

SANscreen **Connect API 5.1.3** **Reference Manual**

NetApp, Inc.
495 East Java Drive
Sunnyvale, CA 94089 U.S.A.
Telephone: +1 (408) 822-6000
Fax: +1 (408) 822-4501
Support telephone: +1 (888) 4-NETAPP
Documentation comments: doccomments@netapp.com
Information Web: <http://www.netapp.com>

Part number 215-05011_A0
October 2009

Copyright and trademark information

Copyright information

Copyright © 1994-2009 NetApp, Inc. All rights reserved. Printed in the U.S.A.

The third-party copyright statements and licenses that apply to open source software components distributed with the product are contained in the section “Third-Party Copyrights and Licenses.”

No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S.A. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

Trademark information

NetApp, the Network Appliance logo, the bolt design, NetApp—the Network Appliance Company, Cryptainer, Cryptoshred, DataFabric, DataFort, Data ONTAP, Decru, FAServer, FilerView, FlexClone, FlexVol, Manage ONTAP, MultiStore, NearStore, NetCache, NOW NetApp on the Web, SANscreen, SecureShare, SnapDrive, SnapLock, SnapManager, SnapMirror, SnapMover, SnapRestore, SnapValidator, SnapVault, Spinnaker Networks, SpinCluster, SpinFS, SpinHA, SpinMove, SpinServer, StoreVault, SyncMirror, Topio, VFM, VFM (Virtual File Manager), and WAFL are registered trademarks of NetApp, Inc. in the U.S.A. and/or other countries. gFiler, Network Appliance, SnapCopy, Snapshot, and The evolution of storage are trademarks of NetApp, Inc. in the U.S.A. and/or other countries and registered trademarks in some other countries. The NetApp arch logo; the StoreVault logo; ApplianceWatch; BareMetal; Camera-to-Viewer; ComplianceClock; ComplianceJournal; ContentDirector; ContentFabric; EdgeFiler; FlexShare; FPolicy; Go Further, Faster; HyperSAN; InfoFabric; Lifetime Key Management, LockVault; NOW; ONTAPI; OpenKey, RAID-DP; ReplicatorX; RoboCache; RoboFiler; SecureAdmin; Serving Data by

Design; SharedStorage; Simplicore; Simulate ONTAP; Smart SAN; SnapCache; SnapDirector; SnapFilter; SnapMigrator; SnapSuite; SohoFiler; SpinMirror; SpinRestore; SpinShot; SpinStor; vFiler; Virtual File Manager; VPolicy; and Web Filer are trademarks of NetApp, Inc. in the U.S.A. and other countries. NetApp Availability Assurance and NetApp ProTech Expert are service marks of NetApp, Inc. in the U.S.A.

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. A complete and current list of other IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>.

Apple is a registered trademark and QuickTime is a trademark of Apple, Inc. in the U.S.A. and/or other countries. Microsoft is a registered trademark and Windows Media is a trademark of Microsoft Corporation in the U.S.A. and/or other countries. RealAudio, RealNetworks, RealPlayer, RealSystem, RealText, and RealVideo are registered trademarks and RealMedia, RealProxy, and SureStream are trademarks of RealNetworks, Inc. in the U.S.A. and/or other countries.

All other brands or products are trademarks or registered trademarks of their respective holders and should be treated as such.

NetApp, Inc. is a licensee of the CompactFlash and CF Logo trademarks. NetApp, Inc. NetCache is certified RealSystem compatible.

NetApp SANscreen Products

The following products comprise the NetApp SANscreen suite of solutions.

Product Name	Description
Service Insight	Offers visibility into your SAN environment, including host-to-storage access paths, storage availability and change management.
Service Assurance	Enables real-time monitoring of storage services and risk-free migrations.
Capacity Manager	Capacity Manager manages the end-to-end resource order and allocation process to allow companies to accelerate application provisioning, manage resource reservations and forecast based on past trends and future demand, enabling educated capacity management decisions.
VM Insight	Enables virtual machine to array path visibility for organizations deploying VMware ESX technology combined with VMware Virtual Center.
Application Insight	Uses SAN traffic data to reclaim underutilized resources, manage tiers, identify multipath risks and troubleshoot ongoing performance bottlenecks. Discovers, collects and maintains performance information measured by critical array metrics throughout the SAN.

Documentation Set

The following table lists some of the documentation set for NetApp's suite of products. All manuals are available from the client Help menu.

User Guide	Description of Content
SANscreen Installation and Administration User Guide	Overview of the suite's architecture with instructions to get the system up and running, discover the SAN and perform administrative tasks.
SANscreen Service Insight User Guide	Information about the tools and features available to help you gain visibility into your SAN environment.
SANscreen Service Assurance User Guide	Overview of how to use Service Assurance to analyze and validate your storage network environment and to automate operational monitoring, as well as procedures to make safe changes without disrupting availability.
SANscreen VM Insight	Overview of Virtualization in SANscreen.
SANscreen Application Insight User Guide	Overview of how to use Application Insight to reclaim underutilized resources, manage tiers, identify multipath risks and troubleshoot on-going performance bottlenecks. Provides information about how the Array Performance Module for Application Insight discovers, collects and maintains performance information measured by critical array metrics throughout the SAN.
SANscreen Capacity Manager User Guide	Overview of how to use Capacity Manager to enable educated capacity management decisions by managing the end-to-end resource order and allocation process.
SANscreen Connect API User Guide	Overview of how to use the API interface to enable integration with other applications, such as reporting and monitoring systems.

Overview	1
What is SANscreen Connect?	1
Prerequisites	1
Main API Object Types	2
Use Cases	3
Integrate Storage Data Into Centralized IT Dashboard	4
Integrate Storage Monitoring Into Standard Help Desk Network Operations Center (NOC)	4
Chargeback Reports	4
Software Vendors Integrate the API Into Their Own Applications to Provide SAN Visibility	5
Frequently Asked Questions	6
How is the API licensed?	6
What are the main object types?	6
How is authentication performed?	6
How do I initiate a session?	6
How to Use the SANscreen Connect API	7
Prerequisites	8
Connecting to the SANscreen Connect API	8
Classes	9
APIBadRequestException	11
Constructors	11
Methods	11
Action	13
Constructors	13
Methods	13

getOwner	13
setOwner	13
getTaskId	13
setTaskId.	13
getType	13
setType	13
getActionOrder	13
setActionOrder	14
getState	14
setState.	14
getParameters	14
setParameters.	14
getErrors.	14
setErrors.	14
ActionResponse	16
Constructors	16
Methods.	16
getActions.	16
setActions.	16
AnnotationType.	17
Constructors	17
Methods.	17
getName.	17
setName	17
getDisplayName	17
setDisplay_name	17
getDescription.	17
setDescription.	17
getDataType	18
setDataType	18
getOwner	18
setOwner	18
getSupportedObjectTypes.	18

setSupportedObjectTypes	18
getEnumerationValues	18
setEnumerationValues	18
isUserDefined	18
setUserDefined	18
isAllowMultipleValues	18
setAllowMultipleValues	19
AnnotationValue	21
Constructors	21
Methods	21
getEnumId	21
setEnumId	21
getTypeId	21
setTypeId	21
getObjectReference	21
setObjectReference	21
getValue	21
setValue	22
getSetTime	22
setSetTime	22
APIException	23
Constructors	23
Methods	23
APISessionContext	24
Constructors	25
Methods	25
getId	25
Returns	25
getTime	25
Returns	25
getClientVersion	25
Returns	25

getLimit	25
Returns	26
toString	26
Overrides	26
Application	28
Constructors	28
Methods	28
getBusinessUnit	28
setBusinessUnit	28
isIgnoreSharing	28
setIgnoreSharing	28
getName	28
setName	28
getPriority	28
setPriority	29
getHostIds	29
setHostIds	29
ApplicationResponse	30
Constructors	30
Methods	30
getApplications	30
setApplications	30
DeviceGroup	31
Constructors	31
Methods	31
getName	31
setName	31
getType	31
setType	31
getStorageManagementId	31
setStorageManagementId	31
getStorageManagementName	31

setStorageManagementName	32
DeviceGroupResponse	33
Constructors	33
Methods	33
getDeviceGroups	33
setDeviceGroups	33
DRPath	34
Constructors	34
Methods	34
getHostId	34
setHostId	34
getStorageId	34
setStorageId	34
getVolumeId	34
setVolumeId	34
getReplicaVolumeId	34
setReplicaVolumeId	35
getRemoteReplicaVolumeId	35
setRemoteReplicaVolumeId	35
getSecondaryReplicaVolumeId	35
setSecondaryReplicaVolumeId	35
getRemoteHostId	35
setRemoteHostId	35
getRemoteStorageId	35
setRemoteStorageId	35
DRPathResponse	36
Constructors	36
Methods	36
getDrPaths	36
setDrPaths	36
FindObjectsResponse	37
Constructors	37

Methods.	37
getDevices	37
setDevices	37
Generic.	38
Constructors	38
Methods.	38
getId	38
Returns	38
setId	38
Parameters	38
getWwn.	38
Returns	38
setWwn.	39
Parameters	39
isDead	39
Returns	39
setDead	39
Parameters	39
GenericResponse	40
Constructors	40
Methods.	40
Returns	40
setRequestIterator	40
Parameters	40
getSize	40
Returns	40
setSize	41
Parameters	41
getGenerics	41
Returns	41
setGenerics	41
Parameters	41

Host	42
Constructors	42
Methods	42
getName	42
Returns	42
setName	42
Parameters	42
getIp	42
Returns	43
setIp	43
Parameters	43
isDead	43
Returns	43
setDead	43
Parameters	43
getId	43
Returns	43
setId	43
Parameters	44
HostResponse	45
Constructors	45
Methods	45
getRequestIterator	45
Returns	45
setRequestIterator	45
Parameters	45
getSize	45
Returns	46
setSize	46
Parameters	46
getHosts	46
Returns	46

setHosts	46
Parameters	46
LogicalPort	47
Constructors	47
Methods	47
getId	47
Returns	47
setId	47
Parameters	47
getState	47
Returns	47
setState	48
Parameters	48
getStatus	48
Returns	48
setStatus	48
Parameters	48
getWwn	48
Returns	48
setWwn	48
Parameters	48
getDeviceld	48
Returns	49
setDeviceld	49
Parameters	49
getNodeId	49
Returns	49
setNodeId	49
Parameters	49
getConnectedPortId	49
Returns	49
setConnectedPortId	49

Parameters	50
isDead	50
Returns	50
setDead	50
Parameters	50
getName	50
Returns	50
setName	50
Parameters	50
getPhysicalPortId	50
Returns	50
setPhysicalPortId	51
Parameters	51
LogicalPortResponse	52
Constructors	52
Methods.	52
getRequestIterator	52
Returns	52
setRequestIterator	52
Parameters	53
getSize	53
Returns	53
setSize	53
Parameters	53
getLogicalPorts	53
Returns	53
setLogicalPorts	53
Parameters	54
Node.	56
Constructors	56
Methods.	56
getId	56

Returns	56
setId	56
Parameters	56
getDeviceID	56
Returns	57
setDeviceID	57
Parameters	57
getWwn	57
Returns	57
setWwn	57
Parameters	57
isDead	57
Returns	57
setDead	57
Parameters	58
NodeResponse	59
Constructors	59
Methods	59
getRequestIterator	59
Returns	59
setRequestIterator	59
Parameters	59
getSize	59
Returns	60
setSize	60
Parameters	60
getNodes	60
Returns	60
setNodes	60
Parameters	60
Path	61
Constructors	61

Methods	61
getId	61
Returns	61
setId	61
Parameters	61
getArrayId	61
Returns	62
setArrayId	62
Parameters	62
getHostId	62
Returns	62
setHostId	62
Parameters	62
getVolumId	62
Returns	62
setVolumId	62
Parameters	63
PathResponse	64
Constructors	64
Methods	64
getRequestIterator	64
Returns	64
setRequestIterator	64
Parameters	64
getSize	64
Returns	65
setSize	65
Parameters	65
getPaths	65
Returns	65
setPaths	65
Parameters	65

Policy	66
Constructors	66
Methods	66
getHostId	66
setHostId	66
getArrayId	66
setArrayId	66
getVolumeId	66
setVolumeId	66
getRedundancy	66
setRedundancy	67
getHopsNumber	67
setHopsNumber	67
getHostPortsRedundancy	67
setHostPortsRedundancy	67
getVolumeSharingScope	67
setVolumeSharingScope	67
getOwner	67
setOwner	67
PolicyResponse	69
Constructors	69
Methods	69
getPolicies	69
setPolicies	69
Port	70
Direct Known Subclasses	70
Constructors	70
Methods	70
getId	70
Returns	70
setId	70
Parameters	70

getState	70
Returns	71
setState.	71
Parameters	71
getStatus	71
Returns	71
setStatus.	71
Parameters	71
getWwn.	71
Returns	71
setWwn.	71
Parameters	72
getDeviceld.	72
Returns	72
setDeviceld.	72
Parameters	72
getNodeId.	72
Returns	72
setNodeId.	72
Parameters	72
getConnectedPortId	72
Returns	73
setConnectedPortId	73
Parameters	73
isDead	73
Returns	73
setDead	73
Parameters	73
getName.	73
Returns	73
setName.	73
Parameters	73

PortResponse	75
Constructors	75
Methods	75
getRequestIterator	75
Returns	75
setRequestIterator	75
Parameters	75
getSize	75
Returns	76
setSize	76
Parameters	76
getPorts	76
Returns	76
setPorts	76
Parameters	76
RequestIterator	78
Constructors	78
Methods	78
getNextKey	78
Returns	78
setNextKey	78
Parameters	78
isHasMore	79
Returns	79
setHasMore	79
Parameters	79
getMethodName	79
Returns	79
setMethodName	79
Parameters	79
ReservationRequest	81
Constructors	81

Methods	81
isSatisfied	81
setSatisfied	81
getDisplayName	81
setDisplayName	81
getDescription	81
setDescription	81
getOwner	81
setOwner	82
getRequester	82
setRequester	82
getRequesterEmail	82
setRequesterEmail	82
getState	82
setState	82
getJustification	82
setJustification	82
getComments	82
setComments	82
getCreationDate	83
setCreationDate	83
getDueDate	83
setDueDate	83
getCompletionDate	83
setCompletionDate	83
getErrorCount	83
setErrorCount	83
getWarningCount	83
setWarningCount	83
getTargetType	83
setTargetType	84
getRequestTargets	84
setRequestTargets	84
getProjectId	84

setProjectId	84
getDataCenterId	84
setDataCenterId	84
getSanId	84
setSanId	84
getBusinessUnitId	84
setBusinessUnitId	84
getTicket	85
setTicket	85
ReservationRequirement	87
Constructors	87
Methods	87
isSatisfied	87
setSatisfied	87
getNote	87
setNote	87
getType	87
setType	87
getAttributes	88
setAttributes	88
Task	90
Constructors	90
Methods	90
getHostId	90
getName	90
setName	90
getUserTaskId	90
setUserTaskId	90
getReferenceNumber	90
setReferenceNumber	90
getState	91
setState	91
getStatus	91

setStatus	91
getViolationsCount	91
setViolationsCount	91
getErrorsCount	91
setErrorsCount	91
getOwner	91
setOwner	91
getComments	91
setComments	92
getStartTime	92
setStartTime	92
getEndTime	92
setEndTime	92
getDependedTaskId	92
setDependedTaskId	92
TaskResponse	94
Constructors	94
Methods	94
getTask	94
setTask	94
TaskViolationResponse	95
Constructors	95
Methods	95
getTaskViolations	95
setTaskViolations	95
SANscreen API Endpoint	96
closeSession	97
Throws	97
Parameters	97
Throws	97
findObjectByWwn	98
Parameters	98

Throws.	98
findObjectsByIp	99
Parameters	99
Throws.	99
addHostToApplication	100
Parameters	100
Returns	100
Throws.	100
addReservationRequests	102
Parameters	102
Returns:.	102
Throws:	102
addReservationRequirements	104
Parameters	104
Returns:.	104
Throws:	104
getAllApplications	105
Parameters	105
Returns	105
Throws.	105
getAPIVersion	106
Returns	106
Throws.	106
getApplication	107
Parameters	107
Returns	107
Throws.	107
getApplicationsOfHost	108
Parameters	108
Returns	108
getConnectedPorts	109
Parameters	109

