i>"Ultra-640 SCSI"</i> <i>"SSA"</i> <i>"SSA-40"</i> <i>"Fibre Channel"</i>	-
Removable	boolean	-	Is this physical storage medium removable? Removable storage implies that the removal action is part of normal operations and is expected to leave the data on the removable storage intact.	-
Capacity	unsignedInt	-	The formatted capacity of the physical storage medium in MB.	-
Status	string	-	Enumerated type indicating the general activity status of this physical storage medium: <i>"Online"</i> <i>"Standby"</i> <i>"Offline"</i>	-
Uptime	unsignedInt	-	Number of hours since boot. This MAY reflect S.M.A.R.T. PowerOnHours.	-
SMARTCapable	boolean	-	Is this physical medium capable of reporting S.M.A.R.T. statistics?	-
Health	string	-	Enumerated <sup>4</sup> type indicating the general health of this physical storage medium: <i>"OK"</i> <i>"Failing"</i> <i>"Error"</i>  Note: Health MAY be obtained from S.M.A.R.T. data where available.	-
HotSwappable	boolean	-	Is this physical medium capable of being removed while the device is powered up? Hot-Swappable storage does not imply or infer Removable storage as hot-swappable is an operation taken only when the disk has failed and needs to be replaced. The data on the hot-swapped storage will not remain intact.	-

Name <sup>1</sup>	Type	Write <sup>2</sup>	Description	Object Default <sup>3</sup>
.StorageService.{i}.StorageArray.{i}.	object	W	<p>This object provides information about each storage array (RAID) configured on this device.</p> <p>Creating an instance of this object generates a disabled StorageArray instance. Before this new StorageArray instance can be enabled (via a SetParameterValues command), it MUST have the following parameters configured: Name, RaidType, and PhysicalMediumReference. The unique<sup>5</sup> key for this object is the PhysicalMediumReference parameter, but once an instance is enabled the following parameters become immutable for the life of the instance: Name, RaidType, and PhysicalMediumReference.</p>	-
Name	string(64)	W	<p>A user-friendly name for this Storage Array.</p> <p>Once this instance becomes enabled, this parameter will be immutable for the life of the instance.</p>	<Empty>

Name <sup>1</sup>	Type	Write <sup>2</sup>	Description	Object Default <sup>3</sup>
Status	string	-	<p>The current status of this StorageArray. This is an enumeration of the following status strings:</p> <ul style="list-style-type: none"> <li><i>"Rebuilding"</i></li> <li><i>"Initializing"</i></li> <li><i>"Offline"</i></li> <li><i>"Online"</i></li> <li><i>"Error"</i></li> <li><i>"Degraded"</i></li> <li><i>"Critical"</i></li> </ul> <p><i>Rebuilding indicates a drive participating in an array is replaced, while the data striping and/or redundancy of the array are reestablished on the new disk. In this state, data operations to the array will function properly, but its performance could be significantly diminished.</i></p> <p><i>Initializing is the state when the RAID array is being first constructed, and the data striping and/or redundancy of the array are first being established. In this state, data operations to the array will function properly, but its performance could be significantly diminished and the redundancy might not be fully established.</i></p> <p><i>Offline indicates the RAID array is not available. This might occur because the administrator explicitly has disabled the array, or because underlying elements of the array, such as the physical disks, are not ready.</i></p> <p><i>Online indicates the RAID array is fully operational.</i></p> <p><i>Error indicates an error condition exists within the RAID array. In this state, data operations to the array are not possible, and data loss might have occurred.</i></p> <p><i>Degraded indicates the loss of a drive only in a Raid Type of more than 2 drives, that can support multiple failures. Data redundancy continues to be available but with degraded capability.</i></p> <p><i>Critical indicates the loss of data redundancy, and a possible degraded state. The array continues to perform read/write operations.</i></p> <p>The default "Offline" status will exist until this Storage Array is enabled.</p>	"Offline"
Enable	boolean	W	Enables or disables this StorageArray instance.	False

Name <sup>1</sup>	Type	Write <sup>2</sup>	Description	Object Default <sup>3</sup>
RaidType	string	W	<p>This value is one of the Enumerated<sup>4</sup> values found in the Capabilities parameter SupportedRaidTypes type:</p> <p>Note that after creation of the Storage Array, any subsequent writes to this parameter MUST be ignored as array type migration is not supported. To identify which RAID Types are supported, see the SupportedRaidTypes parameter in the Capabilities object.</p> <p>Once this instance becomes enabled, this parameter will be immutable for the life of the instance.</p>	"Linear"
UsableCapacity	unsignedInt	-	<p>The total Usable Capacity of the Storage Array in MB. This is computed by the system based on the PhysicalMediumReference and RaidType. Disk size is a consideration, as many RAID Types use the smallest drive in the group for calculations.</p>	-
PhysicalMediumReference	string(1024)	W	<p>A comma-separated list of Physical Medium references. Each Physical Medium referenced by this parameter MUST exist within the same StorageService instance. A Physical Medium MUST only be referenced by one Storage Array instance. Each reference can be either in the form of ".PhysicalMedium.{i}" or a fully qualified object name. For example:</p> <p>.PhysicalMedium.1,.PhysicalMedium.2</p> <p>This indicates that this Storage Array has 2 physical media.</p> <p>Once this instance becomes enabled, this parameter will be immutable for the life of the instance. This parameter acts as the unique identifier for the instance, thus the device MUST NOT allow multiple Storage Array instances to use the same PhysicalMediumReference.</p>	-

Name <sup>1</sup>	Type	Write <sup>2</sup>	Description	Object Default <sup>3</sup>
.StorageService.{i}.LogicalVolume.{i}.	object	W	<p>This object provides information about each logical volume configured on this device. A logical volume can reside either on an instance of a single PhysicalMedium or on an instance of a single StorageArray instance, but it can not span multiple instances of either. The PhysicalReference parameter is used to define where this LogicalVolume instance resides.</p> <p>Creating an instance of this object generates a disabled LogicalVolume instance. Before this new LogicalVolume instance can be enabled (via a SetParameterValues command), it MUST have the following parameters configured: Name, PhysicalReference, and Capacity. The unique<sup>5</sup> key for this object is the Name parameter, but once an instance is enabled the following parameters become immutable for the life of the instance: Name, PhysicalReference, and Capacity.</p>	-
Name	string(64)	W	<p>The unique name of the partition for this logical volume.</p> <p>Once this instance becomes enabled, this parameter will be immutable for the life of the instance. This parameter acts as the unique identifier for the instance, thus the device MUST NOT allow multiple Logical Volume instances to use the same Name.</p>	<Empty>
Status	string	-	<p>The current status of this Logical Volume. This is an enumeration of the following status strings:</p> <p><i>"Offline"</i>  <i>"Online"</i>  <i>"Error"</i></p> <p>The default "Offline" status will exist until this Logical Volume is enabled.</p>	"Offline"
Enable	boolean	W	Enables or disables this LogicalVolume instance.	False