Returns	109
Throws.....	109
getDeviceGroup	110
Parameters	110
Returns	110
Throws.....	110
getDeviceGroups	111
Parameters	111
Returns	111
Throws.....	111
getDeviceGroupsByStorage	112
Parameters	112
Returns	112
Throws.....	112
getDeviceGroupsByVolume	113
Parameters	113
Returns	113
Throws.....	113
getDeviceGroupVolumes	114
Parameters	114
Returns	114
Throws.....	114
.....	114
getDRPathsByVolume	115
Parameters	115
Returns	115
Throws.....	115
getGeneric	116
Parameters	116
Returns	116
Throws.....	116
getGenerics	117

Parameters	117
Returns	117
Throws.	117
getHost	118
Parameters	118
Returns	118
Throws.	118
getHosts	119
Parameters	119
Throws.	119
getHostsOfApplication	120
Parameters	120
Returns	120
Throws.	120
getHostPolicy	121
Parameters	121
Returns	121
Throws.	121
getLogicalPortsByPhysicalPort	122
Parameters	122
Returns	122
Throws.	122
getLogicalPortsByLogicalSwitch	123
Parameters	123
Returns	123
Throws.	123
getLogicalPortsByPhysicalSwitch	124
Parameters	124
Returns	124
Throws.	124
getLogicalSwitchesByPhysicalSwitch	125
Parameters	125

Returns	125
Throws.....	125
getNodeOfDevice	126
Parameters	126
Returns	126
Throws.....	126
getPathPolicy	127
Parameters	127
Returns	127
Throws.....	127
getPaths	128
Parameters	128
Returns	128
Throws.....	128
getPathsByHost	129
Parameters	129
Returns	129
Throws.....	129
getPhysicalPortByLogicalPort.....	130
Parameters	130
Returns	130
Throws.....	130
getPolicies	131
Parameters	131
Returns	131
Throws.....	131
getPort	132
Parameters	132
Returns	132
Throws.....	132
getPorts	133
Parameters	133

Returns	133
Throws.....	133
getPortsOfDevice	134
Parameters	134
Returns	134
Throws.....	134
getPortsOfNode	135
Parameters	135
Returns	135
Throws.....	135
getPortsOfDeviceType	136
Parameters	136
Returns	136
Throws.....	136
getRecentDevice	137
Parameters	137
Throws.....	137
getRecentHost	138
Throws.....	138
Parameters	138
Returns	138
getRecentStorageArray	139
Parameters	139
Returns	139
Throws.....	139
getRecentSwitch.....	140
Parameters	140
Returns	140
Throws.....	140
getRecentTape	141
Parameters	141
Returns	141

Throws.	141
getRecentVolume	142
Parameters	142
Returns	142
Throws.	142
getRecentVolumeMember	144
Parameters	144
Returns	144
Throws.	144
getReservationRequests.	146
Parameters	146
Returns:.	146
Throws:	146
getReservationRequestsByStates	148
Parameters	148
Returns:.	148
Throws:	148
getReservationRequirements	150
Parameters	150
Returns:.	150
Throws:	150
getSANscreenVersion.	151
Returns	151
Throws.	151
getStorageArray	152
Parameters	152
Returns	152
Throws.	152
getStorageArrays	153
Parameters	153
Returns	153
Throws.	153

getSwitch	154
Parameters	154
Returns	154
Throws.	154
getSwitches	155
Parameters	155
Returns	155
Throws.	155
getTape	156
Parameters	156
Returns	156
Throws.	156
getTapes	157
Parameters	157
Returns	157
Throws.	157
getViolations	158
Parameters	158
Returns	158
Throws.	158
getViolationsByHost	159
Parameters	159
Returns	159
Throws.	159
getVolume	160
Parameters	160
Returns	160
Throws.	160
getVolumes	161
Parameters	161
Returns	161
Throws.	161

getVolumesByStorageArray	162
Parameters	162
Returns	162
Throws.	162
getVolumeMapsByStorageArray	163
Parameters	163
Returns	163
Throws.	163
getVolumeMasksByStorageArray	164
Parameters	164
Returns	164
Throws.	164
getVolumeMember	166
Parameters	166
Returns	166
Throws.	166
getVolumeMembersbyVolume	168
Parameters	168
Returns	168
Throws.	168
getVolumesSynchStatesByDRPath	169
Throws.	169
getVolumesSynchStatesBySourceVolume	170
Throws.	170
openSession.	171
Throws.	171
Parameters	171
Returns	171
Throws.	171
removeHostFromApplication.	172
Parameters	172
Returns	172

Throws.	172
removeHostPolicies	173
Parameters	173
Throws.	173
removePolicies	174
Parameters	174
Throws.	174
removeReservationRequirements.	176
Parameters	176
Returns:.	176
Throws:	176
setAnnotationValues.	178
Parameters	178
Returns	178
Throws.	178
setApplication	179
Parameters	179
Returns	179
Throws.	179
setHostsPolicy	180
Parameters	180
Throws.	180
setPathsPolicy	181
Parameters	181
Throws.	181
updateReservationRequirements	182
Parameters	182
Returns:.	182
Throws:	182
updateReservationRequests.	184
Parameters	184
Returns:.	184

Throws:	184
StorageArray	185
Constructors	185
Methods	185
getName	185
Returns	185
setName	185
Parameters	185
getIpl	185
Returns	185
setIpl	186
Parameters	186
isDead	186
Returns	186
setDead	186
Parameters	186
getId	186
Returns	186
setId	186
Parameters	187
getMicrocodeVersion	187
Returns	187
setMicrocodeVersion	187
Parameters	187
getModel	187
Returns	187
setModel	187
Parameters	187
getRawCapacityGB	187
Returns	187
setRawCapacityGB	188
Parameters	188

getSerialNumber.	188
Returns	188
setSerialNumber.	188
Parameters	188
getVendor	188
Returns	188
setVendor	188
Parameters	188
StorageArrayResponse	189
Constructors	189
Methods.	189
getRequestIterator	189
Returns	189
setRequestIterator	189
Parameters	189
getSize	189
Returns	190
setSize	190
Parameters	190
getArrays	190
Returns	190
setArrays.	190
Parameters	190
Switch.	191
Constructors	191
Methods.	191
getName.	191
Returns	191
setName	191
Parameters	191
getIp	191
Returns	191

setIp	192
Parameters	192
isDead	192
Returns	192
setDead	192
Parameters	192
getId	192
Returns	192
setId	192
Parameters	193
getModel	193
Returns	193
setModel	193
Parameters	193
getStatus	193
Returns	193
setStatus	193
Parameters	193
getVendor	193
Returns	193
setVendor	194
Parameters	194
getWwn	194
Returns	194
setWwn	194
Parameters	194
SwitchResponse	195
Constructors	195
Methods	195
getRequestIterator	195
Returns	195
setRequestIterator	195

Parameters	195
getSize	195
Returns	196
setSize	196
Parameters	196
getSwitches	196
Returns	196
setSwitches	196
Parameters	196
SynchronizationResponse	198
Constructors	198
Methods	198
getSynchronizations	198
setSynchronizations	198
Tape	199
Constructors	199
Methods	199
getName	199
Returns	199
setName	199
Parameters	199
getIp	199
Returns	199
setIp	200
Parameters	200
isDead	200
Returns	200
setDead	200
Parameters	200
getId	200
Returns	200
setId	200

Parameters	201
getModel	201
Returns	201
setModel	201
Parameters	201
getSerialNumber	201
Returns	201
setSerialNumber	201
Parameters	201
getVendor	201
Returns	201
setVendor	202
Parameters	202
TapeResponse	203
Constructors	203
Methods	203
getRequestIterator	203
Returns	203
setRequestIterator	203
Parameters	203
getSize	203
Returns	204
setSize	204
Parameters	204
getTapes	204
Returns	204
setTapes	204
Parameters	204
Violation	205
Constructors	205
Methods	205
getHostId	205

setHostId	205
getArrayId	205
setArrayId	205
getVolumeId	205
setVolumeId	205
getType	205
setType	206
ViolationResponse	207
Constructors	207
Methods	207
getViolations	207
setViolations	207
Volume	208
Constructors	208
Methods	208
getId	208
Returns	208
setId	208
Parameters	208
getCapacityGB	208
Returns	208
setCapacityGB	209
Parameters	209
getName	209
Returns	209
setName	209
Parameters	209
getRawCapacityGB	209
Returns	209
setRawCapacityGB	209
Parameters	209
getRedundancy	209

Returns	210
setRedundancy	210
Parameters	210
getType	210
Returns	210
setType	210
Parameters	210
getStorageID	210
Returns	210
setStorageID	210
Parameters	210
getDiskType	211
Returns	211
setDiskType	211
Parameters	211
VolumeMap	212
Constructors	212
Methods	212
getId	212
Returns	212
setId	212
Parameters	212
getLun	212
Returns	212
setLun	213
Parameters	213
getStoragePortId	213
Returns	213
setStoragePortId	213
Parameters	213
getProtocolController	213
Returns	213

setStoragePortId	213
Parameters	214
getVolumeId	214
Returns	214
setVolumeId	214
Parameters	214
getStoragePortWwn	214
Returns	214
setStoragePortWwn	214
Parameters	214
isWwnsecurityvalidity	214
Returns	215
setWwnsecurityvalidity	215
Parameters	215
VolumeMapResponse	216
Constructors	216
Methods.	216
getRequestIterator	216
Returns	216
setRequestIterator	216
Parameters	216
getSize	216
Returns	217
setSize	217
Parameters	217
getVolumeMaps	217
Returns	217
setVolumeMaps	217
Parameters	217
VolumeMask.	218
Constructors	218
Methods.	218

getId	218
Returns	218
setId	218
Parameters	218
getInitiatorPortOrNodeWwn	218
Returns	218
setInitiatorPortOrNodeWwn	219
Parameters	219
getStoragePortWwn	219
Returns	219
setStoragePortWwn	219
Parameters	219
getVolumeld	219
Returns	219
setVolumeld	219
Parameters	219
getStoragePortId	219
Returns	220
setStoragePortId	220
Parameters	220
getProtocolController	221
Returns	221
setProtocolController	221
Parameters	221
VolumeMaskResponse	222
Constructors	222
Methods	222
getRequestIterator	222
Returns	222
setRequestIterator	222
Parameters	222
getSize	223

Returns	223
setSize	223
Parameters	223
getMasks	223
Returns	223
setMasks	223
Parameters	223
VolumeResponse	224
Constructors	224
Methods	224
getRequestIterator	224
Returns	224
setRequestIterator	224
Parameters	225
getSize	225
Returns	225
setSize	225
Parameters	225
getVolumes	225
Returns	225
setVolumes	225
Parameters	225
VolumesSynchronizationState	226
Constructors	226
Methods	226
getSourceStorageId	226
setSourceStorageId	226
getSourceVolumeId	226
setSourceVolumeId	226
getTargetStorageId	226
setTargetStorageId	226
getTargetVolumeId	226

setTargetVolumeId	227
getVolumesSynchronizationState	227
setVolumesSynchronizationState	227
getDetailedSynchronizationState	227
setDetailedSynchronizationState	227
getTimeStamp	227
setTimeStamp.	227
VolumesSynchronizationStateResponse	228
Constructors	228
Methods.	228
getVolumesSynchronizationState	228
setVolumesSynchronizationState	228
Sample Code	229
Java	230
CSV Example in Java	237
CSV Example	239
GetConnectedPorts	240
GetInventory	241
GetReplicas	244
GetVolumes	247
Perl	252
APISessionContext Package.	252
Iterator Package	256
APIVersion Package	259

Sample Scripts	260
Computing Total Storage Per Array	261
Finding Disconnected Hosts.	265
Get Paths	269
Get Storage	272
Host Package	275
Path Package	278
Perl Examples Requirements	281
Requirements	281
Examples.	281
Client side API.	281
SOAPParam Package	282
Timeout Package	284
StorageArray Package	285
Volume Package.	290
Appendix A: WSDL File	294

Overview

What is SANscreen Connect?

SANscreen Connect API enables NetApp customers and Independent Software Vendors (ISVs) to develop applications on top of the SANscreen change and service repository and engine. Applications can use the API to perform IT management tasks, such as generate reports, or integrate SANscreen into existing management systems and processes. The API provides read-only access into the information that resides in the SANscreen repository.

To use the API, you must have a current version of SANscreen, a SANscreen license, and a separate API license.

The API is based on SOAP and Web services. This provides the flexibility to use the API from almost any programming environment, including Java and Perl.

Prerequisites

The following is a list of software and requirements that are necessary in order to develop programs that interface with NetApp's API environment.

- ◆ Download and install latest JDK.
- ◆ To use **build.xml**, that is part of the provided Java examples, download and install Apache ANTI.
- ◆ Download and unzip: <http://localhost/download/sanscreenapi-examples.zip>
- ◆ Configure environment variables for ANT and JDK to run together. If you want quick access to Ant, you must setup JAVA_HOME, PATH.
- ◆ To use our Perl-based examples, you must install a Perl interpreter. ActivePerl is available for free download.
- ◆ Perl examples have a readme.txt file that explains what Perl libraries must be downloaded and installed.

Note

To develop in Java or Perl you must be able to configure environment variables.

Note

NetApp does not actively support languages other than Java for use with the API.

Main API Object Types

The API provides access to the following SAN objects:

- ◆ Devices: Host, HBA, Tape, Storage Arrays, Switch
- ◆ Physical connectivity
- ◆ Device configurations: Volumes, mapping, masking, zones, fabrics, VSAN
- ◆ Paths
- ◆ Policies
- ◆ Path Violations

Use Cases

This topic describes likely use cases for the SANscreen Connect API. Select any of the following use cases for details.

- ♦ [Integrate Storage Data Into Centralized IT Dashboard](#)
- ♦ [Integrate Storage Monitoring Into Standard Help Desk Network Operations Center \(NOC\)](#)
- ♦ [Chargeback Reports](#)
- ♦ [Software Vendors Integrate the API Into Their Own Applications to Provide SAN Visibility](#)

Integrate Storage Data Into Centralized IT Dashboard

Integrating storage data into a centralized IT dashboard improves visibility and troubleshooting.

IT frameworks are being used as a centralized database for IT assets, configurations, status and relationships. (sometimes referred to as CMDB) These frameworks extend their reach to include storage information from the server to the data. The integrated view is usually used to provide:

- ◆ Consolidated reports including cost, utilization, and activity
- ◆ Consolidated chargeback and capacity planning
- ◆ Information for end-to-end troubleshooting

Integrate Storage Monitoring Into Standard Help Desk Network Operations Center (NOC)

SANscreen API is used to query outstanding violations and notify a monitoring framework system of any service violation such as outage, redundancy or clustering sharing problem. This enables the NOC service personnel to start monitoring the real-time status of the SAN.

SANscreen includes alternative methods for violation alerting using SNMP and email notification. However, using the API provides the required flexibility to add organization-specific logic around the violation notification.

Chargeback Reports

The API is used to provide frequent reports on the allocation of storage to different servers, enabling storage users to be charged based on their actual usage of production and replicated capacity.

Software Vendors Integrate the API Into Their Own Applications to Provide SAN Visibility

Software vendors use the API to build applications that retrieve storage information into their own IT management software to provide visibility from their own applications into storage services, problems and root-cause analysis. Such vendors include IT management framework developers, database management systems, and others.

Frequently Asked Questions

This topic lists questions frequently asked about the SANscreen Connect API for SANscreen.

How is the API licensed?

Using the SANscreen Connect API requires a software license key as well as signing an API license agreement.

What are the main object types?

Through the API, the following objects are accessible:

- ◆ Devices: Host, Generic (HBA), Tape, Storage Arrays, Switch
- ◆ Device configurations: Volumes, mapping, masking, zones, fabrics, VSAN.
- ◆ Paths, Policies, Violations
- ◆ Changes: Device changes, Violation and path changes (ST)
- ◆ Tasks, actions (ST)

How is authentication performed?

- ◆ API access is based on basic authentication over HTTPS, sending the user name and password over an encrypted communication channel.
- ◆ The user name and password must correlate with a user entered in the NetApp User Management screen in the Administration Portal.
- ◆ The user must have Administrator privileges.
- ◆ Every action done through the API is recorded in the SANscreen audit table.

How do I initiate a session?

See [How to Use the SANscreen Connect API](#).

How to Use the SANscreen Connect API

This topic describes how to use the SANscreen Connect API to initiate a session with the SANscreen server, issue queries, retrieve data, and then close the session.

Designed to support large, ever changing environments, the API requires the use of sessions and iterators.

Opening a session provides a session context that is a required argument for all other queries.

The mechanism is similar to the way connections and cursors are used for database access.

- ♦ A session is used to maintain data consistency between multiple queries.
- ♦ Iterators are used to divide data when the data a query returns is larger than a user-defined maximum.
- ♦ To assure consistency, queries to the same session will respond with data that is consistent with a specific point in time.

Prerequisites

To use the SANscreen Connect API, you must:

- ♦ Have a license to use the SANscreen Connect API
- ♦ Make sure the SANscreen server is running at the endpoint location

Connecting to the SANscreen Connect API

To use the SANscreen Connect API:

1. **Connect to the server** - Depending on the platform being used, connecting to the server yields a `SANscreenAPIEndPoint` end point that you can use to call API methods.
2. **Open a session** - Use the method `openSession(String, int)` to open a session. A session is a snapshot of SANscreen data at the time the session is created. Therefore, any queries you make to a session after opening it will not reflect changes that may have occurred after opening the session. The session context variable is a required argument for every method that queries the SANscreen server. The purpose of using a data snapshot, as opposed to live data, is to guarantee that multiple API queries provide a consistent view of SANscreen data.
3. **Issue queries** - Use any of the `get*` methods, such as `getStorageArrays(APISessionContext sessionContext, RequestIterator requestIterator)`, to query the server. The primary `get*` methods are listed in the `SANscreenAPIEndPoint` class.
4. **Retrieve the data** - Queries for multiple values matching a given criteria, such as `getStorageArrays(APISessionContext sessionContext, RequestIterator requestIterator)`, return a response object encapsulating a subset of the values matching a given query. The number of values in the response is limited by the `limit` argument specified when a session is opened. In addition, a response also includes an iterator that is used for determining if there are more values to be queried.

If the iterator indicates that further queries should be issued, a client should repeat querying and retrieving the data providing the iterator with every new request.
5. **Close the session** - Once the client finishes querying the server, the session should be closed by invoking the `closeSession(APISessionContext sessionContext)` method.

Classes

The following table lists the SANSscreen Connect API classes alphabetically. For detailed information about a class, click the name of the class.

Class	Description
APIBadRequestException on page 11	Indicates that a client made a bad request to the API server and the request could not be fulfilled.
APIException on page 23	Indicates errors have occurred during API execution.
APISessionContext on page 24	A session context is required with every query to the API server in order to provide a consistent view of the data.
Generic on page 38	Represents a generic device.
GenericResponse on page 40	A partial response to a SANSscreenAPIEndPoint.getGenerics(APISessionContext, RequestIterator) method call containing an array of Generic objects.
Host on page 42	Represents a host/server in the SAN.
HostResponse on page 45	A partial response to a SANSscreenAPIEndPoint.getHosts(APISessionContext, RequestIterator) method call containing an array of Host objects.
LogicalPort on page 47	Represents a logical port in the SAN.
LogicalPortResponse on page 52	A partial response to any of several method calls that retrieve ports known to SANSscreen in the SAN being monitored. The response contains an array of LogicalPort objects.
Node on page 56	Represents a node in the SAN.
NodeResponse on page 59	A partial response to a SANSscreenAPIEndPoint.getNodesOfDevice(APISessionContext, RequestIterator) method call containing an array of Node objects.
Path on page 61	Represents a path in the SAN.
PathResponse on page 64	A partial response to a SANSscreenAPIEndPoint.getPaths(APISessionContext, RequestIterator) method call containing an array of Path objects
Port on page 70	Represents a port in the SAN.

PortResponse on page 75	A partial response to any of several method calls that retrieve ports known to SANscreen in the SAN being monitored. The response contains an array of Port objects.
RequestIterator on page 78	A RequestIterator is used by SANscreenAPI to track and manage responses to queries issued by clients.
SANscreenAPIEndPoint on page 93	SANscreen API enables users to programmatically query and modify SANscreen's database.
StorageArray on page 185	Represents a storage array in the SAN.
StorageArrayResponse on page 189	A partial response to a SANscreenAPIEndPoint.getStorageArrays(APISessionContext, RequestIterator) method call containing an array of StorageArray objects.
Switch on page 191	Represents a switch in the SAN.
SwitchResponse on page 195	A partial response to method calls that retrieve switch devices known to SANscreen. The response contains an array of Switch objects.
Tape on page 199	Represents a tape storage device in the SAN.
TapeResponse on page 203	A partial response to a SANscreenAPIEndPoint.getTapes(APISessionContext, RequestIterator) method call containing an array of Tape objects.
Volume on page 208	Represents a volume in the SAN.
VolumeMap on page 212	Represents the mapping of a volume to a disk storage port.
VolumeMapResponse on page 216	A partial response to a SANscreenAPIEndPoint.getVolumeMapsByStorageArray(APISessionContext, RequestIterator, String) method call containing an array of VolumeMap objects.
VolumeMaskResponse on page 222	Adds masking between a volume (disk storage port) and a host.
VolumeMaskResponse on page 222	A partial response to the method call SANscreenAPIEndPoint.getVolumeMapsByStorageArray(APISessionContext, RequestIterator, String) containing an array of volume maps.
VolumeResponse on page 224	A partial response to a SANscreenAPIEndPoint.getVolumes(APISessionContext, RequestIterator) method call containing an array of VolumeMap objects.
VolumesSynchronizationState on page 226	Synchronizes volumes depending on their state.
VolumesSynchronizationStateResponse on page 228	Synchronizes volumes depending on the response from them in a given state

APIBadRequestException

Thrown to indicate that a client application provided wrong or inconsistent information to an API method call. All the methods that can throw this type of exception are defined in the class [SANscreenAPIEndPoint](#).

Constructors

[Fields](#) | [Constructors](#) | [Methods](#)

```
public APIBadRequestException()
```

- To be used by WSAPI only (to deserialize an exception).

Methods

[Fields](#) | [Constructors](#) | [Methods](#)

The following methods are available from this class:

fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString, clone, equals, finalize, getClass, hashCode, notify, notifyAll

Action

Gets action for given action ID.

Constructors

Constructors | Methods

```
public Action()
```

Methods

Constructors | Methods

The following methods are available from this class:

getOwner

```
public java.lang.String getOwner()
```

setOwner

```
public void setOwner(java.lang.String owner)
```

getTaskId

```
public java.lang.String getTaskId()
```

setTaskId

```
public void setTaskId(java.lang.String taskId)
```

getType

```
public java.lang.String getType()
```

setType

```
public void setType(java.lang.String type)
```

getActionOrder


```
public int getActionOrder()
```

setActionOrder

```
public void setActionOrder(int actionOrder)
```

getState

```
public java.lang.String getState()
```

setState

```
public void setState(java.lang.String state)
```

getParameters

```
public ActionParam[] getParameters()
```

setParameters

```
public void setParameters(ActionParam[] parameters)
```

getErrors

```
public java.lang.String[] getErrors()
```

setErrors

```
public void setErrors(java.lang.String[] errors)
```

Action

ActionResponse

Gets actions for given task.

Constructors

Constructors | Methods

```
public ActionResponse()
```

- Initiates an ActionResponse object

Methods

Constructors | Methods

The following methods are available from this class:

getActions

```
public Action[] getActions()
```

setActions

```
public void setActions(Action[] actions)Parameters
```

```
public java.lang.String getWwn()
```

AnnotationType

Gets annotation type by name (for example: Tier, Data Center, etc.).

Constructors

Constructors | Methods

```
public AnnotationType()
```

Methods

Constructors | Methods

The following methods are available from this class:

getName

```
public java.lang.String getName()
```

Gets the name of this device (usually, this is the DNS alias of the IP address).

setName

```
public void setName(java.lang.String name)
```

getDisplayName

```
public java.lang.String getDisplayName()
```

setDisplayName

```
public void setDisplayName(java.lang.String displayName)
```

getDescription

```
public java.lang.String getDescription()
```

setDescription

```
public void setDescription(java.lang.String description)
```

getDataType

```
public java.lang.String getDataType()
```

setDataType

```
public void setDataType(java.lang.String dataType)
```

getOwner

```
public java.lang.String getOwner()
```

setOwner

```
public void setOwner(java.lang.String owner)
```

getSupportedObjectTypes

```
public java.lang.String[] getSupportedObjectTypes()
```

setSupportedObjectTypes

```
public void setSupportedObjectTypes(java.lang.String[]  
supportedObjectTypes)
```

getEnumerationValues

```
public AnnotationEnumValue[] getEnumerationValues()
```

setEnumerationValues

```
public void setEnumerationValues(AnnotationEnumValue[]  
enumerationValues)
```

isUserDefined

```
public boolean isUserDefined()
```

setUserDefined

```
public void setUserDefined(boolean userDefined)
```

isAllowMultipleValues

```
public boolean isAllowMultipleValues()
```

setAllowMultipleValues

```
public void setAllowMultipleValues(boolean allowMultipleValues)
```


AnnotationValue

Gets annotation values for object references and annotation types

Constructors

Constructors | Methods

```
public AnnotationValue()
```

Methods

Constructors | Methods

The following methods are available from this class:

getEnumId

```
public java.lang.String getEnumId()
```

setEnumId

```
public void setEnumId(java.lang.String enumId)
```

getTypeId

```
public java.lang.String getTypeId()
```

setTypeId

```
public void setTypeId(java.lang.String typeId)
```

getObjectReference

```
public ObjectReference getObjectReference()
```

setObjectReference

```
public void setObjectReference(ObjectReference objectReference)
```

getValue

```
public java.lang.String getValue()
```


setValue

```
public void setValue(java.lang.String value)
```

getSetTime

```
public long getSetTime()
```

setSetTime

```
public void setSetTime(long setTime)
```

APIException

Thrown to indicate a failure in the API server while processing a request from an API client application. All the methods that can throw this type of exception are defined in the class SANscreenAPIEndPoint.

Constructors

[Fields](#) | [Constructors](#) | [Methods](#)

```
public APIException()
```

- To be used by WSAPI only (for deserializing the exception).

Methods

[Fields](#) | [Constructors](#) | [Methods](#)

The following methods are available in this class:

fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString, clone, equals, finalize, getClass, hashCode, notify, notifyAll

APISessionContext

A session context is required with every query to the API server in order to provide a consistent view of the data. It is also used to enforce the API version being used by the client, as well as the maximum number of values returned in a single query.

Each session context equates to a snapshot of the data that was available when the session was created. Therefore, queries to a session context do not reflect any data changes that may have occurred after a session was opened.

A session context should only be instantiated using the `SANscreenAPIEndPoint.openSession(String, int)` method.

Typically, the life cycle of a session context in a client program should be:

- ♦ **Open a New Session** - the server returns an instance of a session context that should be used for further queries.
- ♦ **Issue Queries** - the client passes the session context to the server with every query, thus maintaining a consistent view data.
- ♦ **Close the Session** - once the client is done querying the server, it should close the session and discard the session context.

Constructors

Constructors | Methods

```
public APISessionContext()
```

- Constructs an uninitialized session context. This constructor should be used only by the WSAPI.

Methods

Constructors | Methods

The following methods are available for this class.

getId

```
public java.lang.String getId()
```

Returns

The unique identifier of the context.

getTime

```
public long getTime()
```

Returns

The time when a session was opened.

getClientVersion

```
public java.lang.String getClientVersion()
```

Returns

The version of the API client that is using the context.

getLimit

```
public int getLimit()
```

Returns

The maximum number of values in a response to a query.

toString

```
public java.lang.String toString()
```

Overrides

toString in class java.lang.Object

Application

Represents an application that a host in the SAN can belong to.

Constructors

Constructors | Methods

```
public Application()
```

- Initializes an Application object. -

Methods

Constructors | Methods

The following methods are available from this class:

getBusinessUnit

```
public java.lang.String getBusinessUnit()
```

setBusinessUnit

```
public void setBusinessUnit(java.lang.String businessUnit)
```

isIgnoreSharing

```
public boolean isIgnoreSharing()
```

setIgnoreSharing

```
public void setIgnoreSharing(boolean ignoreSharing)
```

getName

```
public java.lang.String getName()
```

setName

```
public void setName(java.lang.String name)
```

getPriority

```
public java.lang.String getPriority()
```

setPriority

```
public void setPriority(java.lang.String priority)
```

getHostIds

```
public java.lang.String[] getHostIds()
```

setHostIds

```
public void setHostIds(java.lang.String[] hostIds)
```


ApplicationResponse

Constructors

Constructors | Methods

```
public ApplicationResponse()
```

- Initializes an Application response object.

Methods

Constructors | Methods

The following methods are available from this class:

getApplications

```
public Application[] getApplications()
```

setApplications

```
public void setApplications(Application[] applications)
```

DeviceGroup

Constructors

Constructors | Methods

```
public DeviceGroup()
```

- Initializes a device group object.

Methods

Constructors | Methods

The following methods are available from this class:

getName

```
public java.lang.String getName()
```

setName

```
public void setName(java.lang.String name)
```

getType

```
public java.lang.String getType()
```

setType

```
public void setType(java.lang.String type)
```

getStorageManagementId

```
public java.lang.String getStorageManagementId()
```

setStorageManagementId

```
public void setStorageManagementId(java.lang.String  
storageManagementId)
```

getStorageManagementName

```
public java.lang.String getStorageManagementName()
```

setStorageManagementName

```
public void setStorageManagementName(java.lang.String  
storageManagementName)
```

DeviceGroupResponse

Constructors

Constructors | Methods

```
public DeviceGroupResponse()
```

- Initializes a device group response object.

Methods

Constructors | Methods

The following methods are available from this class:

getDeviceGroups

```
public DeviceGroup[] getDeviceGroups()
```

setDeviceGroups

```
public void setDeviceGroups(DeviceGroup[] deviceGroups)
```

DRPath

Constructors

[Constructors | Methods](#)

```
public DRPath()
```

- Initiates a DRPath object.

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getHostId

```
public java.lang.String getHostId()
```

setHostId

```
public void setHostId(java.lang.String hostId)
```

getStorageId

```
public java.lang.String getStorageId()
```

setStorageId

```
public void setStorageId(java.lang.String storageId)
```

getVolumeId

```
public java.lang.String getVolumeId()
```

setVolumeId

```
public void setVolumeId(java.lang.String volumeId)
```

getReplicaVolumeId

```
public java.lang.String getReplicaVolumeId()
```

setReplicaVolumeId

```
public void setReplicaVolumeId(java.lang.String replicaVolumeId)
```

getRemoteReplicaVolumeId

```
public java.lang.String getRemoteReplicaVolumeId()
```

setRemoteReplicaVolumeId

```
public void setRemoteReplicaVolumeId(java.lang.String  
remoteReplicaVolumeId)
```

getSecondaryReplicaVolumeId

```
public java.lang.String getSecondaryReplicaVolumeId()
```

setSecondaryReplicaVolumeId

```
public void setSecondaryReplicaVolumeId(java.lang.String  
secondaryReplicaVolumeId)
```

getRemoteHostId

```
public java.lang.String getRemoteHostId()
```

setRemoteHostId

```
public void setRemoteHostId(java.lang.String remoteHostId)
```

getRemoteStorageId

```
public java.lang.String getRemoteStorageId()
```

setRemoteStorageId

```
public void setRemoteStorageId(java.lang.String remoteStorageId)
```

DRPathResponse

Constructors

Constructors | Methods

```
public DRPathResponse()
```

- Initiates a DRPath response object.

Methods

Constructors | Methods

The following methods are available from this class:

getDrPaths

```
public DRPath[] getDrPaths()
```

setDrPaths

```
public void setDrPaths(DRPath[] drPaths)
```

FindObjectResponse

Constructors

[Constructors](#) | [Methods](#)

```
public FindObjectResponse()
```

- Initiates a find objects response object.