Name <sup>1</sup>	Type	Write <sup>2</sup>	Description	Object Default <sup>3</sup>
PhysicalReference	string(256)	W	<p>This is a reference to a Physical Medium or a Storage Array that defines where this Logical Volume resides. The Physical Medium or Storage Array referenced by this parameter MUST exist within the same StorageService instance. Each reference can be either in the form of ".PhysicalMedium.{i}", ".StorageArray.{i}" or a fully qualified object name. For example:</p> <p>.PhysicalMedium.1</p> <p>This indicates that this Logical Volume resides on this StorageService enabled device within Physical Drive instance #1.</p> <p>Once this instance becomes enabled, this parameter will be immutable for the life of the instance.</p> <p>An AddObject followed by an enabling SetPropertyValues on the Name, PhysicalReference, and Capacity will cause the formatting of this logical partition and these parameters to become immutable for the life of this instance.</p>	-
FileSystem	string	-	The file system for this LogicalVolume as it is currently formatted. This parameter is one of the enumerated <sup>4</sup> values found in the Capabilities parameter SupportedFileSystemTypes.	-
Capacity	unsignedInt	W	<p>The Capacity of the Logical Volume in MB.</p> <p>Once this instance becomes enabled, this parameter will be immutable for the life of the instance.</p>	-
UsedSpace	unsignedInt	-	The Amount of Used Space on the Logical Volume in MB	-
ThresholdLimit	unsignedInt	W	<p>This value is specified in MB and controls when the ThresholdReached parameter will have its value altered. If the value of the UsedSpace parameter plus the value of this parameter is greater than or equal to the value of the Capacity parameter then the value of the ThresholdReached parameter will be True, otherwise it will be False. Setting the value of this parameter to 0 will disable the Thresholding mechanism.</p>	0
ThresholdReached	boolean	-	When ThresholdLimit > 0 and UsedSpace + ThresholdLimit >= Capacity this will be True, else False.	False
Encrypted	boolean	-	Is the Volume Encrypted? The type of encryption will be handled by the device internally and is not a matter for remote management.	-
FolderNumberOfEntries	unsignedInt	-	The number of instances of Folder on this LogicalVolume.	-

Name <sup>1</sup>	Type	Write <sup>2</sup>	Description	Object Default <sup>3</sup>
.StorageService.{i}.LogicalVolume.{j}.Folder.{i}	object	W	<p>This object provides information about each top-level folder configured on this logical volume. Each top-level folder allows the configuration of quotas and access permissions.</p> <p>Creating an instance of this object generates a disabled Folder instance. Before this new Folder instance can be enabled (via a SetParameterValues command), it MUST have the Name parameter configured. Folder instances are unique per Logical Volume and the unique<sup>5</sup> key for this object is the Name parameter, which also means that once an instance is enabled the Name parameter becomes immutable for the life of the instance.</p>	-
Name	string(64)	W	<p>Setting of this variable will add a new folder. The full hierarchical pathname of the folder MUST be specified.</p> <p>All folders created on the Logical Volume will appear here, created locally or via AddObject.</p> <p>The local directory name will be formatted as a UNIX-style directory name, for example:</p> <p>/home/ftp</p> <p>The presence of a trailing '/' character is inconsequential, meaning that /home/ftp/ is the same as /home/ftp.</p> <p>Once this instance becomes enabled, this parameter will be immutable for the life of the instance. This parameter acts as the unique identifier for the instance, thus the device MUST NOT allow multiple Folder instances within a LogicalVolume to use the same Name.</p>	<Empty>
Enable	boolean	W	Entry is enabled or disabled	False
UserAccountAccess	unsignedInt [0:3]	W	<p>What are the User authentication requirements of this folder? The following bitmap is used.</p> <p>11 = Authenticated Access required for Network protocols (NFS/AFP/SMB) and for non Network protocols (FTP/SFTP/HTTP/HTTPS)</p> <p>10 = Authenticated Access required for Network protocols (NFS/AFP/SMB)</p> <p>01 = Authenticated Access Required for non Network protocols (FTP/SFTP/HTTP/HTTPS)</p> <p>00 = No Authentication required</p>	2
UserAccessNumberOfEntries	unsignedInt	-	The number of instances of UserAccess on this Folder.	-
GroupAccessNumberOfEntries	unsignedInt	-	The number of instances of GroupAccess on this Folder.	-

Name <sup>1</sup>	Type	Write <sup>2</sup>	Description	Object Default <sup>3</sup>
.StorageService.{i}.LogicalVolume.{i}.Folder.{i}.UserAccess.{i}	object	W	This object provides information about each user account configured for access permissions on this folder. The unique <sup>5</sup> key for this object is UserReference	-
UserReference	string(256)	W	This is a reference to a User Account that has access to this Folder. The User Account referenced by this parameter MUST exist within the same StorageService instance. Each reference can be either in the form of ".UserAccount.{i}" or a fully qualified object name. For example: .StorageService.1.UserAccount.3 This indicates a user on this StorageService enabled device of UserAccount instance #3 has the designated access to this folder. This parameter is unique within the UserAccess table and this Folder instance, meaning that for each Folder instance the UserAccess table MUST NOT contain multiple UserAccess objects with the same UserReference value.	<Empty>
Permissions	unsignedInt [0:7]	W	What permissions the user account has over this Folder. This is an integer value like the xNIX systems use for access permissions, meaning that it is a 3-bit field with the following structure: 100 = read field 010 = write field 001 = execute field	7
.StorageService.{i}.LogicalVolume.{i}.Folder.{i}.GroupAccess.{i}	object	W	This object provides information about each user group configured for access permissions on this folder. The unique <sup>5</sup> key for this object is UserReference	-
GroupReference	string(256)	W	This is a reference to a User Group that is a collection of User Accounts that have access to this Folder. The User Group referenced by this parameter MUST exist within the same StorageService instance. Each reference can be either in the form of ".UserGroup.{i}" or a fully qualified object name. For example: .UserGroup.2 This indicates that a UserGroup on this StorageService enabled device with the instance #2 has the designated access to this folder. This parameter is unique within the GroupAccess table and this Folder instance, meaning that for each Folder instance the GroupAccess table MUST NOT contain multiple GroupAccess objects with the same GroupReference value.	<Empty>

Name <sup>1</sup>	Type	Write <sup>2</sup>	Description	Object Default <sup>3</sup>
Permissions	unsignedInt [0:7]	W	What permissions the group members have over this Folder. This is an integer value like the xNIX systems use for access permissions, meaning that it is a 3-bit field with the following structure:  100 = read field 010 = write field 001 = execute field	7
.StorageService.{i}.LogicalVolume.{i}.Folder.{j}.Quota.	object	W	This object allows the configuration of per-folder storage quota management.	-
Enable	boolean	W	Enables or disables quota management for this folder.	False
Capacity	unsignedInt	W	Maximum size in MB of the quota for this folder.	-
UsedSpace	unsignedInt	-	Current size in MB of this folder.	-
ThresholdLimit	unsignedInt	W	This value is specified in MB and controls when the ThresholdReached parameter will have its value altered. If the value of the UsedSpace parameter plus the value of this parameter is greater than or equal to the value of the Capacity parameter then the value of the ThresholdReached parameter will be True, otherwise it will be False. Setting the value of this parameter to 0 will disable the Thresholding mechanism.	0
ThresholdReached	boolean	-	When ThresholdLimit > 0 and UsedSpace + ThresholdLimit >= Capacity this will be True, else False.	False

## 5 Notification Requirements

CPE MUST support Active Notification (see [2]) for all parameters defined in the Parameter Definitions (section 4) with the exception of those parameters listed in Table 2. For only those parameters listed Table 2, the CPE MAY reject a request by an ACS to enable Active Notification via the SetParameterAttributes RPC by responding with fault code 9009 as defined in [2] (Notification request rejected).