Methods

[Constructors](#) | [Methods](#)

The following methods are available from this class:

getDevices

```
public DeviceBase[] getDevices()
```

setDevices

```
public void setDevices(DeviceBase[] devices)
```


Generic

Represents a generic device. Generic devices are those known to SANscreen but not yet identified. You can retrieve generic device information using the method call `SANscreenAPIEndPoint.getGenerics(APISessionContext, RequestIterator)`.

Constructors

[Constructors | Methods](#)

```
public Generic()
```

- Initializes a generic object.

Methods

[Constructors | Methods](#)

The following methods are inherited from this class:

getId

```
public java.lang.String getId()
```

Returns

The unique identifier of this object.

setId

```
public void setId(java.lang.String id)
```

Should only be used by WSAPI.

Parameters

id - The unique identifier of this object.

getWwn

```
public java.lang.String getWwn()
```

Returns

The World Wide Name.

setWwn

```
public void setWwn(java.lang.String wwn)
```

Parameters

wwn - The World Wide Name.

isDead

```
public boolean isDead()
```

Returns

True if this device is not currently visible to SANscreen; otherwise false.

setDead

```
public void setDead(boolean dead)
```

Parameters

dead - Indicates if a device is currently visible to SANscreen.

True = visible. **False** = not visible.

GenericResponse

A partial response to a `SANscreenAPIEndPoint.getGenerics(APISessionContext, RequestIterator)` method call containing an array of Generic objects.

Constructors

[Constructors | Methods](#)

```
public GenericResponse()
```

- Initializes a generic response object.

Methods

[Constructors | Methods](#)

The following methods are available from this class:`getRequestIterator`

Returns

The request iterator for this response.

setRequestIterator

```
public void setRequestIterator(RequestIterator iterator)
```

Sets the request iterator to be used with this response. To be used by WSAPI only.

Parameters

iterator - The iterator of this response.

getSize

```
public int getSize()
```

Gets the number of objects contained in this response. Note that this is not the total number of objects matching a query.

Returns

Number of objects returned in this response.

setSize

```
public void setSize(int size)
```

Sets the number of objects returned in this response. To be used by WSAPI only (for deserializing the response).

Parameters

size - Number of objects in this response.

getGenerics

```
public Generic[] getGenerics()
```

Returns

Generic devices known to SANscreen. Generic devices are ports on unknown devices. An unknown device could be a switch, tape, disc storage controller, or an HBA with ports.

setGenerics

```
public void setGenerics(Generic[] generics)
```

Should only be used by the SANscreen WSAPI.

Parameters

generics - Generic devices known to SANscreen. Generic devices are ports on unknown devices. An unknown device could be a switch, tape, disc storage controller, or an HBA with ports.

Host

Represents a host/server in the SAN. You can retrieve host information using the following method calls: `SANscreenAPIEndPoint.getHosts(APISessionContext, RequestIterator)` and `SANscreenAPIEndPoint.getHost(APISessionContext, hostId)`.

Constructors

[Constructors | Methods](#)

```
public Host()
```

- To be used by WSAPI only (for deserializing the exception).

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getName

```
public java.lang.String getName()
```

Gets the name of this device (usually, this is the DNS alias of the IP address).

Returns

The name of this device.

setName

```
public void setName(java.lang.String name)
```

Should only be used by the SANscreen WSAPI. Sets the name of this device (usually, this is the DNS alias of the IP address).

Parameters

name - The name of this device.

getIp

```
public java.lang.String getIp()
```

Returns

The TCP/IP address of this device, cannot be NULL.

setIp

```
public void setIp(java.lang.String ip)
```

Should only be used by the SANscreen WSAPI. Sets the TCP/IP address of this device.

Parameters

ip - The TCP/IP address of this device, cannot be NULL.

isDead

```
public boolean isDead()
```

Returns

True if the device is not currently visible to SANscreen; otherwise set to false. A device isn't visible to SANscreen if no active data-source is currently reporting on it.

setDead

```
public void setDead(boolean dead)
```

Should only be used by the SANscreen WSAPI.

Parameters

dead - True if the device is not currently visible to SANscreen; otherwise false.

getId

```
public java.lang.String getId()
```

Returns

The unique identifier of this object.

setId

```
public void setId(java.lang.String id)
```

Host

Should only be used by WSAPI.

Parameters

id - The unique identifier for this object.

HostResponse

A partial response to a `SANscreenAPIEndPoint.getHosts(APISessionContext, RequestIterator)` method call containing an array of Host objects.

Constructors

[Constructors | Methods](#)

```
public HostResponse()
```

- To be used by WSAPI only (for deserializing the exception).

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getRequestIterator

```
public RequestIterator getRequestIterator()
```

Gets the iterator which is used by the client to determine if there are more objects to be fetched in addition to the objects in this response, used by the server to determine which objects to return in subsequent invocations.

Returns

The request iterator for this response.

setRequestIterator

```
public void setRequestIterator(RequestIterator iterator)
```

To be used by WSAPI only. Sets the request iterator to be used with this response.

Parameters

iterator - The iterator of this response.

getSize

```
public int getSize()
```

Gets the number of objects contained in this response. Note that this is not the total number of objects matching a query.

Returns

Number of objects returned in this response.

setSize

```
public void setSize(int size)
```

To be used by WSAPI only. Sets the number of objects returned in this response.

Parameters

size - Number of objects in this response.

getHosts

```
public Host[] getHosts()
```

Returns

The hosts and servers contained in the host response.

setHosts

```
public void setHosts(Host[] hosts)
```

Should only be used by WSAPI.

Parameters

hosts - The hosts in the host response.

LogicalPort

Represents a logical port in the SAN. You can retrieve port information.

Constructors

[Constructors | Methods](#)

```
public LogicalPort()
```

- Initiates the logical port object

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getId

```
public java.lang.String getId()
```

Gets the unique identifier of this object.

Returns

The unique identifier of this object.

setId

```
public void setId(java.lang.String id)
```

Should only be used by WSAPI. Sets the unique identifier of this object.

Parameters

id - The unique identifier for this object.

getState

```
public java.lang.String getState()
```

Returns

Logical port state

setState

```
public void setState(java.lang.String state)
```

Parameters

State - Logical port state

getStatus

```
public java.lang.String getStatus()
```

Returns

Logical port status - either connected or disconnected

setStatus

```
public void setStatus(java.lang.String status)
```

Parameters

Status - Logical port status - either connected or disconnected

getWwn

```
public java.lang.String getWwn()
```

Returns

The World Wide Name of the port.

setWwn

```
public void setWwn(java.lang.String wwn)
```

Parameters

wwn - The World Wide Name.

getDeviceId

```
public java.lang.String getDeviceId()
```

Returns

The ID of the device to which the node belongs.

setDeviceId

```
public void setDeviceId(java.lang.String deviceId)
```

Parameters

deviceId - The ID of the device to which the node belongs..

getNodeId

```
public java.lang.String getNodeId()
```

Returns

ID of the device that corresponds to the node that this port belongs to. For example, HBA of the host or protocol controller of the storage device.

setNodeId

```
public void setNodeId(java.lang.String nodeId)
```

Parameters

nodeId - ID of the device that corresponds to the node that this port belongs to. For example, HBA of the host or protocol controller of the storage device.

getConnectedPortId

```
public java.lang.String getConnectedPortId()
```

Returns

The ID of the port to which the other port is connected.

setConnectedPortId

```
public void setConnectedPortId(java.lang.String connectedPortId)
```

Parameters

connectedPortId - The ID of the port to which the other port is connected.

isDead

```
public boolean isDead()
```

Returns

True if this device is not currently visible to SANscreen.

setDead

```
public void setDead(boolean dead)
```

Parameters

dead - True if this device is not currently visible to SANscreen.

getName

```
public java.lang.String getName()
```

Returns

Name of the logical port.

setName

```
public void setName(java.lang.String name)
```

Parameters

name - Name of the logical port.

getPhysicalPortId

```
public java.lang.String getPhysicalPortId()
```

Returns

The physical port corresponding to the logical port.

setPhysicalPortId

```
public void setPhysicalPortId(java.lang.String physicalPortId)
```

Parameters

physicalPortId - The physical port corresponding to the logical port.

LogicalPortResponse

A partial response to any of several method calls that retrieve ports known to SANscreen in the SAN being monitored. The response contains an array of LogicalPort objects. It is returned by the following methods:

- ♦ SANscreenAPIEndPoint.getConnectedPorts(APISessionContext, RequestIterator)
- ♦ SANscreenAPIEndPoint.getLogLogicalPortsByPhysicalPort(APISessionContext, RequestIterator)
- ♦ SANscreenAPIEndPoint.getLogLogicalPortsByLogicalSwitch(APISessionContext, RequestIterator)
- ♦ SANscreenAPIEndPoint.getLogLogicalPortsByPhysicalSwitch(APISessionContext, RequestIterator)

Constructors

[Constructors | Methods](#)

```
public LogicalPortResponse()
```

- Initiates the logical port response object

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getRequestIterator

```
public RequestIterator getRequestIterator()
```

Gets the iterator which is used by the client to determine if there are more objects to be fetched in addition to the objects in this response, used by the server to determine which objects to return in subsequent invocations.

Returns

The request iterator for this response.

setRequestIterator

```
public void setRequestIterator(RequestIterator iterator)
```

Sets the request iterator to be used with this response. To be used by WSAPI only (for deserializing this response).

Parameters

iterator - The iterator of this response.

getSize

```
public int getSize()
```

Gets the number of objects contained in this response. Note that this is not the total number of objects matching a query.

Returns

Number of objects returned in this response.

setSize

```
public void setSize(int size)
```

Sets the number of objects returned in this response. To be used by WSAPI only (for deserializing the response).

Parameters

size - Number of objects in this response.

getLogicalPorts

```
public LogicalPort[] getLogicalPorts()
```

Returns

The logical ports known to SANscreen.

setLogicalPorts

```
public void setLogicalPorts(LogicalPort[] logicalPorts)
```

Should only be used by WSAPI.

Parameters

logicalPorts - The logical ports known to SANscreen.

Node

Represents a node in the SAN. You can retrieve known nodes using the method call `SANscreenAPIEndPoint.getNodesOfDevice(APISessionContext, RequestIterator)`.

Constructors

[Constructors | Methods](#)

```
public Node()
```

- Initiates the node object

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getId

```
public java.lang.String getId()
```

Gets the unique identifier of this object.

Returns

The unique identifier of this object.

setId

```
public void setId(java.lang.String id)
```

Should only be used by WSAPI. Sets the unique identifier of this object.

Parameters

id - The unique identifier for this object.

getDeviceID

```
public java.lang.String getDeviceID()
```

Returns

The ID of the device where a node resides.

setDeviceID

```
public void setDeviceID(java.lang.String deviceID)
```

Should only be used by WSAPI.

Parameters

deviceID - The ID of the device where a node resides.

getWwn

```
public java.lang.String getWwn()
```

Returns

The World Wide Name of the node.

setWwn

```
public void setWwn(java.lang.String wwn)
```

Should only be used by WSAPI.

Parameters

wwn - The World Wide Name of the node.

isDead

```
public boolean isDead()
```

Returns

True if this device is not currently visible to SANscreen; otherwise false.

setDead

```
public void setDead(boolean dead)
```

Should only be used by WSAPI.

Parameters

dead - Indicates whether a device is currently visible to SANscreen.

True = visible. **False** = not visible.

NodeResponse

A partial response to a `SANscreenAPIEndPoint.getNodesOfDevice(APISessionContext, RequestIterator)` method call containing an array of Node objects.

Constructors

[Constructors | Methods](#)

```
public NodeResponse()
```

- Initiates a node response object.

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getRequestIterator

```
public RequestIterator getRequestIterator()
```

Gets the iterator which is used by the client to determine if there are more objects to be fetched in addition to the objects in this response, used by the server to determine which objects to return in subsequent invocations.

Returns

The request iterator for this response.

setRequestIterator

```
public void setRequestIterator(RequestIterator iterator)
```

Sets the request iterator to be used with this response. To be used by WSAPI only (for deserializing this response).

Parameters

iterator - The iterator of this response.

getSize

```
public int getSize()
```

Gets the number of objects contained in this response. Note that this is not the total number of objects matching a query.

Returns

Number of objects returned in this response.

setSize

```
public void setSize(int size)
```

Sets the number of objects returned in this response. To be used by WSAPI only (for deserializing the response).

Parameters

size - Number of objects in this response.

getNodes

```
public Node[] getNodes()
```

Returns

The nodes of a device, such as a host's HBA or the controller of a storage device.

setNodes

```
public void setNodes(Node[] nodes)
```

Parameters

nodes - The nodes of a device, such as a host's HBA or the controller of a storage device.

Path

Represents a path in the SAN. You can retrieve known paths using the `SANscreenAPIEndPoint.getPaths(APISessionContext, RequestIterator)` method call.

Constructors

[Constructors | Methods](#)

```
public Path()
```

- Initiates a path object

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getId

```
public java.lang.String getId()
```

Gets the unique identifier of this object.

Returns

The unique identifier of this object.

setId

```
public void setId(java.lang.String id)
```

Should only be used by WSAPI. Sets the unique identifier of this object.

Parameters

id - The unique identifier for this object.

getArrayId

```
public java.lang.String getArrayId()
```


Returns

The name of the storage device where the data for this path resides.

setArrayId

```
public void setArrayId(java.lang.String arrayId)
```

Should be used only by WSAPI.

Parameters

arrayId - The name of the storage device where the data for this path resides.

getHostId

```
public java.lang.String getHostId()
```

Returns

The name of the host from which the path originates.

setHostId

```
public void setHostId(java.lang.String hostId)
```

Should only be used by WSAPI.

Parameters

hostId - The name of the host from which the path originates.

getVolumeId

```
public java.lang.String getVolumeId()
```

Returns

The name of the volume where the data for this path resides.

setVolumeId

```
public void setVolumeId(java.lang.String volumeId)
```

Should only be used by WSAPI.

Path

Parameters

volumeld - The name of the volume where the data for this path resides.

PathResponse

A partial response to a `SANscreenAPIEndPoint.getPaths(APISessionContext, RequestIterator)` method call containing an array of Path objects.

Constructors

[Constructors | Methods](#)

```
public PathResponse()
```

- Initiates a path response object

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getRequestIterator

```
public RequestIterator getRequestIterator()
```

Gets the iterator which is used by the client to determine if there are more objects to be fetched in addition to the objects in this response, used by the server to determine which objects to return in subsequent invocations.

Returns

The request iterator for this response.

setRequestIterator

```
public void setRequestIterator(RequestIterator iterator)
```

Should only be used by WSAPI. Sets the request iterator to be used with this response.

Parameters

iterator - The iterator of this response.

getSize

```
public int getSize()
```

Gets the number of objects contained in this response. Note that this is not the total number of objects matching a query.

Returns

Number of objects returned in this response.

setSize

```
public void setSize(int size)
```

Should only be used by WSAPI. Sets the number of objects returned in this response.

Parameters

size - Number of objects in this response.

getPaths

```
public Path[] getPaths()
```

Returns

The logical paths detected by SANSscreen in the SAN being monitored

setPaths

```
public void setNodes(Path[] paths)
```

Should only be used by WSAPI.

Parameters

paths - The logical paths detected by SANSscreen in the SAN being monitored.