CPE MUST support Passive Notification (see [2]) for all parameters defined in the Parameter Definitions (section 4), with no exceptions.

**Table 2 – Parameters for which Active Notification MAY be denied by the CPE**

Parameter <sup>6</sup>
.StorageService.{i}.PhysicalMedium.{i}.UpTime
.StorageService.{i}.LogicalVolume.{i}.UsedSpace
.StorageService.{i}.LogicalVolume.{i}.Folder.{i}.Quota.{i}.UsedSpace

## 6 Profile Definitions

### 6.1 Notation

The following abbreviations are used to specify profile requirements:

Abbreviation	Description
R	Read support is REQUIRED
W	Both Read and Write support is REQUIRED
P	The object is REQUIRED to be present
C	Creation and deletion of the object via AddObject and DeleteObject is REQUIRED

### 6.2 Baseline Profile

Table 3 defines the Baseline:1 profile for the StorageService:1 object. The minimum required version for this profile is StorageService:1.0.

**Table 3 – Baseline:1 profile definition for StorageService:1**

Name	Requirement
StorageService.{i}.Enable	P
StorageService.{i}.PhysicalMediumNumberOfEntries	W
StorageService.{i}.LogicalVolumeNumberOfEntries	R
StorageService.{i}.Capabilities	R
StorageService.{i}.FTPCapable	P
StorageService.{i}.FTPCapable	R

<sup>6</sup> The name of a Parameter referenced in this table is the concatenation of the object name shown in the yellow header, and the individual Parameter name.

Name	Requirement
SFTPCapable	R
HTTPCapable	R
HTTPSCapable	R
HTTPWritable	R
SupportedNetworkProtocols	R
SupportedFileSystemTypes	R
VolumeEncryptionCapable	R
StorageService.{i}.NetInfo.	P
HostName	W
DomainName	W
StorageService.{i}.NetworkServer.	P
AFPEnable	R
NFSEnable	R
SMBEnable	R
NetworkProtocolAuthReq	R
StorageService.{i}.PhysicalMedium.{i}.	P
Name	R
Vendor	R
Model	R
SerialNumber	R
FirmwareVersion	R
ConnectionType	R
Removable	R
Capacity	R
Status	R
SMARTCapable	R
HotSwappable	R
StorageService.{i}.LogicalVolume.{i}.	P
Name	R
Status	R
Enable	R
PhysicalReference	R
FileSystem	R
Capacity	R
UsedSpace	R
Encrypted	R
FolderNumberOfEntries	R
StorageService.{i}.LogicalVolume.{i}.Folder.{i}.	P
Name	R
Enable	R
UserAccountAccess	R

### 6.3 User Access Profile

Table 4 defines the UserAccess:1 profile for the StorageService:1 object. The minimum required version for this profile is StorageService:1.0.

**Table 4 – UserAccess:1 profile definition for StorageService:1**

Name	Requirement
StorageService.{i}.	P
UserAccountNumberOfEntries	R
StorageService.{i}.NetworkServer.	P
NetworkProtocolAuthReq	W
StorageService.{i}.UserAccount.{i}.	C
Enable	W
Username	W
Password	W
StorageService.{i}.LogicalVolume.{i}.Folder.{i}.	P
UserAccountAccess	W
UserAccessNumberOfEntries	R
StorageService.{i}.LogicalVolume.{i}.Folder.{i}.UserAccess.{i}.	C
UserReference	W
Permissions	W

#### 6.4 Group Access Profile

The GroupAccess:1 profile for the StorageService:1 object is defined as the union of the UserAccess:1 profile and the additional requirements defined in Table 5. The minimum required version for this profile is StorageService:1.0.

**Table 5 – GroupAccess:1 profile definition for StorageService:1**

Name	Requirement
StorageService.{i}.	P
UserGroupNumberOfEntries	R
StorageService.{i}.UserGroup.{i}.	C
Enable	W
GroupName	W
StorageService.{i}.UserAccount.{i}.	C
UserGroupParticipation	W
StorageService.{i}.LogicalVolume.{i}.Folder.{i}.GroupAccess.{i}.	C
GroupReference	W
Permissions	W

## 6.5 FTP Server Profile

Table 6 defines the FTPServer:1 profile for the StorageService:1 object. The minimum required version for this profile is StorageService:1.0.

**Table 6 – FTPServer:1 profile definition for StorageService:1**

Name	Requirement
StorageService.{i}.FTPServer.	P
Enable	W
Status	R
MaxNumUsers	W
IdleTime	W
PortNumber	W
StorageService.{i}.FTPServer.AnonymousUser.	P
Enable	W
StartingFolder	W
ReadOnlyAccess	W

## 6.6 SFTP Server Profile

The SFTPServer:1 profile for the StorageService:1 object is defined as the union of the UserAccess:1 and FTPServer:1 profile and the additional requirements defined in Table 7. The minimum required version for this profile is StorageService:1.0.

**Table 7 – SFTPServer:1 profile definition for StorageService:1**

Name	Requirement
StorageService.{i}.UserAccount.{i}.	P
AllowFTPAccess	W
StorageService.{i}.SFTPServer.	P
Enable	W
Status	R
MaxNumUsers	W
IdleTime	W
PortNumber	W

## 6.7 HTTP Server Profile

Table 8 defines the HTTPServer:1 profile for the StorageService:1 object. The minimum required version for this profile is StorageService 1.0.

**Table 8 - HTTPServer:1 profile definition for StorageService:1**

Name	Requirement
StorageService.{i}.HTTPServer.	P
Enable	W
Status	R
MaxNumUsers	W
IdleTime	W
HTTPWritingEnabled	R

Name	Requirement
PortNumber	W
AuthenticationReq	W

## 6.8 HTTPS Server Profile

The HTTPSServer:1 profile for the StorageService:1 object is defined as the union of the UserAccess:1 and HTTPSServer:1 profile and the additional requirements defined in Table 9. The minimum required version for this profile is StorageService:1.0.

**Table 9 - HTTPSServer:1 profile definition for StorageService:1**

Name	Requirement
StorageService.{i}.UserAccount.{i}.	P
AllowHTTPAccess	W
StorageService.{i}.HTTPSServer.	P
Enable	W
Status	R
MaxNumUsers	W
IdleTime	W
HTTPWritingEnabled	R
PortNumber	W
AuthenticationReq	W

## 6.9 Volume Config Profile

The VolumeConfig:1 profile for the StorageService:1 object is defined as the union of the BaselineProfile:1 and the additional requirements defined in Table 10. The minimum required version for this profile is StorageService:1.0.

**Table 10 – VolumeConfig:1 profile definition for StorageService:1**

Name	Requirement
StorageService.{i}.PhysicalMedium.{i}.	P
Name	W
StorageService.{i}.LogicalVolume.{i}.	C
Name	W
Enable	W
FileSystem	W
Capacity	W
StorageService.{i}.LogicalVolume.{i}.Folder.{i}.	C
Name	W
Enable	W
UserAccountAccess	W

## 6.10 RAID Profile

Table 11 defines the RAID:1 profile for the StorageService:1 object. The minimum required version for this profile is StorageService:1.0.

**Table 11 – RAID:1 profile definition for StorageService:1**

Name	Requirement
StorageService.{i}.	P
StorageArrayNumberOfEntries	R
StorageService.{i}.Capabilites.	P
SupportedRaidTypes	R
StorageService.{i}.StorageArray.{i}.	C
Name	W
Status	R
Enable	W
RaidType	W
UsableCapacity	R
PhysicalMediumReference	W

## 6.11 Folder Quota Profile

Table 12 defines the FolderQuota:1 profile for the StorageService:1 object. The minimum required version for this profile is StorageService:1.0.

**Table 12 - FolderQuota:1 profile definition for StorageService:1**

Name	Requirement
StorageService.{i}.LogicalVolume.{i}.Folder.{i}.Quota.	P
Enable	W
Capacity	W
UsedSpace	R
ThresholdLimit	W
ThresholdReached	R

## 6.12 Volume Threshold Profile

Table 13 defines the VolumeThresh:1 profile for the StorageService:1 object. The minimum required version for this profile is StorageService:1.0.

**Table 13 - VolumeThresh:1 profile definition for StorageService:1**

Name	Requirement
StorageService.{i}.LogicalVolume.{i}.	P
Capacity	R
ThresholdLimit	W
ThresholdReached	R

### 6.13 Network Server Profile

Table 14 defines the NetServer:1 profile for the StorageService:1 object. The minimum required version for this profile is StorageService:1.0.

**Table 14 - NetServer:1 profile definition for StorageService:1**

StorageService.{i}.NetworkServer.	P
AFPEnable	W
NFSEnable	W
SMBEnable	W

## 7 Use Cases

Below are some use case scenarios; user experience is a critical driver when introducing a new feature or device in the home network. This is an important reason to allow Service Providers a way to access complex devices by preventing trouble calls (auto-configuration) and resolving trouble calls in a timely fashion by directly accessing the device(s).

### 7.1 Basic Managed Storage Service

A basic managed storage service offers a Service Provider the option to assist the customer as soon as the customer's StorageService-enabled device is activated and being managed by the ACS. The following is a sample list of support capabilities an ACS can provide using CWMP (NOTE: Not all of these capabilities are handled with this data model; some are handled from a protocol perspective and some are handled via other data models):

- Basic configuration and setup during device activation [addressed by this document (configuration parameters) & TR-106 (configuration parameters)]
- User credentials setup and file privilege access [addressed by this document (Folder Access)]
- Firmware upgrade [addressed by TR-069 (Download command)]
- Retrieval of device status [addressed by this document (parameters) and TR-106]
- Wireless setup (e.g., WEP security) for a Storage Service device with Wi-Fi access [addressed by a future version of TR-106]
- Configuration and log file retrieval for root cause analysis of problems [addressed by a future version of TR-106]
- Monitoring active/passive notification events, e.g., volume capacity reached, and potential physical media failures [addressed by this document (parameters) and TR-069 (notification mechanism)]
- Network diagnostics and troubleshooting, e.g., network connectivity to the Internet gateway device, and to the Internet [addressed by TR-106 (connection parameters)]

With such support capabilities from the ACS, a technical support agent will be better equipped to help a customer with Storage Service issues during trouble calls. This is in addition to the ACS performing basic configuration/setup tasks during device

registration/activation. Overall, the goal is to provide a true “*plug-and-play*” solution for the customers.

## **7.2 Remote Storage Back-up Service**

A service provider can offer a file back-up service for selected volumes or folders on a Storage Service device. A Storage Service device allows the ability to upload files residing locally, initiated by a centralized network storage server via standard file transfer mechanisms such as SFTP and HTTPS. The file back-up operation can be initiated automatically on a regular basis using a time schedule selected by the customer or based on a default schedule established during service activation.

This is a service option that customers can use to back-up their “*precious content*” remotely on a network server in case of potential device disk failures. Since a majority of Storage Service devices currently do not have RAID support for disk redundancy, the remote back-up service can be a viable alternative for a certain segment of customers.

## **7.3 Remote access of Storage Service Content from a Remote Location**

This service provides a simple and secure solution for a customer to remotely access (over the internet) content on a Storage Service device installed in his/her home network. Access implies that the customer will be able to retrieve or load files to the Storage Service device. Since the Storage Service device is typically sitting behind a NAT-enabled Internet Gateway Device, this service would need to consider how the customer could access it through the NAT-enabled device.

## **Annex A: Theory of Operations**

The addition of the Storage Service device provides a data model for devices that have storage capabilities and are used for purposes such as file backup services, media server, and private storage for users or devices. These devices are usually connected within the home network, addressable behind the Gateway. These devices are typically accessible on the home network similarly to a PC through file sharing protocols such as Windows Networking (NetBIOS) and AppleTalk. The device providing the storage service is not usually a PC (although a PC is not excluded from providing the Storage Service) and thus the only interface to their functionality (file access) and setup (physical and logical drive formatting, wireless setup, DHCP server & client configuration, security provided by user access accounts) is through a Web based GUI, possibly UPnP, and (with TR-140) via TR-069.

### ***A.1 Physical Storage Theory of Operations***

Physical media are attached to a Storage Service through various methods. Some allow for hot-swapping and some are classified as removable media. The Storage Service device is responsible for discovery of all physical media attached to the device, and the removal of physical media that are no longer attached. The parameter `PhysicalMedium.{i}.Removable` indicates that this physical medium can be removed during runtime operations, and the parameter `PhysicalMedium.{i}.HotSwappable` indicates that this physical medium can be swapped with another drive during runtime operations.

If a Physical Medium is implemented to be removable from the Storage Service device, some additional operational factors need to be considered.

Most external storage devices SHOULD be treated as removable, for example: USB hard-drives, FireWire hard-drives, eSATA hard-drives, USB memory drives, portable media players, PDAs, memory cards, etc.

Whether a Physical Medium is removable MUST be indicated by the `StorageService-{i}.PhysicalMedium.{i}.Removable` parameter.

A hot-swappable Physical Medium is not considered removable storage in this discussion; removable storage implies that the removal action is part of normal operation and is expected to occur leaving the data on the removed storage intact; whereas, hot-swap capability is intended for removal in the case of disk error or failure.