Policy

Constructors

Constructors | Methods

```
public Policy()
```

- Initiates a policy object

Methods

Constructors | Methods

The following methods are available from this class:

getHostId

```
public java.lang.String getHostId()
```

setHostId

```
public void setHostId(java.lang.String hostId)
```

getArrayId

```
public java.lang.String getArrayId()
```

setArrayId

```
public void setArrayId(java.lang.String arrayId)
```

getVolumId

```
public java.lang.String getVolumeId()
```

setVolumId

```
public void setVolumeId(java.lang.String volumeId)
```

getRedundancy

```
public java.lang.String getRedundancy()
```

setRedundancy

```
public void setRedundancy(java.lang.String redundancy)
```

getHopsNumber

```
public int getHopsNumber()
```

setHopsNumber

```
public void setHopsNumber(int hopsNumber)
```

getHostPortsRedundancy

```
public int getHostPortsRedundancy()
```

setHostPortsRedundancy

```
public void setHostPortsRedundancy(int hostPortsRedundancy)
```

getVolumeSharingScope

```
public java.lang.String getVolumeSharingScope()
```

setVolumeSharingScope

```
public void setVolumeSharingScope(java.lang.String  
volumeSharingScope)
```

getOwner

```
public java.lang.String getOwner()
```

setOwner

```
public void setOwner(java.lang.String owner)
```


PolicyResponse

Constructors

Constructors | Methods

```
public PolicyResponse()
```

- Initiates a policy object

Methods

Constructors | Methods

The following methods are available from this class:

getPolicies

```
public Policy[] getPolicies()
```

setPolicies

```
public void setPolicies(Policy[] policies)
```


Port

Represents a port in the SAN. You can retrieve known ports.

Direct Known Subclasses

LogicalPort

Constructors

Constructors | Methods

```
public Port()
```

- Initiates a Port object

Methods

Constructors | Methods

The following methods are available from this class:

getId

```
public java.lang.String getId()
```

Gets the unique identifier of this object.

Returns

The unique identifier of this object

setId

```
public void setId(java.lang.String id)
```

Should only be used by WSAPI. Sets the unique identifier of this object.

Parameters

id - The unique identifier for this object.

getState

```
public java.lang.String getState()
```

Returns

Port state

setState

```
public void setState(java.lang.String state)
```

Parameters

state - Port state

getStatus

```
public java.lang.String getStatus()
```

Returns

Port Status - either connected or disconnected.

setStatus

```
public void setStatus(java.lang.String status)
```

Parameters

Status - Port status - either connected or disconnected..

getWwn

```
public java.lang.String getWwn()
```

Returns

The World Wide Name of the port.

setWwn

```
public void setWwn(java.lang.String wwn)
```

Parameters

wwn - The World Wide Name.

getDeviceId

```
public java.lang.String getDeviceId()
```

Returns

The ID of the device to which the node belongs.

setDeviceId

```
public void setDeviceId(java.lang.String deviceId)
```

Parameters

deviceId - The ID of the device to which the node belongs..

getNodeId

```
public java.lang.String getNodeId()
```

Returns

ID of the device that corresponds to the node that this port belongs to. For example, HBA of the host or protocol controller of the storage device.

setNodeId

```
public void setNodeId(java.lang.String nodeId)
```

Parameters

nodeId - ID of the device that corresponds to the node that this port belongs to. For example, HBA of the host or protocol controller of the storage device.

getConnectedPortId

```
public java.lang.String getConnectedPortId()
```

Returns

The ID of the port to which the other port is connected.

setConnectedPortId

```
public void setConnectedPortId(java.lang.String connectedPortId)
```

Parameters

connectedPortId - The ID of the port to which the other port is connected.

isDead

```
public boolean isDead()
```

Returns

True if this device is not currently visible to SANscreen.

setDead

```
public void setDead(boolean dead)
```

Parameters

dead - True if this device is not currently visible to SANscreen.

getName

```
public java.lang.String getName()
```

Returns

Name of the port.

setName

```
public void setName(java.lang.String name)
```

Parameters

name - Name of the port.

PortResponse

A partial response to any of several method calls that retrieve ports known to SANscreen in the SAN being monitored. The response contains an array of Port objects.

Constructors

[Constructors | Methods](#)

```
public PortResponse()
```

- Initiates a Port Response object

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getRequestIterator

```
public RequestIterator getRequestIterator()
```

Gets the iterator which is used by the client to determine if there are more objects to be fetched in addition to the objects in this response, used by the server to determine which objects to return in subsequent invocations.

Returns

The request iterator for this response.

setRequestIterator

```
public void setRequestIterator(RequestIterator iterator)
```

Sets the request iterator to be used with this response. To be used by WSAPI only (for deserializing this response).

Parameters

iterator - The iterator of this response.

getSize

```
public int getSize()
```

Gets the number of objects contained in this response. Note that this is not the total number of objects matching a query.

Returns

Number of objects returned in this response.

setSize

```
public void setSize(int size)
```

Sets the number of objects returned in this response. To be used by WSAPI only (for deserializing the response).

Parameters

size - Number of objects in this response.

getPorts

```
public Port[] getPorts()
```

Returns

The request iterator for this response.

setPorts

```
public void setPorts(Port[] ports)
```

Parameters

ports - The iterator of this response.

RequestIterator

A RequestIterator is used by SANSscreen API to track and manage responses to queries issued by clients. The number of objects in a single response is limited by a session's setting; clients use the iterator in each response to determine if there are more objects to be fetched by invoking the query again. The server uses the iterator to determine which objects should be returned in subsequent invocations of the query.

Constructors

[Constructors | Methods](#)

```
public RequestIterator()
```

- To be used by WSAPI only (for deserializing the iterator).

Methods

[Constructors | Methods](#)

The following methods are available from this class.

getNextKey

```
public java.lang.String getNextKey()
```

Gets the identifier of objects which should be returned on the next invocation of the query. Used internally by the API server.

Returns

The identifier of the objects that should be returned on the next invocation of the query.

setNextKey

```
public void setNextKey(java.lang.String nextKey)
```

Sets the identifier of the objects which will be returned on the next invocation of the query. To be used by WSAPI only (for deserializing this iterator).

Parameters

nextKey - Identifies the objects that will be returned on the next invocation of the query.

isHasMore

```
public boolean isHasMore()
```

Tells the client if there are more objects to be fetched by subsequent invocations of the query.

Returns

True if the client should invoke the query again using this iterator to get the next bulk of objects matching the query.

setHasMore

```
public void setHasMore(boolean hasMore)
```

Sets the flag telling the client if there are more objects to be fetched. To be used by WSAPI only (for deserializing this response).

Parameters

hasMore - True if the client should invoke the query again using this iterator to get the next bulk of objects matching the query).

getMethodName

```
public java.lang.String getMethodName()
```

Gets the name of the API query for which this iterator is used as part of the response.

Returns

The name of the query for which this iterator was created.

setMethodName

```
public void setMethodName(java.lang.String methodName)
```

Sets the name of the API query for which this iterator is used as part of the response. To be used by WSAPI only (for deserializing this response).

Parameters

methodName - The name of the query for which this iterator was created.

ReservationRequest

Constructors

[Constructors | Methods](#)

```
public ReservationRequest()
```

Methods

[Constructors | Methods](#)

The following methods are available from this class:

isSatisfied

```
public boolean isSatisfied()
```

setSatisfied

```
public void setSatisfied(boolean satisfied)
```

getDisplayName

```
public java.lang.String getDisplayName()
```

setDisplay_name

```
public void setDisplay_name(java.lang.String displayName)
```

getDescription

```
public java.lang.String getDescription()
```

setDescription

```
public void setDescription(java.lang.String description)
```

getOwner

```
public java.lang.String getOwner()
```

setOwner

```
public void setOwner(java.lang.String owner)
```

getRequester

```
public java.lang.String getRequester()
```

setRequester

```
public void setRequester(java.lang.String requester)
```

getRequesterEmail

```
public java.lang.String getRequesterEmail()
```

setRequesterEmail

```
public void setRequesterEmail(java.lang.String requesterEmail)
```

getState

```
public java.lang.String getState()
```

setState

```
public void setState(java.lang.String state)
```

getJustification

```
public java.lang.String getJustification()
```

setJustification

```
public void setJustification(java.lang.String justification)
```

getComments

```
public java.lang.String getComments()
```

setComments

```
public void setComments(java.lang.String comments)
```

getCreationDate

```
public long getCreationDate()
```

setCreationDate

```
public void setCreationDate(long creationDate)
```

getDueDate

```
public long getDueDate()
```

setDueDate

```
public void setDueDate(long dueDate)
```

getCompletionDate

```
public long getCompletionDate()
```

setCompletionDate

```
public void setCompletionDate(long completionDate)
```

getErrorCount

```
public int getErrorCount()
```

setErrorCount

```
public void setErrorCount(int errorCount)
```

getWarningCount

```
public int getWarningCount()
```

setWarningCount

```
public void setWarningCount(int warningCount)
```

getTargetType

```
public java.lang.String getTargetType()
```

setTargetType

```
public void setTargetType(java.lang.String targetType)
```

getRequestTargets

```
public ReservationRequestTarget[] getRequestTargets()
```

setRequestTargets

```
public void setRequestTargets(ReservationRequestTarget[] requestTargets)
```

getProjectId

```
public long getProjectId()
```

setProjectId

```
public void setProjectId(long projectId)
```

getDataCenterId

```
public long getDataCenterId()
```

setDataCenterId

```
public void setDataCenterId(long dataCenterId)
```

getSanId

```
public long getSanId()
```

setSanId

```
public void setSanId(long sanId)
```

getBusinessUnitId

```
public long getBusinessUnitId()
```

setBusinessUnitId

```
public void setBusinessUnitId(long businessUnitId)
```

getTicket

```
public java.lang.String getTicket()
```

setTicket

```
public void setTicket(java.lang.String ticket)
```


ReservationRequirement

Represents requirement for Reservation Request

Constructors

Constructors | Methods

```
public ReservationRequirement()
```

Methods

Constructors | Methods

The following methods are available from this class:

isSatisfied

```
public boolean isSatisfied()
```

setSatisfied

```
public void setSatisfied(boolean satisfied)
```

getNote

```
public java.lang.String getNote()
```

setNote

```
public void setNote(java.lang.String note)
```

getType

```
public java.lang.String getType()
```

setType

```
public void setType(java.lang.String type)
```

getAttributes

```
public Attribute[] getAttributes()
```

setAttributes

```
public void setAttributes(Attribute[] attributes)
```


Task

Gets planning task by ID.

Constructors

Constructors | Methods

```
public Task()
```

Methods

Constructors | Methods

The following methods are available from this class:

getHostId

```
public java.lang.String getHostId()
```

getName

```
public java.lang.String getName()
```

setName

```
public void setName(java.lang.String name)
```

getUserTaskId

```
public java.lang.String getUserTaskId()
```

setUserTaskId

```
public void setUserTaskId(java.lang.String userTaskId)
```

getReferenceNumber

```
public java.lang.String getReferenceNumber()
```

setReferenceNumber

```
public void setReferenceNumber(java.lang.String referenceNumber)
```

getState

```
public java.lang.String getState()
```

setState

```
public void setState(java.lang.String state)
```

getStatus

```
public java.lang.String getStatus()
```

setStatus

```
public void setStatus(java.lang.String status)
```

getViolationsCount

```
public int getViolationsCount()
```

setViolationsCount

```
public void setViolationsCount(int violationsCount)
```

getErrorsCount

```
public int getErrorsCount()
```

setErrorsCount

```
public void setErrorsCount(int errorsCount)
```

getOwner

```
public java.lang.String getOwner()
```

setOwner

```
public void setOwner(java.lang.String owner)
```

getComments

```
public java.lang.String getComments()
```

setComments

```
public void setComments(java.lang.String comments)
```

getStartTime

```
public long getStartTime()
```

setStartTime

```
public void setStartTime(long startTime)
```

getEndTime

```
public long getEndTime()
```

setEndTime

```
public void setEndTime(long endTime)
```

getDependedTaskId

```
public java.lang.String getDependedTaskId()
```

setDependedTaskId

```
public void setDependedTaskId(java.lang.String dependedTaskId)
```


TaskResponse

Gets all planning tasks.

Constructors

Constructors | Methods

```
public TaskResponse()
```

Methods

Constructors | Methods

The following methods are available from this class:

getTask

```
public java.lang.String getTasks()
```

setTask

```
public void setTasks(java.lang.String details)
```

TaskViolationResponse

Gets future violations for a given task.

Constructors

Constructors | Methods

```
public TaskViolationResponse()
```

Methods

Constructors | Methods

The following methods are available from this class:

getTaskViolations

```
public TaskViolation[] getTaskViolations()
```

setTaskViolations

```
public void setTaskViolations(TaskViolation[] taskViolations)
```

SANscreen API Endpoint

closeSession

```
public boolean closeSession(APISessionContext sessionContext)
```

Throws

- ♦ **APIException**
- ♦ **APIBadRequestException**
- ♦ **java.rmi.RemoteException** - Closes an API session, discarding any resources being used by the server.

Parameters

sessionContext - Identifies the session to close. The session context is returned by the [openSession](#) method.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - If the sessionContext is null.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

findObjectByWwn

```
public DeviceBase findObjectByWwn(APISessionContext sessionContext,  
                                   RequestIterator requestIterator,  
                                   java.lang.String wwn)
```

Finds SAN device based on their WWN if it is switch or WWN of the host adapter or WWN of the port of the host adapter or WWN of the protocol controller or WWN of the port of protocol controller. Fabrics are excluded from search.

Parameters

wwn - WWN value to use in search

Throws

- ♦ **APIException**
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more hosts to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException**

findObjectsByIp

```
public DeviceResponse findObjectsByIp(APISessionContext
                                         sessionContext,
                                         RequestIterator requestIterator,
                                         java.lang.String ip)
```

Finds SAN devices based on their ip address/dns name which is a regular expression.

Parameters

ip - regular expression for device ip

Throws

- ♦ **APIException**
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more hosts to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** -

addHostToApplication

```
public boolean addHostToApplication(APISessionContext sessionContext,  
                                     java.lang.String hostIp,  
                                     java.lang.String hostName,  
                                     java.lang.String applicationId)
```

- Associates a host with an application

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the hosts that were known at the time the session was created
- ♦ **hostIp** - ip of the host to be associated with the application
- ♦ **hostName** - name of the host to be associated with the application
- ♦ **applicationId** - id of the application to associate with the host

Returns

true if the host was not associated before with the application

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more hosts to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

addReservationRequests

```
public ReservationRequests[] addReservationRequests(APISessionContext  
sessionContext,  
  
                                                    java.lang.String[] states)  
                                                    throws APIException,  
                                                    java.rmi.RemoteException
```

- Add new request objects for capacity management product

Parameters

- ♦ **sessionContext** - Requests are not historical data
- ♦ **requests** - Requests to add

Returns:

Has to return something for WSAPI to work, so it is a placeholder

Throws:

- ♦ **APIException** - the server experienced an internal error
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

addReservationRequirements

```
public ReservationRequests[]  
addReservationRequirements(APISessionContext sessionContext,  
                             java.lang.String[] states)  
    throws APIException,  
           java.rmi.RemoteException
```

- Add requirement objects of the request for capacity management product

Parameters

- ♦ **sessionContext** - Requests are not historical data
- ♦ **requestId** - ID of the request for which requirements will be added
- ♦ **requirements** - Requirements to add

Returns:

Has to return something for WSAPI to work, so it is a placeholder

Throws:

- ♦ **APIException** - the server experienced an internal error
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getAllApplications

```
public ApplicationResponse getAllApplications (APISessionContext  
                                              sessionContext,  
                                              RequestIterator RequestIterator)
```

- ♦ Retrieves all the applications (see [Application](#)) which are known to SANscreen.
- ♦ Each application contains a list of the hosts that are associated with it.
- ♦ Optional attributes describing the application, such as priority, may be included too.

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the nodes that were known at the time the session was created
- ♦ **requestIterator** - optional, if specified, used to fetch subsequent nodes for a query that exceeded the session limit, if null, begins a new query

Returns

The response includes an array of Applications up to the session limit together with an [RequestIterator](#) which is used to fetch the rest of the nodes by subsequent invocations of this method

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more nodes to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getAPIVersion

```
public java.lang.String getAPIVersion()
```

- Gets the version of the API deployed in the SANscreen server to which the client is connected.

Returns

The version of API deployed in the SANscreen server.

Throws

java.rmi.RemoteException - Sent if there was a failure connecting to the server

getApplication

```
public ApplicationResponse getApplication(APISessionContext sessionContext,  
                                           RequestIterator requestIterator,  
                                           java.lang.String applicationName)
```

- ◆ Retrieves a single application (see [Application](#)) based on its name.
- ◆ The application contains a list of the hosts that are associated with it.
- ◆ Optional attributes describing the application, such as priority, may be included too.

Parameters

- ◆ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the nodes that were known at the time the session was created
- ◆ **requestIterator** - optional, if specified, used to fetch subsequent nodes for a query that exceeded the session limit, if null, begins a new query
- ◆ **applicationName** - unique name of the application.

Returns

The response includes an array of one or no [Application](#) up to the session limit together with an [RequestIterator](#) which is used to fetch the rest of the nodes by subsequent invocations of this method

Throws

- ◆ **APIException** - the server experienced an internal error
- ◆ **APIBadRequestException** - if an iterator was specified but there are no more nodes to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ◆ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getApplicationsOfHost

```
public ApplicationResponse getApplicationsOfHost(APISessionContext
                                                    sessionContext,
                                                    RequestIterator requestIterator,
                                                    java.lang.String hostId)
```

- ◆ Retrieves all the applications (see [Application](#)) that are associated with a given host.
- ◆ Each application contains a list of the hosts that are associated with it.
- ◆ Optional attributes describing the application, such as priority may be included too.

Parameters

- ◆ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the nodes that were known at the time the session was created
- ◆ **requestIterator** - optional, if specified, used to fetch subsequent nodes for a query that exceeded the session limit, if null, begins a new query
- ◆ **hostId** - id of a host.

Returns

The response includes an array of Applications that are associated with the given host, up to the session limit together with an [RequestIterator](#) which is used to fetch the rest of the nodes by subsequent invocations of this method

- ◆ **APIException** - the server experienced an internal error
- ◆ **APIBadRequestException** - if an iterator was specified but there are no more nodes to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ◆ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getConnectedPorts

```
public LogicalPortResponse getConnectedPorts(APISessionContext
                                             sessionContext,
                                             RequestIterator requestIterator,
```

- Retrieves all connected logical-ports associated with the specified session context. You can use [LogicalPort](#) class methods to access logical port data, such as physical port ID and status.

Parameters

- ♦ **sessionContext** - The session context returned by the [openSession](#) method. The context corresponds to the snapshot of data to be used, thus the response will show ports that were known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent ports for a query that exceeded the session limit; if null, begins a new query.

Returns

The response includes an array of logical ports, up to the session limit, together with a RequestIterator, used to fetch the rest of the logical ports by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified but there are no more ports to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getDeviceGroup

Retrieves symmetrix device group (see [DeviceGroup](#)) by its id

```
public DeviceGroup getDeviceGroup(APISessionContext sessionContext,  
                                   java.lang.String deviceId)
```

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the device group that was known at the time the session was created
- ♦ **deviceId** - id of the device group

Returns

Device group object or null if none was found

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if the session is null or input parameter is incorrect
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getDeviceGroups

Retrieves all the symmetrix device groups (see [DeviceGroup](#))

```
public DeviceGroupResponse getDeviceGroups (APISessionContext
                                             sessionContext,
                                             RequestIterator requestIterator)
```

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the device groups that were known at the time the session was created
- ♦ **requestIterator** - optional, if specified, used to fetch subsequent device groups for a query that exceeded the session limit, if null, begins a new query

Returns

The response includes an array of DeviceGroups up to the session limit together with an [RequestIterator](#) which is used to fetch the rest of the device groups by subsequent invocations of this method

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more device groups to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getDeviceGroupsByStorage

Retrieves all the symmetrix device groups (see [DeviceGroup](#)) that have volumes for given storage

```
public DeviceGroupResponse getDeviceGroupsByStorage (APISessionContext
                                                    sessionContext,
                                                    RequestIterator requestIterator,
                                                    java.lang.String storageId)
```

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the device groups that were known at the time the session was created
- ♦ **requestIterator** - optional, if specified, used to fetch subsequent device groups for a query that exceeded the session limit, if null, begins a new query
- ♦ **storageId** - id of the storage to get device groups for

Returns

The response includes an array of DeviceGroups up to the session limit together with an [RequestIterator](#) which is used to fetch the rest of the device groups by subsequent invocations of this method

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more device groups to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getDeviceGroupsByVolume

```
public DeviceGroupResponse getDeviceGroupsByVolume (APISessionContext sessionContext,  
                                                    RequestIterator requestIterator,  
                                                    java.lang.String volumeId)
```

- Retrieves all the symmetrix device groups (see [DeviceGroup](#)) that a volume is part of.

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the device groups that were known at the time the session was created
- ♦ **requestIterator** - optional, if specified, used to fetch subsequent device groups for a query that exceeded the session limit, if null, begins a new query
- ♦ **volumeId** - id of the volume to get device groups for

Returns

The response includes an array of DeviceGroups up to the session limit together with an RequestIterator which is used to fetch the rest of the device groups by subsequent invocations of this method

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more device groups to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getDeviceGroupVolumes

```
public VolumeResponse getDeviceGroupVolumes (APISessionContext
                                              sessionContext,
                                              RequestIterator requestIterator,
                                              java.lang.String deviceGroupId)
```

- Retrieves all the volumes that are part of symmetrix device groups (see [DeviceGroup](#))

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the volumes that were known at the time the session was created
- ♦ **requestIterator** - optional, if specified, used to get subsequent volumes for a query that exceeded the session limit, if null, begins a new query

Returns

The response includes an array of Volumes up to the session limit together with an RequestIterator which is used to fetch the rest of the volumes by subsequent invocations of this method

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more volumes to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getDRPathsByVolume

```
public DRPathResponse getDRPathsByVolume (APISessionContext
                                           sessionContext,
                                           RequestIterator requestIterator,
                                           java.lang.String volumeId)
```

- Retrieves all the DR(disaster recovery)-paths (see [DRPath](#)) detected by SANscreen for a given volume. Volume id can refer to any volume from DRPath for DRPath to be returned in this call.

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the DR-paths that were known at the time the session was created
- ♦ **requestIterator** - optional, if specified, used to fetch subsequent DR-paths for a query that exceeded the session limit, if null, begins a new query
- ♦ **volumeId** - id of the volume to return DR-paths for

Returns

The response includes an array of DRPaths up to the session limit together with an RequestIterator which is used to fetch the rest of the DR-path by subsequent invocations of this method

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more DR-paths to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getGeneric

```
public Generic getGeneric(APISessionContext sessionContext,  
                        java.lang.String id)
```

- Retrieves a generic device. A generic device is a SAN device that is not yet identified.

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus the response will show devices that were known at the time the session was created.
- ♦ **id** - Unique identifier for the generic device.

Returns

A Generic if it exists, otherwise NULL.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getGenerics

```
public GenericResponse getGenerics(APISessionContext
                                     sessionContext
                                     RequestIterator requestIterator)
```

- Retrieves all generic devices. Generic devices are those that are known to SANSscreen (including inactive ones), but not yet identified.

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus the response will show devices that were known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent devices for a query that exceeded the session limit; if null, begins a new query. The session limit is set when a session is opened.

Returns

The response includes an array of Generics up to the session limit together with an RequestIterator which is used to fetch the rest of the devices by subsequent invocations of this method.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified but there are no more devices to retrieve; a specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getHost

```
public Host getHost(APISessionContext sessionContext,  
                   java.lang.String hostId)
```

- Retrieves a host/server.

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus the response will show the hosts that were known at the time the session was created.
- ♦ **hostId** - The ID of the host that needs to be fetched.

Returns

A single instance of Host, if it exists. Otherwise, returns NULL.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getHosts

```
public HostResponse getHosts(APISessionContext sessionContext,  
                             RequestIterator requestIterator)
```

- Retrieves all the hosts/servers known to SANscreen, including inactive hosts.

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus the response will show the hosts that were known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent hosts for a query that exceeded the session limit; if null, begins a new query. The session limit is set when a session is first opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - If an iterator was specified but there are no more hosts to retrieve or if the specified iterator is part from a response to a different query or if the session is null.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getHostsOfApplication

```
public HostResponse getHostsOfApplication(APISessionContext
                                           sessionContext,
                                           RequestIterator requestIterator,
                                           java.lang.String applicationName)
```

- Retrieves all the hosts/servers (see [Host](#)) which are associated with a given application.

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the hosts that were known at the time the session was created
- ♦ **requestIterator** - optional, if specified, used to fetch subsequent hosts for a query that exceeded the session limit, if null, begins a new query
- ♦ **applicationName** - unique name of an application

Returns

The response includes an array of [Hosts](#) up to the session limit together with an [RequestIterator](#) which is used to fetch the rest of the hosts by subsequent invocations of this method

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more hosts to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getHostPolicy

```
public Policy getHostPolicy(APISessionContext sessionContext,  
                           java.lang.String hostId)
```

- Retrieves the policy (see [Policy](#)) defined for host

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the policies that were known at the time the session was created
- ♦ **hostId** - id of the host to get policy for

Returns

Policy for given host which is used to fetch the rest of the volumes by subsequent invocations of this method

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more policies to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getLogicalPortsByPhysicalPort

```
public LogicalPortResponse getLogicalPortsByPhysicalPort  
    (APISessionContext sessionContext,  
     RequestIterator requestIterator,  
     java.lang.String physicalPortId)
```

- Retrieves all logical ports associated with the specified physical port. You can use LogicalPort class methods to access logical port data, such as physical port ID and status.

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus the response will show ports that were known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent ports for a query that exceeded the session limit; if null, begins a new query.
- ♦ **physicalPortID** - Uniquely identifies the physical port whose logical ports are requested.

Returns

The response includes an array of logical ports, up to the session limit, together with a RequestIterator, used to fetch the rest of the logical ports by subsequent invocations of this method. The session limit is set when a session is opened

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified but there are no more ports to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getLogicalPortsByLogicalSwitch

```
public LogicalPortResponse  
getLogicalPortsByLogicalSwitch(APISessionContext sessionContext,  
                                RequestIterator requestIterator,  
                                java.lang.String logicalSwitchId)
```

- Retrieves the logical ports associated with the specified switch. You can use [LogicalPort](#) class methods to access logical port data, such as physical port ID and status.

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus the response can only include ports known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent ports for a query that exceeded the session limit; if null, begins a new query.
- ♦ **logicalSwitchId** - Uniquely identifies the switch whose logical ports are requested.

Returns

The response includes an array of ports, up to the session limit, together with a [RequestIterator](#), used to fetch the rest of the ports by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified but there are no more ports to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getLogicalPortsByPhysicalSwitch

```
public LogicalPortResponse  
getLogicalPortsByPhysicalSwitch(APISessionContext sessionContext,  
                                RequestIterator requestIterator,  
                                java.lang.String switchId)
```

- Retrieves the logical ports associated with the specified switch. You can use [LogicalPort](#) class methods to access logical port data, such as physical port ID and status.

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus the response can only include ports known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent ports for a query that exceeded the session limit; if null, begins a new query.
- ♦ **switchId** - Uniquely identifies the switch whose logical ports are requested.

Returns

The response includes an array of ports, up to the session limit, together with a [RequestIterator](#), used to fetch the rest of the ports by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified but there are no more ports to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getLogicalSwitchesByPhysicalSwitch

```
public SwitchResponse  
getLogicalSwitchesByPhysicalSwitch(APISessionContext sessionContext,  
                                   RequestIterator requestIterator,  
                                   java.lang.String switchId)
```

- Retrieves the logical switches associated with the specified physical switch. You can use [Switch](#) class methods to access switch data, such as WWN and status.

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus the response can only include switches known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent switches for a query that exceeded the session limit; if null, begins a new query.
- ♦ **switchId** - Uniquely identifies the physical switch whose logical switches are requested.

Returns

The response includes an array of switches, up to the session limit, together with a [RequestIterator](#), used to fetch the rest of the switches by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified but there are no more switches to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getNodesOfDevice

```
public NodeResponse getNodesOfDevice(APISessionContext
                                     sessionContext,
                                     RequestIterator requestIterator,
                                     java.lang.String storageArrayId)
```

- Retrieves the nodes of a device, such as a host's HBA or the controller of a storage-device. You can use [Node](#) class methods to access node data, such as WWN and device ID.

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus the response will show the nodes that were known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent nodes for a query that exceeded the session limit; if null, begins a new query.
- ♦ **deviceId** - Uniquely identifies the device whose nodes are requested.

Returns

The response includes an array of nodes, up to the session limit, together with a [RequestIterator](#), used to fetch the rest of the nodes by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified but there are no more nodes to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getPathPolicy

```
public Policy getPathPolicy(APISessionContext sessionContext,  
                           Path path)
```

- Retrieves the policy (see [Policy](#)) defined for path

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the policies that were known at the time the session was created
- ♦ **path** - logical path for which policy will be returned

Returns

[Policy](#) for the path

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more policies to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getPaths

```
public PathResponse getPaths(APISessionContext sessionContext,  
                             RequestIterator requestIterator,
```

- Retrieves all the logical paths detected by SANscreen in the SAN being monitored. You can use [Path](#) class methods to access path data, such as host ID and array ID.

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus the response will show the logical paths that were known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent logical paths for a query that exceeded the session limit; if null, begins a new query.

Returns

The response includes an array of paths, up to the session limit, together with a [RequestIterator](#), used to fetch the rest of the logical-path by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified but there are no more logical-paths to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getPathsByHost

```
public PathResponse getPathsByHost(APISessionContext sessionContext,  
                                     RequestIterator requestIterator,  
                                     java.lang.String hostId)
```

- Retrieves all the logical-paths (see [Path](#)) detected by SANscreen in the SAN being monitored for a given host

Parameters

- ♦ **sessionContext** - Mandatory, the session identifies the snapshot of the data to be used, thus the response will show the logical-paths that were known at the time the session was created
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent logical-paths for a query that exceeded the session limit, if null, begins a new query
- ♦ **hostId** - Id of the host to return paths for

Returns

The response includes an array of Paths up to the session limit together with an [RequestIterator](#) which is used to fetch the rest of the logical-path by subsequent invocations of this method

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more logical-paths to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getPhysicalPortByLogicalPort

```
public PortResponse getPhysicalPortByLogicalPort (APISessionContext
                                                    sessionContext,
                                                    RequestIterator requestIterator,
                                                    java.lang.String logicalPortId)
```

- Retrieves the physical port associated with the specified logical port. You can use [Port](#) class methods to access physical port data, such as WWN and status.

Parameters

- ♦ **sessionContext** - The session context returned by the [openSession](#) method. The context corresponds to the snapshot of data to be used, thus the response can only include a port known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent ports for a query that exceeded the session limit; if null, begins a new query.
- ♦ **logicalPortId** - Uniquely identifies the logical port whose physical port is requested.

Returns

The response includes an array of ports, up to the session limit, together with a [RequestIterator](#), used to fetch the rest of the ports by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified but there are no more ports to retrieve; the specified iterator is part of a response to a different query; or the session is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getPolicies

```
public PolicyResponse getPolicies(APISessionContext sessionContext,  
                                   RequestIterator requestIterator)
```

- Retrieves all the policies (see [Policy](#)) defined for the SAN

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the policies that were known at the time the session was created
- ♦ **requestIterator** - optional, if specified, used to fetch subsequent policies for a query that exceeded the session limit, if null, begins a new query

Returns

The response includes an array of Policies up to the session limit together with an [RequestIterator](#) which is used to fetch the rest of the volumes by subsequent invocations of this method

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more policies to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getPort

```
public Port getPort(APISessionContext sessionContext,  
                  java.lang.String id)
```

- Retrieves a port. You can use Port methods to access port data, such as WWN and status.

Parameters

- ♦ **sessionContext** - The session context returned by the [openSession](#) method. The context corresponds to the snapshot of data to be used, thus the response will show the ports that were known at the time the session was created.
- ♦ **id** - Unique identifier for the port.

Returns

The port, if it exists; otherwise Null.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getPorts

```
public PortResponse getPorts(APISessionContext sessionContext,  
                             RequestIterator requestIterator)
```

- Retrieves all the ports in the SAN. You can use [Port](#) methods to access port data, such as WWN and status.

Parameters

- ♦ **sessionContext** - The session context returned by the [openSession](#) method. The context corresponds to the snapshot of data to be used, thus the response will show the ports that were known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent ports for a query that exceeded the session limit; if null, begins a new query.