It is expected that removable storage will typically be used for data transfer to and from the Storage Service, and not permanently participate in the Storage Service (although it is allowed to do so). It is generally bad practice to have removable storage participate in a Storage Array, as removal will cause degradation or failure of the Storage Array

### **A.1.1 Physical Medium Discovery**

The Storage Service device **MUST** discover any changes to the Physical Media in the system, including the addition or removal of a removable storage device during runtime.

The ACS can set up active notification on `PhysicalMediumNumberOfEntries` if it wishes to track addition or removal of a storage device.

Since a storage device could be removed and replaced with a different storage device, the ACS could check vendor/model/serial number before configuration of each removable Physical Medium to determine whether it is the same device or a different one from the last time it was configured.

### **A.1.2 Logical Volume Discovery**

There is no requirement for a Storage Service device to recognize specific file systems that might have been externally formatted on a Physical Medium; however, the Storage Service device will report any recognized Logical Volume on the new Physical Medium.

### **A.1.3 Folder Discovery**

The Storage Service device will report any recognized top-level Folder on the storage device.

### **A.1.4 Access Permissions**

User Account and User Group access permissions **MUST** be re-configured each and every time a removable storage device is (re-)added to the system. Previous access permissions are not expected to be restored automatically. Access permissions for all newly discovered folders are `UserAccountAccess = 0` (no Authentication required for Network Access or Remote Access Protocols).

### **A.1.5 Data Operations**

Like any other Physical Media in the system, the Storage Service device does not define any data operations involving removable storage. Operations like backup to/from the removable storage would be up to other services, applications, or the user.

Also, the Storage Services does not define any methods for ensuring safe removal as required by some removable storage devices; it is left to the user to perform any steps necessary to ensure safe removal.

## ***A.2 File Structure Management Theory of Operations***

Devices capable of supporting a Storage Service device can have one or more physical media and or storage arrays formatted with a hierarchical file system. Many physical media and storage arrays support partitioning of the usable capacity into smaller units,

which represent logical partitions or logical volumes, each of which is represented by an instance of the LogicalVolume object. The hierarchical file system can be structured further by creating and removing folders. These folders are modeled as instances of the Folder object residing under the LogicalVolume instance.

### **A.2.1 File Structure Management**

A Storage Service provides one or more Folders that can be configured by the administrator to allow remote file access by specified User Accounts and/or User Groups.

It is likely that the Folders exist within a hierarchical directory structure. However the Storage Service device only defines and administers one level (i.e. top-level) of Folder. Users can further create, administer and delete subfolders and subfolder hierarchies (according to the user permissions on the Folder).

If the administrator wants to manage user-created subfolder hierarchies within a Folder, the administrator cannot use the Storage Service mechanisms; rather the administrator would use a file manager facility (such as Windows Explorer) to access the Folder and its contents. This is possible because the administrator will have remote file access to all Folders created by the Storage Service.

### **A.2.2 FTP Root File Directory**

Storage Services devices that provide FTP protocol for remote file access will have special folders related to the FTP server.

FTP servers have a concept of a root directory. In the Storage Service, this is administered by the StorageService.{i}.FTPServer.AnonymousUser.StartingFolder. The FTPServer.AnonymousUser cannot be enabled until StorageService.{i}.FTPServer.AnonymousUser.StartingFolder references an existing Folder object. This reference overrides any existing security on the referenced Folder for the Anonymous user only.

### ***A.3 Access and Security Theory of operation***

Access to the Storage Service can be via Network Protocols (typically found on the LAN) and Remote Access Protocols (typically used for access over the WAN). Each supported protocol can be enabled or disabled entirely from access to the Storage Service.

#### **A.3.1 Security through UserAccess and GroupAccess profiles**

Security for device access and User access is optionally provided through the UserAccess and GroupAccess profiles. Both utilize the same username and password on a per user basis. The username and password are stored in the UserAccount object, which allows multiple users to be defined on the system. With GroupAccess profile support each UserAccount can be associated with a UserGroup, which allows multiple Users the same folder permissions within a UserGroup. When a user logs in, they might be required to enter a username and password. This entry MUST match at least one of the UserAccount objects. Once the user has passed the authentication, they will be able to manipulate files within the folders that they have permissions to do so. Permissions to view and manipulate files and folders are assigned on a per folder basis (Folder.{i}.Name), each folder containing a list of Users and Groups with permission to access them in Folder.{i}.UserAccess.{i} and Folder.{i}.GroupAccess.{i}.

With this type of access control, users logging in via a Network protocol such as HTTP or FTP will be compared to the same set of User Account objects for authentication (when required) and authorization purposes, in essence using the same set of credentials for all access methods.

#### **A.3.2 Levels of Security**

With this Model there are two levels of access. One is access to the device with a Storage Service, and the other is to the actual folders (and files) on the device.

##### **Device Access**

In a Storage Service, users can connect through a variety of protocols, including but not limited to Remote Access protocols such as HTTP/FTP, and Network protocols such as NFS/SMB/AFP.

When the user connects via an enabled Network/Remote Access protocol server to the Storage Service with no User Access or Group Access profile support, no authentication is required.

If User Access or Group Access profile is supported and the user connects via an enabled Network protocol, access is granted if any of the following are true:

- The account is valid and enabled
- The account is non-existent and .StorageService.{i}.NetworkServer.NetworkProtocolAuthReq = False

If User Access or Group Access profile is supported and the user connects via an enabled Remote Access protocol access is only granted:

- If the account is valid<sup>7</sup> AND the protocol is allowed for this user (AllowHTTPAccess or AllowFTPAccess).

### Folder Access

After the User is granted device access, the user can access folders they are authorized to access. Permissions to view and manipulate files and folders are assigned on a per folder basis (Folder.{i}.Name), each folder containing a list of Users and Groups with permission to access them in Folder.{i}.UserAccess.{i} and Folder.{i}.GroupAccess.{i}. Folder authorization can occur in one or more of the following ways:

1. The folder sets UserAccountAccess to '0' allowing any user from the Network protocols and Remote Access protocols to access this file.
2. The folder sets UserAccountAccess to '3<sup>8</sup>' requiring a user from either the Network protocols or Remote Access protocols to be present and enabled in the Folders.{i}.UserAccess table to access this file.
3. The folder sets UserAccountAccess to '3<sup>9</sup>' requiring a user from either the Network protocols or Remote Access protocols to be present and enabled in the Folders.{i}.GroupAccess table that the UserAccount is a member of through the UserGroupParticipation parameter, to access this file.
4. FTP Anonymous is handled outside of this mechanism and applies to only one top-level folder.

Note : Each UserAccount SHOULD be granted access to at least one account

Figure #1 and Access Table #1 are an example depicting the possible references set by TR-140 and the decision process of the device to determine if the user has access to a folder.