Returns

The response includes an array of ports, up to the session limit, together with a [RequestIterator](#), used to fetch the rest of the ports by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified, but there are no more devices to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getPortsOfDevice

```
public PortResponse getPortsOfDevice(APISessionContext
                                     sessionContext,
                                     RequestIterator requestIterator,
                                     java.lang.String storageArrayId)
```

- Retrieves all the ports of a device in all the nodes of the device. You can use [Port](#) methods to access port data, such as WWN and status.

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus the response will show the ports that were known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent ports for a query that exceeded the session limit; if null, begins a new query.
- ♦ **deviceId** - Uniquely identifies the device whose ports are requested.

Returns

The response includes an array of ports, up to the session limit, together with a [RequestIterator](#), used to fetch the rest of the ports by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified but there are no more ports to retrieve; the specified iterator is part from a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getPortsOfNode

```
public PortResponse getPortsOfNode(APISessionContext
                                     sessionContext,
                                     RequestIterator requestIterator,
                                     java.lang.String storageArrayId)
```

- ◆ Retrieves all the ports in a single node of a device. You can use [Port](#) methods to access port data, such as WWN and status.

Parameters

- ◆ **sessionContext** - The session context returned by the [openSession](#) method. The context corresponds to the snapshot of data to be used, thus the response will show the ports that were known at the time the session was created.
- ◆ **requestIterator** - Optional, if specified, used to fetch subsequent ports for a query that exceeded the session limit; if null, begins a new query.
- ◆ **nodeId** - Uniquely identifies the node and device whose ports are requested.

Returns

The response includes an array of ports, up to the session limit, together with a [RequestIterator](#), used to fetch the rest of the ports by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ◆ **APIException** - The server experienced an internal error.
- ◆ **APIBadRequestException** - An iterator was specified but there are no more ports to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ◆ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getPortsOfDeviceType

```
public PortResponse getPortsOfDeviceType(APISessionContext
                                           sessionContext,
                                           RequestIterator requestIterator,
                                           java.lang.String type)
```

- Retrieves ports on all the devices of the specified type. You can use [Port](#) methods to access port data, such as WWN and status.

Parameters

- ♦ **sessionContext** - The session context returned by the [openSession](#) method. The context corresponds to the snapshot of data to be used, thus the response will show the ports that were known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent ports for a query that exceeded the session limit; if null, begins a new query.
- ♦ **type** - Type of the devices.

Returns

The response includes an array of ports, up to the session limit, together with a [RequestIterator](#), used to fetch the rest of the ports by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified, but there are no more devices to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getRecentDevice

```
public DeviceBase getRecentDevice(APISessionContext sessionContext,  
                                     java.lang.String id)
```

- Gets most recent SAN device in the history

Parameters

id - device id

Throws

- ♦ **APIException**
- ♦ **APIBadRequestException**
- ♦ **java.rmi.RemoteException**

getRecentHost

```
public Host getRecentHost(APISessionContext sessionContext,  
                           java.lang.String hostId)
```

- Retrieves a host/server (see [Host](#)) most recent in history

Throws

- ♦ **APIException**
- ♦ **java.rmi.RemoteException**

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the hosts that were known at the time the session was created
- ♦ **hostId** - the id of the Host that needs to be fetched

Returns

a single instance of Host, if exists. Null otherwise.

getRecentStorageArray

```
public StorageArray getRecentStorageArray(APISessionContext  
sessionContext, java.lang.String id)
```

- Retrieves a storage array (see [StorageArray](#)) most recent in history

Parameters

sessionContext - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the arrays that were known at the time the session was created

Returns

the array [StorageArray](#) if exists, Null otherwise

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getRecentSwitch

```
public Switch getRecentSwitch(APISessionContext sessionContext,  
                               java.lang.String id)
```

- Retrieves a switch device (see [Switch](#)) most recent in history

Parameters

sessionContext - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the devices that were known at the time the session was created

Returns

the [Switch](#) if exists, Null otherwise

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred
- ♦ **APIBadRequestException**

getRecentTape

```
public Tape getRecentTape(APISessionContext sessionContext,  
                           java.lang.String id)
```

- Retrieves a tape device (see [Tape](#)) most recent in history

Parameters

sessionContext - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the devices that were known at the time the session was created

Returns

The tape of Tape if exists, Null Otherwise

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getRecentVolume

```
public Volume getRecentVolume(APISessionContext sessionContext,  
                               java.lang.String id)
```

- Retrieves a volume (see [Volume](#)) most recent in history

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the volumes that were known at the time the session was created
- ♦ **id** - id of the volume

Returns

The Volume if exists, Null otherwise

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getRecentVolumeMember

```
public VolumeMember getRecentVolumeMember  
    (APISessionContext sessionContext,  
     java.lang.String Id)
```

- Retrieves a Volume member (see [VolumeMember](#)) most recent in history

Parameters

- ♦ **sessionContext** - Mandatory, the session identifies the snapshot of the data to be used. Thus the response will show the volume members that were known at the time the session was created
- ♦ **id** - the id of the volume member.

Returns

The VolumeMember if it exists, null otherwise.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **java.rmi.RemoteException** - If an RMI or WSAPI failure occurred.

getReservationRequests

```
public ReservationRequest[] getReservationRequests(APISessionContext
sessionContext,
                                                    java.lang.String[] states)
                                                    throws APIException,
                                                    java.rmi.RemoteException
```

- Gets all request objects for capacity management product

Parameters

- ♦ **sessionContext** - Requests are not historical data

Returns:

Array of reservation requests

Throws:

- ♦ **APIException** - the server experienced an internal error
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getReservationRequestsByStates

```
public ReservationRequest[]  
getReservationRequestsByStates(APISessionContext sessionContext,  
                                java.lang.String[] states)  
    throws APIException,  
           java.rmi.RemoteException
```

- Gets request objects for capacity management product filtered by state(s)

Parameters

- ♦ **sessionContext** - Requests are not historical data
- ♦ **States** - request states to get requests for: NEW, OPEN, CANCELLED, REJECTED, etc.)

Returns:

Array of reservation requests

Throws:

- ♦ **APIException** - the server experienced an internal error
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getReservationRequirements

```
public ReservationRequirements[]  
getReservationRequirements(APISessionContext sessionContext,  
                             java.lang.String[] states)  
    throws APIException,  
           java.rmi.RemoteException
```

- Gets all requirement objects of the request for capacity management product

Parameters

- ♦ **sessionContext** - Requests are not historical data
- ♦ **requestId** - ID of the request for which requirements will be retrieved

Returns:

Array of reservation requirements

Throws:

- ♦ **APIException** - the server experienced an internal error
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getSANscreenVersion

```
public java.lang.String getSANscreenVersion()
```

- Gets the version of the SANscreen server to which the client is connected. Use `setSessionContext` to retrieve the version number from the server before calling this method.

Returns

The version of the SANscreen server.

Throws

java.rmi.RemoteException - Sent if there was a failure connecting to the server

getStorageArray

```
public StorageArray getStorageArray(APISessionContext  
                                     sessionContext,  
                                     java.lang.String id)
```

- Retrieves a storage array. After you retrieve a storage array, you can use [StorageArray](#) methods to access array data such as vendor name, model name, and raw capacity.

Parameters

- ♦ **sessionContext** - The session context returned by the [openSession](#) method. The context corresponds to the snapshot of data to be used, thus the response will show devices that were known at the time the session was created.
- ♦ **id** - Unique identifier for the storage array.

Returns

The [StorageArray](#), if it exists; otherwise NULL.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getStorageArrays

```
public StorageArrayResponse getStorageArrays(APISessionContext sessionContext,  
                                              RequestIterator requestIterator)
```

- Retrieves all the storage arrays known to SANscreen (including inactive ones). After you retrieve a storage array, you can use [StorageArray](#) methods to access array data such as vendor name, model name, and raw capacity.

Parameters

- ♦ **sessionContext** - The session context returned by the [openSession](#) method. The context corresponds to the snapshot of data to be used, thus the response will show devices that were known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent devices for a query that exceeded the session limit, if null, begins a new query.

Returns

The response includes an array of StorageArrays, up to the session limit, together with a [RequestIterator](#), used to fetch the rest of the arrays by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified, but there are no more devices to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getSwitch

```
public Switch getSwitch(APISessionContext sessionContext,  
                        java.lang.String id)
```

- Retrieves a switch device.

Parameters

- ♦ **sessionContext** - The session context returned by the [openSession](#) method. The context corresponds to the snapshot of data to be used, thus the response will show devices that were known at the time the session was created.
- ♦ **id** - Unique identifier for the switch.

Returns

A Switch if it exists, otherwise NULL.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - Incorrect input parameters.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getSwitches

```
public SwitchResponse getSwitches(APISessionContext sessionContext,  
                                   RequestIterator requestIterator)
```

- Retrieves all the switch devices which are known to SANscreen (including inactive ones).

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus the response will show devices that were known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent devices for a query that exceeded the session limit; if null, begins a new query. The session limit is set when a session is opened.

Returns

The response includes an array of switches, up to the session limit, together with a [RequestIterator](#), used to fetch the rest of the devices by subsequent invocations of this method.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified but there are no more devices to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getTape

```
public Tape getTape(APISessionContext sessionContext,  
                   java.lang.String id)
```

- Retrieves a tape device. You can use Tape methods to access tape device data, such as vendor name and model name.

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus the response will show devices that were known at the time the session was created.
- ♦ **id** - Unique identifier for the tape.

Returns

The tape, if it exists; otherwise NULL.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getTapes

```
public TapeResponse getTapes(APISessionContext sessionContext,  
                             RequestIterator requestIterator)
```

- Retrieves all the tape devices known to SANscreen (including inactive ones). You can use Tape methods to access tape data, such as vendor name and model name.

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus the response will show devices that were known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent devices for a query that exceeded the session limit; if null, begins a new query.

Returns

The response includes an array of type Tape, up to the session limit, together with a RequestIterator, used to fetch the rest of the devices by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified, but there are no more devices to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getViolations

```
public ViolationResponse getViolations (APISessionContext
                                           sessionContext,
                                           RequestIterator requestIterator)
```

- Retrieves all the violations (see [Violation](#)) in the SAN

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the violations that were known at the time the session was created
- ♦ **requestIterator** - optional, if specified, used to fetch subsequent violations for a query that exceeded the session limit, if null, begins a new query

Returns

The response includes an array of Violations up to the session limit together with a RequestIterator which is used to fetch the rest of the volumes by subsequent invocations of this method

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more violations to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getViolationsByHost

```
public ViolationResponse getViolationsByHost (APISessionContext
                                              sessionContext,
                                              RequestIterator requestIterator,
                                              java.lang.String hostId)
```

- Retrieves all the violations for given host

Parameters

- ♦ **hostId** - id of the host to get violations for
- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the violations that were known at the time the session was created
- ♦ **requestIterator** - optional, if specified, used to retrieve subsequent violations for a query that exceeded the session limit, if null, begins a new query

Returns

The response includes an array of Violations up to the session limit together with a [RequestIterator](#) which is used to retrieve the rest of the volumes by subsequent invocations of this method

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more violations to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

getVolume

```
public Volume getVolume(APISessionContext sessionContext,  
                        java.lang.String id)
```

- Retrieves a volume. You can use Volume class methods to access volume data, such as disk type and capacity.

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus responses can contain only those volumes known at the time the session was created.
- ♦ **id** - Unique identifier for the volume.

Returns

The volume if it exists; otherwise, NULL.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getVolumesByStorageArray

```
public VolumeResponse getVolumesByStorageArray(APISessionContext
                                                sessionContext,
                                                RequestIterator requestIterator,
                                                java.lang.String storageArrayId)
```

- Retrieves all volumes known to SANscreen on the specified storage array. You can use Volume class methods to access volume data, such as disk type and capacity.

Parameters

- ♦ **sessionContext** - The session context returned by the [openSession](#) method. The context corresponds to the snapshot of data to be used, thus responses can contain only those volumes known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent volumes for a query that exceeded the session limit; if null, begins a new query.
- ♦ **storageArrayId** - Uniquely identifies the storage array whose volumes are requested. You can get a list of storage arrays by using the [getStorageArrays](#) method.

Returns

The response includes an array of volumes, up to the session limit, together with a [RequestIterator](#), used to fetch the rest of the volumes by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified but there are no more volumes to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getVolumes

```
public VolumeResponse getVolumes(APISessionContext sessionContext,  
                                   RequestIterator requestIterator,
```

- Retrieves all the volumes in the SAN. You can use [Volume](#) class methods to access volume data, such as disk type and capacity.

Parameters

- ♦ **sessionContext** - The session context returned by the openSession method. The context corresponds to the snapshot of data to be used, thus the response will show the volumes that were known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent volumes for a query that exceeded the session limit; if null, begins a new query.

Returns

The response includes an array of volumes, up to the session limit, together with a RequestIterator, used to fetch the rest of the volumes by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified but there are no more volumes to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getVolumeMapsByStorageArray

```
public VolumeMapResponse  
getVolumeMapsByStorageArray(APISessionContext sessionContext,  
                             RequestIterator requestIterator,  
                             java.lang.String storageArrayId)
```

- Retrieves all the volume-maps known to SANscreen on the specified storage array. You can use [VolumeMap](#) class methods to access volume-map data, such as storage port ID and protocol controller.

Parameters

- ♦ **sessionContext** - The session context returned by the [openSession](#) method. The context corresponds to the snapshot of data to be used, thus the response will show the volume-maps that were known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to fetch subsequent volume-maps for a query that exceeded the session limit; if null, begins a new query.
- ♦ **storageArrayId** - Uniquely identifies the storage array whose volume-maps are requested. You can get a list of storage arrays by using the [getStorageArrays](#) method.

Returns

The response includes an array of volume-maps, up to the session limit, together with a [RequestIterator](#), used to fetch the rest of the volume-maps by subsequent invocations of this method. The session limit is set at the time a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified but there are no more volume-maps to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getVolumeMasksByStorageArray

```
public VolumeMaskResponse  
getVolumeMasksByStorageArray(APISessionContext sessionContext,  
                             RequestIterator requestIterator,  
                             java.lang.String storageArrayId)
```

- Retrieves all the volume-masks known to SANscreen on the specified storage array. You can use [VolumeMask](#) class methods to access volume-mask data, such as storage port WWN and ID.

Parameters

- ♦ **sessionContext** - The session context returned by the [openSession](#) method. The context corresponds to the snapshot of data to be used, thus the response will show the volume-masks that were known at the time the session was created.
- ♦ **requestIterator** - Optional, if specified, used to retrieve subsequent volume-masks for a query that exceeded the session limit; if null, begins a new query.
- ♦ **storageArrayId** - Uniquely identifies the storage array whose volume-masks are requested. You can get a list of storage arrays by using the [getStorageArrays](#) method.

Returns

The response includes an array of volume-masks, up to the session limit, together with a [RequestIterator](#), used to retrieve the rest of the volume-masks by subsequent invocations of this method. The session limit is set when a session is opened.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **APIBadRequestException** - An iterator was specified but there are no more volume-masks to retrieve; the specified iterator is part of a response to a different query; or the session context is NULL.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

getVolumeMember

```
public VolumeMember getVolumeMember(APISessionContext sessionContext,  
                                     java.lang.String Id)
```

- Retrieves a Volume member (see [VolumeMember](#))

Parameters

- ♦ **sessionContext** - Mandatory, the session identifies the snapshot of the data to be used. Thus the response will show the volume members that were known at the time the session was created.

Returns

The VolumeMember if it exists, null otherwise.

Throws

- ♦ **APIException** - The server experienced an internal error.
- ♦ **java.rmi.RemoteException** - If an RMI or WSAPI failure occurred.

getVolumeMembersbyVolume

```
public VolumeMemberResponse getVolumeMembersbyVolume  
    (APISessionContext sessionContext,  
    RequestIterator requestIterator,  
    java.lang.String hostId)
```

- Retrieves all the volume members (see [VolumeMember](#)) of the specified volume which are known to SANscreen

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used. Thus the response will show the volume members that were known at the time the session was created
- ♦ **requestIterator** - optional, if specified, used to retrieve subsequent violations for a query that exceeded the session limit, if null, begins a new query
- ♦ **VolumeId** - mandatory, uniquely identifies the Volume whose volumes are requested

Returns

The VolumeMember if it exists, null otherwise.