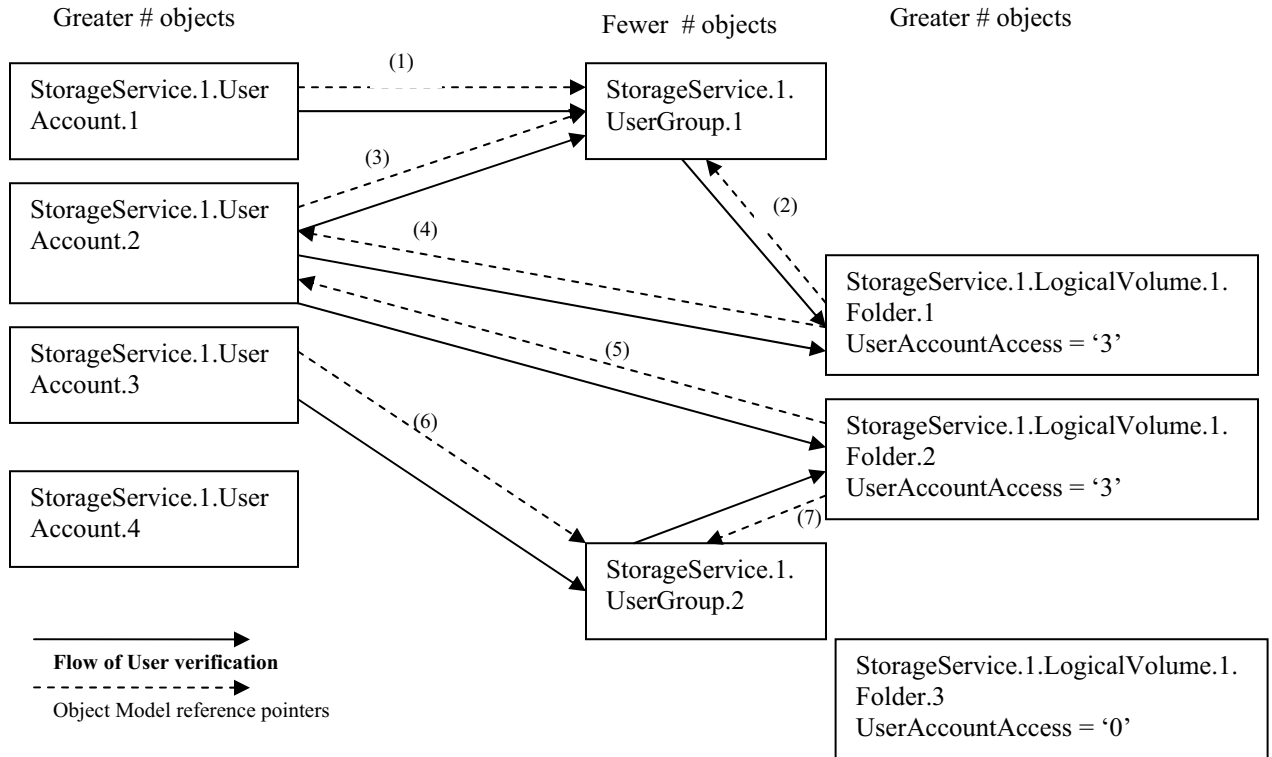
---

<sup>7</sup> If using HTTP the AuthenticationReq = False, the user will not have a valid account

<sup>8</sup> This can be set to '1' requiring only the non Network protocols user to be present in the Folders UserAccess or Group Access tables. This can also be set to '2' requiring only the Network protocols user to be present in the Folders UserAccess or Group Access tables.

<sup>9</sup> This can be set to '1' requiring only the non Network protocols user to be present in the Folders UserAccess or Group Access tables. This can also be set to '2' requiring only the Network protocols user to be present in the Folders UserAccess or Group Access tables.

**User Access or Group Access profile support**



**Figure #1**  
How

Folder Access	How
StorageService.UserAccount.1 StorageService.1.LogicalVolume.1.Folder.1	(2)StorageService.1.LogicalVolume.1.Folder.1.GroupAccess.1.UserReference = StorageService.1.UserGroup.1
StorageService.1.LogicalVolume.1.Folder.3	(1)StorageService.1.UserAccount.1.UserGroupParticipation [includes] StorageService.1.UserGroup.1 No Authentication Required.
StorageService.UserAccount.2 StorageService.1.LogicalVolume.1.Folder.1	(4)StorageService.1.LogicalVolume.1.Folder.1.UserAccess.1.UserReference = StorageService.1.UserAccount.2
StorageService.1.LogicalVolume.1.Folder.1	(2)StorageService.1.LogicalVolume.1.Folder.1.GroupAccess.1.UserReference = StorageService.1.UserGroup.1
StorageService.1.LogicalVolume.1.Folder.2	(3)StorageService.1.UserAccount.2.UserGroupParticipation [includes] StorageService.1.UserGroup.1
StorageService.1.LogicalVolume.1.Folder.3	(5)StorageService.1.LogicalVolume.1.Folder.2.UserAccess.1.UserReference = StorageService.1.UserAccount.2 No Authentication Required.
StorageService.UserAccount.3 StorageService.1.LogicalVolume.1.Folder.2	(7)StorageService.1.LogicalVolume.1.Folder.2.GroupAccess.1.UserReference = StorageService.1.UserGroup.2
StorageService.1.LogicalVolume.1.Folder.3	(6)StorageService.1.UserAccount.3.UserGroupParticipation [includes] StorageService.1.UserGroup.2 No Authentication Required.
StorageService.UserAccount.4 StorageService.1.LogicalVolume.1.Folder.3	No Authentication Required.
No Account (Network protocol access only if .StorageService.{i}.Access.NetworkProtocolAuthReq <sup>10</sup> = False. StorageService.1.LogicalVolume.1.Folder.3	No Authentication Required

**Access Table #1**

### A.3.3 Recommended Object Creation Pattern:

- UserGroup – Only if prefer to add future UserAccounts
- UserAccount – Can only reference UserGroups that are defined.
- Folder – done after initial Users and/or Groups are added,

As a benefit of this design, If UserGroups are used, adding UserAccounts going forward only require a single Add User.

## A.4 Data Model Integrity Theory of Operations

### A.4.1 Physical Medium

An ACS is not allowed to add or remove an instance of PhysicalMedium as it is not a writable object.

When the device manages the PhysicalMedium instances, it needs to maintain the following data integrity considerations:

**When a PhysicalMedium instance is removed, all references to it MUST be removed.**

- If a Physical Medium is removed, the device needs to remove any references to it by StorageService.{i}.StorageArray.{i}.PhysicalMediumReference, and depending on the RAID type and number of disks in the RAID this could affect the status of the StorageArray instance.
- If a Physical Medium is removed, the device needs to remove any references to it by StorageService.{i}.LogicalVolume.{i}.PhysicalReference

### A.4.2 Storage Array

A StorageService can have multiple StorageArray instances, each modeling its own RAID configuration. An ACS can create or remove StorageArray instances by utilizing the AddObject and DeleteObject CWMP RPC methods respectively. To create a new storage array an AddObject will be followed by a SetParameterValues RPC invocation. This is required as the StorageArray object will be created in a disabled state. The device supporting the StorageService will not create another new instance or allow the enabling of the storage array until it is provided a unique name, RAID type, required minimum number of physical medium references, and enabled via the SetParameterValues RPC method. To delete a storage array a DeleteObject RPC invocation for the correct StorageArray instance is required. A disabled StorageArray instance is not removed until a DeleteObject RPC invocation is carried out for it. Once the StorageArray instance is enabled, the Name and RAIDType parameters are immutable while the StorageArray object exists.

When managing the StorageArray instances the device needs to maintain the following data integrity considerations:

**When an object reference parameter is being SET, the object being referenced MUST exist**

- The StorageArray.{i}.PhysicalMediumReference parameter needs to refer to instances of PhysicalMedium that currently exist in the data model. A PhysicalMedium MUST NOT be referenced by multiple instances of StorageArray.

**Referenced objects that are removed.**

- If a PhysicalMedium is removed, the status of the StorageArray might be affected.

**When a StorageArray instance is removed, all references to it MUST be removed.**

- If a Storage Array is removed, the device needs to remove any references to it by StorageService.{i}.LogicalVolume.{i}.PhysicalReference

**A.4.3 Logical Volume**

A StorageService can have multiple LogicalVolume instances each modeling a logical partition on a PhysicalMedium or StorageArray. An ACS can create or remove LogicalVolume instances by utilizing the AddObject and DeleteObject CWMP RPC methods respectively. To create a new logical partition an AddObject will be followed by a SetParameterValues RPC invocation. This is required as the LogicalVolume object will be created in a disabled state. The device supporting the StorageService will not create or allow the enabling of the logical partition until it is provided a valid PhysicalReference, Name, Capacity, and enabled via the SetParameterValues RPC method. To delete a logical partition a DeleteObject RPC invocation for the correct LogicalVolume instance is required. A disabled LogicalVolume instance is not removed until a DeleteObject RPC invocation is carried out for it. Once the LogicalVolume is enabled, the PhysicalReference, Name and Capacity parameters are immutable while the LogicalVolume instance exists.

**When an object reference parameter is being SET, the object being referenced MUST exist**

- The LogicalVolume.{i}.PhysicalReference parameter needs to refer to an existing instance of either PhysicalMedium or StorageArray.

**Referenced objects that are disabled**

- If the object being referenced by the parameter PhysicalReference is disabled then the LogicalVolume will be disabled.

**Referenced objects that are removed.**

- If the object being referenced by the parameter PhysicalReference is removed then the LogicalVolume will be removed.

**A.4.4 Folder**

A StorageService can have multiple Folder instances each modeling a top-level folder on a logical partition. An ACS can create or remove Folder instances by utilizing the AddObject and DeleteObject CWMP RPC methods respectively. To create a new folder on the file system an AddObject will be followed by a SetParameterValues RPC invocation. This is required as the Folder object will be created in a disabled state and the device supporting the StorageService will not create the folder on the file system until

it is provided with a locally unique name and enabled via the SetParameterValues RPC method. To delete a folder from the file system a DeleteObject RPC invocation for the correct Folder instance is required. A disabled Folder object is not removed from the file system until a DeleteObject RPC invocation is carried out for it. Once the Folder is enabled, the Name is immutable while the Folder instance exists.

**When an object reference parameter is being SET, the object being referenced MUST exist**

- If a Folder references a UserGroup or a UserAccount, it MUST already exist.

**When a Folder instance is removed, all references to it MUST be removed**

- If the Folder being referenced by the parameter StorageService.{i}.FTPServer.AnonymousUser.StartingFolder is deleted, then the anonymous access to the FTPServer is disabled by the device and the parameter will be set to empty.

**When a Folder instance is disabled, all references to it MUST be disabled**

- If the Folder being referenced by the parameter StorageService.{i}.FTPServer.AnonymousUser.StartingFolder is disabled, then the anonymous access to the FTPServer is disabled by the device.

**Referenced objects that are Disabled**

- If a UserAccount or UserGroup is disabled, the device MUST NOT remove references to it by all Folders. The folders MUST verify the account is enabled prior to granting access.

**Referenced objects that are removed**

- If a UserAccount or UserGroup references is removed, those associated accounts no longer have access to the folder.

#### **A.4.5 UserAccount and UserGroup**

A StorageService can have multiple UserAccount and UserGroup instances. An ACS can create or remove these instances by utilizing the AddObject and DeleteObject CWMP RPC methods respectively. To create a new UserAccount or UserGroup an AddObject will be followed by a SetParameterValues RPC invocation. This is required as the UserAccount or UserGroup object will be created in a disabled state and the device supporting the StorageService will not create the UserAccount or UserGroup until it is provided with a locally unique name and enabled via the SetParameterValues RPC method. To delete a UserAccount or UserGroup a DeleteObject RPC invocation for the correct instance is required. A disabled UserAccount or UserGroup object is not removed from the file system until a DeleteObject RPC invocation is carried out for it. Once the UserAccount or UserGroup is enabled, the Name is immutable while the instance exists.

**When an object reference parameter is being SET, the object being referenced MUST exist**

- If a UserAccount references a UserGroup, it MUST already exist.

**When a UserAccount or UserGroup object is removed, the references to it MUST be removed.**

- If a UserGroup is deleted, the device MUST remove references to it by all UserAccounts and Folders.
- If a UserAccount is deleted, the device MUST remove references to it by all Folders.

### ***A.5 File Retrieval Theory of Operations***

Devices capable of supporting a Storage Service are intended to save files to internal drives, which prompts the question of how to actually get the file onto the device. There are several entities that might want to place a file onto a NAS device, such as a Service Provider, a third-party Content Provider, a network-local STB, or even the owner of the NAS Device. In a typical deployment only the Service Provider would have remote CWMP access to the NAS device, allowing it to use the built-in CWMP Download and Upload commands to actually push files to the device or retrieve files from the device. Third-party Content Providers will need to utilize the remote access capabilities that this device provides; whether that is FTP or HTTP depends on the device and can be identified via the Capabilities object. A network-local STB will need to utilize a LAN based access mechanism like UPnP (in the case that the device is a UPnP Media Server). A network-local STB could also use the device's embedded FTP or HTTP servers, depending on the capabilities of the device. The owner of the device will be able to use either LAN access if inside the LAN or FTP/HTTP if outside the LAN depending on the capabilities of the device.

### ***A.6 Thresholding Theory of Operations***

Within a Logical Volume there is a limit to the amount of space configured for storage. In storage applications it is useful to provide a warning when the amount of available storage space is approaching zero. The mechanism provided to the ACS here is in the form of a notification (active or passive) set on the ThresholdReached parameter. The implementation of this parameter requires the Storage Service to monitor the UsedSpace within the Logical Volume as it relates to the ThresholdLimit. When  $UsedSpace + ThresholdLimit < Capacity$ , ThresholdReached is false. When  $UsedSpace + ThresholdLimit \geq Capacity$ , ThresholdReached is True. Note that if a user is actively adding and deleting files when the  $UsedSpace + ThresholdLimit$  is near to Capacity the ThresholdReached parameter might toggle back and forth, causing excessive CWMP traffic in the event that active notification is set for the ThresholdReached parameter.

## **Annex B: RAID Type Descriptions**

### ***B.1 Basic RAID Levels***

#### **B.1.1 Linear RAID**

Linear RAID is a simple grouping of drives to create a larger virtual drive. In linear RAID, the chunks are allocated sequentially from one member drive, going to the next drive only when the first is completely filled. This grouping provides no performance benefit, as it is unlikely that any I/O operations will be split between member drives. Linear RAID also offers no redundancy, and in fact decreases reliability – if any one drive fails, the entire array cannot be used. The capacity is the total of all member disks.

For Linear RAID Level, the minimum number of disks required is 1.

#### **B.1.2 Level 0**

RAID level 0, often called “striping”, is a performance-oriented striped data mapping technique. That means the data being written to the array is broken down into strips and written across the member disks of the array. This allows high I/O performance at low inherent cost but provides no redundancy. Storage capacity of the array is equal to the total capacity of the member disks.

For RAID Level 0, the minimum number of disks required is 2.

#### **B.1.3 Level 1**

RAID level 1, or “mirroring”, provides redundancy by writing identical data to each member disk of the array, leaving a “mirrored” copy on each disk. It has been used longer than any other form of RAID. Level 1 Mirroring remains popular due to its simplicity and high level of data availability. Level 1 operates with two or more disks that can use parallel access for high data-transfer rates when reading, but more commonly operate independently to provide high I/O transaction rates. Level 1 provides very good data reliability and improves performance for read-intensive applications but at a relatively high cost. Array capacity is equal to the capacity of one member disk.

For RAID Level 1, the minimum number of disks required is 2.

#### **B.1.4 Level 2**

RAID level 2 uses bit-level striping with parity. Data is split at a bit-level and spread over a number of data disks and redundancy disks. The redundant bits are calculated using Hamming codes (which is a form of ECC). On a write, the codes are calculated and written to dedicated redundancy disks alongside the data, which is written to the data disks. On a read, the data and ECC codes are read and the latter is used to verify the

consistency of the former. The ECC codes can correct single-bit errors “on the fly”. The total usable capacity varies by implementation. Due to the overhead of calculating and storing codes, RAID 2 is not so popular today.

For RAID Level 2, the minimum number of disks required is 3.

### **B.1.5 Level 3**

Level 3 uses byte-level striping with dedicated parity. Striping is done at a byte level with the same number of bytes in each stripe. The parity information is stored on a dedicated disk. The latter can be a source of bottlenecks especially during random writes. Further, since every write is striped across all the disks, the array could be writing 1 block of data at a time. Thus RAID3 performance depends on the nature of the writes: it performs well on large sequential writes but poorly on small random writes. Array capacity is equal to the capacity of the member disks, minus the capacity of one member disk.

For RAID Level 3, the minimum number of disks required is 3.

### **B.1.6 Level 4**

Level 4 uses block-level striping with parity concentrated on a single disk drive to protect data. It's better suited to transaction I/O rather than large file transfers. Because the dedicated parity disk represents an inherent bottleneck, level 4 is seldom used without accompanying technologies such as write-back caching. RAID 4 differs from RAID 3 only in the size of the stripes sent to the various disks. Array capacity is equal to the capacity of the member disks, minus the capacity of one member disk.

For RAID Level 4, the minimum number of disks required is 3.

### **B.1.7 Level 5**

This is the most common type of RAID. By distributing parity across some or all of an array's member disk drives, RAID level 5 eliminates the write bottleneck inherent in level 4. The only bottleneck is the parity calculation process. With modern CPUs and software-based RAID, that isn't a very big bottleneck. As with level 4, the result is asymmetrical performance, with reads substantially outperforming writes. Level 5 is often used with write-back caching to reduce the asymmetry. Array capacity is equal to the capacity of the member disks, minus the capacity of one member disk.

For RAID Level 5, the minimum number of disks required is 3.

### **B.1.8 Level 6**

RAID 6 uses block-level striping with dual distributed parity. RAID 6 can be thought as RAID 5, but instead calculates 2 sets of parities on the data. Just as in RAID 5, the

parities are distributed across some or all of an array's member disk drives. Dual-distributed parity helps to improve fault tolerance, by making the subsystem tolerant to 2 disk failures. In terms of performance, RAID 6 is slightly worse than RAID 5 due to the usage of dual parity. Array capacity is equal to the capacity of the member disks, minus the capacity of two member disks.

For RAID Level 6, the minimum number of disks required is 4.

## ***B.2 Combination RAID Levels***

In addition to the RAID levels described above, arrays can implement a combination of the above RAID levels, thus leveraging the benefits of multiple RAID levels. The next section describes some of the useful combinations of RAID levels.

### **B.2.1 Level 10**

RAID 10 is a combination of RAID 1 (mirroring) followed by RAID 0 (striping), in other words it is a "stripe of mirrors". For RAID 10, a bunch of disks is grouped in pairs, with mirroring within pairs, and then data is striped across these pairs. In terms of failures, the array can tolerate  $n/2$  disk failures across the group, as long as no 2 disks within the same "mirrored set" fail. Array capacity is equal to the capacity of the member disks divided by 2.

For RAID Level 10, the minimum number of disks required is 3.

### **B.2.2 Level 01**

RAID level 01 is combination of RAID 0 (striping) followed by RAID 1 (mirroring), in other words it is a "mirror of stripes". For an array with  $n$  disks using RAID 01, data is striped across  $n/2$  disks and the remaining  $n/2$  disks are used as mirrors. In terms of fault tolerance, the array can continue functioning as long as both mirrors across the stripe-sets don't go down. Array capacity is equal to the capacity of the member disks divided by 2.

For RAID Level 01, the minimum number of disks required is 3.

### **B.2.3 Level 30**

Level 30 is formed by striping across a number of RAID 3 sub-arrays. In general, RAID 30 leverages the performance benefit of RAID 0 due to striping, and the fault tolerance benefits of RAID 3. Array capacity is equal to the capacity of the member disks, minus the capacity of one member disk into the number of RAID 3 sets (used for striping). The number of disks MUST be a multiple of 2 integers (one of them 2 or higher and the other 3 or higher).

For RAID Level 30, the minimum number of disks required is 6.

**B.2.4 Level 50**

RAID 50 uses block striping with distributed parity (RAID 5) along with block striping (RAID 0). In other words, data is striped across a number of RAID 5 sub-arrays. Just like RAID 30, RAID 50 leverages the performance benefit of RAID 0 due to striping, and the fault tolerance benefits of RAID 5. Array capacity is equal to the capacity of the member disks, minus the capacity of one member disk into the number of RAID 5 sets (used for striping).

For RAID Level 50, the minimum number of disks required is 6.

**B.2.5 Level 60**

RAID 60 combines stripes (RAID 0) across RAID 6 sets. The dual parity of RAID 6 allows for failure of up to two disks. Striping across RAID 6 sets provides the performance benefits of RAID 0. Array capacity is equal to the capacity of the member disks, minus the capacity of one member disk into the number of RAID 6 sets (used for striping).

For RAID Level 60, the minimum number of disks required is 8.