Throws

- ♦ **APIException** - If the server experienced an internal error.
- ♦ **APIBadException** - If an iterator was specified but there were no more volumes to retrieve, if the specified iterator is part of a response to a different query, or if the sessionContext is null
- ♦ **java.rmi.RemoteException** - If an RMI or WSAPI failure occurred.

getVolumesSynchStatesByDRPath

```
public VolumesSynchronizationStateResponse  
getVolumesSynchStatesByDRPath(APISessionContext sessionContext,  
                                RequestIterator requestIterator,  
                                java.lang.String drPathId,  
                                long period)
```

- Returns synchronization states for the volumes of the DR Path.

Throws

- ♦ `APIException`
- ♦ `APIBadRequestException`
- ♦ `java.rmi.RemoteException`

getVolumesSynchStatesBySourceVolume

```
public VolumesSynchronizationStateResponse  
getVolumesSynchStatesBySourceVolume(APISessionContext sessionContext,  
                                       RequestIterator requestIterator,  
                                       java.lang.String volumeId,  
                                       long period)
```

- Returns all synchronization states for given source volume.

Throws

- ♦ **APIException**
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more hosts to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException**

openSession

```
public APISessionContext openSession(java.lang.String clientVersion,  
                                       int limit)
```

Throws

- ♦ **APIException**
- ♦ **APIBadRequestException**
- ♦ **java.rmi.RemoteException**

Creates the [APISessionContext](#) which is required by the API server in order to provide a consistent view of the data. Opening a session also defines the API version being used by the client as well as the maximum number of values returned in a single query.

Notice that a session defines a snapshot of the data at the time it was created. Therefore, queries which are issued using the same session will reflect the data at the time the session was created.

When you open a session, the system verifies whether you have a valid API license. If a valid API license is not found, an exception is thrown.

Parameters

- ♦ **clientVersion** - The version of the API client. The server will verify that the API client version number equals the supported server version. If they do not match, an exception is thrown.
- ♦ **limit** - Maximum number of values in a response to a query. A [RequestIterator](#) will be used to fetch values exceeding the limit. This limit cannot exceed 10000.

Returns

A new session.

Throws

- ♦ **APIException** - The server experience an internal error.
- ♦ **APIBadRequestException** - The client version does not match the server version; or limit is greater than MAX_LIMIT or smaller than 1.
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred.

removeHostFromApplication

```
public boolean removeHostFromApplication(APISessionContext
                                           sessionContext,
                                           java.lang.String hostIp,
                                           java.lang.String hostName,
                                           java.lang.String applicationId)
```

- Remove association between a host and an application

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the hosts that were known at the time the session was created
- ♦ **hostIp** - ip of the host that its association with the application should be removed.
- ♦ **hostName** - name of the host that its association with the application should be removed.
- ♦ **applicationId** - id of the application to associate with the host

Returns

True if the host was not associated before with the application

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more hosts to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

removeHostPolicies

```
public boolean removeHostPolicies(APISessionContext sessionContext,  
                                   java.lang.String hostId)
```

- Removes policies (see [Policy](#)) for host including path policies and host policy

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the policies that were known at the time the session was created
- ♦ **hostId** - id of the host for which policies will be removed

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more policies to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

removePolicies

```
public boolean removePolicies(APISessionContext sessionContext,  
                             java.lang.String[] policyIds)
```

- Removes policies (see [Policy](#))

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the policies that were known at the time the session was created
- ♦ **policyIds** - ids of the policies to remove

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more policies to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

removeReservationRequirements

```
public ReservationRequirements[]  
removeReservationRequirements(APISessionContext sessionContext,  
                                java.lang.String[] states)  
    throws APIException,  
           java.rmi.RemoteException
```

- Remove existing requirement objects of the request for capacity management product

Parameters

- ♦ **sessionContext** - Requests are not historical data
- ♦ **requestId**- ID of the request for which requirements will be removed
- ♦ **requirementIds**- IDs of the requirements to remove

Returns:

Has to return something for WSAPI to work, so it is a placeholder

Throws:

- ♦ **APIException** - the server experienced an internal error
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

setAnnotationValues

```
public boolean setAnnotationValues(APISessionContext sessionContext,  
                                   AnnotationValue[] annotationValues)  
    throws APIException,  
           java.rmi.RemoteException  
    - Sets annotation values
```

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the hosts that were known at the time the session was created
- ♦ **annotationValues** - annotation values to set

Returns

True if annotation setting is successful.

Throws

- ♦ **APIException** - if failed to set annotation values due to internal server error
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

setApplication

```
public boolean setApplication(APISessionContext sessionContext,  
                               java.lang.String name,  
                               java.lang.String priority,  
                               java.lang.String businessUnit,  
                               boolean ignoreSharing)
```

- Adds or updated attributes of an application.

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the hosts that were known at the time the session was created
- ♦ **name** - unique name of an application
- ♦ **priority** - priority of the application. can be one of the following values : Critical,High,Medium, Low
- ♦ **businessUnit** - Business Unit which the application belongs to
- ♦ **ignoreSharing** - hosts associated with this application will be ignored when calculating volume sharing violation for other hosts.

Returns

True if new application was created or if attributes of existing application were updated.

If the application existed with the same attributes, nothing will be changed and false will be returned

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more hosts to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

setHostsPolicy

```
public boolean setHostsPolicy(APISessionContext sessionContext,  
                               java.lang.String[] hostIds,  
                               java.lang.String redundancy,  
                               int hopsNumber,  
                               int hostPortsRedundancy,  
                               java.lang.String volumeSharingScope)
```

- Creates policy (see [Policy](#)) for several hosts

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the policies that were known at the time the session was created
- ♦ **hostIds** - ids of the hosts to create policies for
- ♦ **redundancy** - policy attribute redundancy, possible values: none, no-spf, redundant
- ♦ **hopsNumber** - maximum number of hops between host and storage
- ♦ **hostPortsRedundancy** - minimum number of ports per host adapter
- ♦ **volumeSharingScope** - possible values NO, ANY, CLUSTER

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more policies to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

setPathsPolicy

```
public boolean setPathsPolicy(APISessionContext sessionContext,  
                               Path[] paths,  
                               java.lang.String redundancy,  
                               int hopsNumber,  
                               int hostPortsRedundancy,  
                               java.lang.String volumeSharingScope)
```

- Creates policy (see [Policy](#)) for several paths

Parameters

- ♦ **sessionContext** - mandatory, the session identifies the snapshot of the data to be used, thus the response will show the policies that were known at the time the session was created
- ♦ **paths** - paths to create policies for
- ♦ **redundancy** - policy attribute redundancy, possible values: none, no-spf, redundant
- ♦ **hopsNumber** - maximum number of hops between host and storage
- ♦ **hostPortsRedundancy** - minimum number of ports per host adapter
- ♦ **volumeSharingScope** - possible values NO, ANY, CLUSTER

Throws

- ♦ **APIException** - the server experienced an internal error
- ♦ **APIBadRequestException** - if an iterator was specified but there are no more policies to retrieve or if the specified iterator is part from a response to a different query or if the session is null
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

updateReservationRequirements

```
public ReservationRequirements[]  
updateReservationRequirements(APISessionContext sessionContext,  
                                java.lang.String[] states)  
    throws APIException,  
           java.rmi.RemoteException
```

- Update existing requirement objects of the request for capacity management product

Parameters

- ♦ **sessionContext** - Requests are not historical data
- ♦ **requestId** - ID of the request for which requirements will be updated
- ♦ **requirements** - Requirements to update

Returns:

Has to return something for WSAPI to work, so it is a placeholder

Throws:

- ♦ **APIException** - the server experienced an internal error
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

updateReservationRequests

```
public ReservationRequests[]  
updateReservationRequests(APISessionContext sessionContext,  
                           java.lang.String[] states)  
    throws APIException,  
           java.rmi.RemoteException
```

- Update existing request objects for capacity management product

Parameters

- ♦ **sessionContext** - Requests are not historical data
- ♦ **requests** - Requests to update

Returns:

Has to return something for WSAPI to work, so it is a placeholder

Throws:

- ♦ **APIException** - the server experienced an internal error
- ♦ **java.rmi.RemoteException** - RMI or WSAPI failure occurred

StorageArray

Represents a storage array in the SAN. You can retrieve storage device information.

Constructors

[Constructors | Methods](#)

```
public StorageArray()
```

– Initiates a StorageArray object

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getName

```
public java.lang.String getName()
```

Gets the name of this device (usually, this is the DNS alias of the IP address).

Returns

The name of this device.

setName

```
public void setName(java.lang.String name)
```

Sets the name of this device (usually, this is the DNS alias of the IP address).

Parameters

name - The name of this device.

getIp

```
public java.lang.String getIp()
```

Gets the IP address of this device.

Returns

The IP address of this device, cannot be NULL.

setIp

```
public void setIp(java.lang.String ip)
```

Sets the IP address of this device.

Parameters

ip - The IP address of this device, cannot be NULL.

isDead

```
public boolean isDead()
```

Returns

True if the device is not currently visible to SANscreen.

setDead

```
public void setDead(boolean dead)
```

Sets if this device is currently visible to SANscreen. A device isn't visible to SANscreen if no active data-source is currently reporting on it.

Parameters

dead - True if the device is not currently visible to SANscreen.

getId

```
public java.lang.String getId()
```

Gets the unique identifier of this object.

Returns

The unique identifier of this object.

setId

```
public void setId(java.lang.String id)
```

Should only be used by WSAPI. Sets the unique identifier of this object.

Parameters

id - The unique identifier for this object.

getMicrocodeVersion

```
public java.lang.String getMicrocodeVersion()
```

Returns

The microcode version.

setMicrocodeVersion

```
public void setMicrocodeVersion(java.lang.String microcodeVersion)
```

Parameters

microcodeVersion - The microcode version..

getModel

```
public java.lang.String getModel()
```

Returns

The vendor model.

setModel

```
public void setModel(java.lang.String model)
```

Parameters

model - The vendor model.

getRawCapacityGB

```
public double getRawCapacityGB()
```

Returns

The raw capacity of the storage.

setRawCapacityGB

```
public void setRawCapacityGB(double capacityGB)
```

Parameters

capacityGB - The raw capacity of the storage.

getSerialNumber

```
public java.lang.String getSerialNumber()
```

Returns

The serial number of the storage.

setSerialNumber

```
public void setSerialNumber(java.lang.String serialNumber)
```

Parameters

serialNumber - The serial number of the storage.

getVendor

```
public java.lang.String getVendor()
```

Returns

The vendor name, such as EMC or HDS.

setVendor

```
public void setVendor(java.lang.String vendor)
```

Parameters

vendor - The vendor name, such as EMC or HDS, etc.

StorageArrayResponse

A partial response to a [SANscreenAPIEndPoint.getStorageArrays\(APISessionContext, RequestIterator\)](#) method call containing an array of [StorageArray](#) objects.

Constructors

[Constructors](#) | [Methods](#)

```
public StorageArrayResponse()
```

- Initializes a StorageArrayResponse object

Methods

[Constructors](#) | [Methods](#)

The following methods are available from this class:

getRequestIterator

```
public RequestIterator getRequestIterator()
```

Gets the iterator which is used by the client to determine if there are more objects to be fetched in addition to the objects in this response, used by the server to determine which objects to return in subsequent invocations.

Returns

The request iterator for this response.

setRequestIterator

```
public void setRequestIterator(RequestIterator iterator)
```

Sets the request iterator to be used with this response. To be used by WSAPI only (for deserializing this response).

Parameters

iterator - The iterator of this response.

getSize

```
public int getSize()
```


Gets the number of objects contained in this response. Note that this is not the total number of objects matching a query.

Returns

Number of objects returned in this response.

setSize

```
public void setSize(int size)
```

Sets the number of objects returned in this response. To be used by WSAPI only (for deserializing the response).

Parameters

size - Number of objects in this response.

getArrays

```
public StorageArray[] getArrays()
```

Returns

An array of storages.

setArrays

```
public void setArrays(StorageArray[] arrays)
```

Parameters

arrays - An array of storages.

Switch

Represents a switch in the SAN. You can retrieve switch information.

Constructors

[Constructors | Methods](#)

```
public Switch()
```

- Initializes a switch object

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getName

```
public java.lang.String getName()
```

Gets the name of this device (usually, this is the DNS alias of the IP address).

Returns

The name of this device.

setName

```
public void setName(java.lang.String name)
```

Sets the name of this device (usually, this is the DNS alias of the IP address).

Parameters

name - The name of this device.

getIp

```
public java.lang.String getIp()
```

Gets the IP address of this device.

Returns

The IP address of this device, cannot be NULL.

setIp

```
public void setIp(java.lang.String ip)
```

Sets the IP address of this device.

Parameters

ip - The IP address of this device, cannot be NULL.

isDead

```
public boolean isDead()
```

Returns

True if the device is not currently visible to SANscreen.

setDead

```
public void setDead(boolean dead)
```

Sets if this device is currently visible to SANscreen. A device isn't visible to SANscreen if no active data-source is currently reporting on it.

Parameters

dead - True if the device is not currently visible to SANscreen.

getId

```
public java.lang.String getId()
```

Gets the unique identifier of this object.

Returns

The unique identifier of this object.

setId

```
public void setId(java.lang.String id)
```

Should only be used by WSAPI. Sets the unique identifier of this object.

Parameters

id - The unique identifier for this object.

getModel

```
public java.lang.String getModel()
```

Returns

The model of the switch.

setModel

```
public void setModel(java.lang.String model)
```

Parameters

model - The model of the switch.

getStatus

```
public java.lang.String getStatus()
```

Returns

The current status of the switch.

setStatus

```
public void setStatus(java.lang.String status)
```

Parameters

status - The current status of the switch.

getVendor

```
public java.lang.String getVendor()
```

Returns

The vendor of the switch.

setVendor

```
public void setVendor(java.lang.String vendor)
```

Parameters

vendor - The vendor of the switch.

getWwn

```
public java.lang.String getWwn()
```

Returns

The World Wide Name of the switch.

setWwn

```
public void setWwn(java.lang.String wwn)
```

Parameters

wwn - The World Wide Name of the switch.

SwitchResponse

A partial response to two method calls that retrieve switch devices known to SANSscreen. The response contains an array of Switch objects.

Constructors

[Constructors](#) | [Methods](#)

```
public SwitchResponse()
```

- Initializes a switch response object

Methods

[Constructors](#) | [Methods](#)

The following methods are available from this class:

getRequestIterator

```
public RequestIterator getRequestIterator()
```

Gets the iterator which is used by the client to determine if there are more objects to be fetched in addition to the objects in this response, used by the server to determine which objects to return in subsequent invocations.

Returns

The request iterator for this response.

setRequestIterator

```
public void setRequestIterator(RequestIterator iterator)
```

Sets the request iterator to be used with this response. To be used by WSAPI only (for deserializing this response).

Parameters

iterator - The iterator of this response.

getSize

```
public int getSize()
```

Gets the number of objects contained in this response. Note that this is not the total number of objects matching a query.

Returns

Number of objects returned in this response.

setSize

```
public void setSize(int size)
```

Sets the number of objects returned in this response. To be used by WSAPI only (for deserializing the response).

Parameters

size - Number of objects in this response.

getSwitches

```
public Switch[] getSwitches()
```

Returns

A switch object.

setSwitches

```
public void setSwitches(Switch[] switches)
```

Parameters

switches - The switch object.

SynchronizationResponse

Constructors

Constructors | Methods

```
public SynchronizationResponse()
```

Methods

Constructors | Methods

The following methods are available from this class:

getSynchronizations

```
public Synchronization[] getSynchronizations()
```

setSynchronizations

```
public void setSynchronizations(Synchronization[] synchronizations)
```

Tape

Represents a tape storage device in the SAN.

Constructors

Constructors | Methods

```
public Tape()
```

– Initializes a tape object

Methods

Constructors | Methods

The following methods are available from this class:

getName

```
public java.lang.String getName()
```

Gets the name of this device (usually, this is the DNS alias of the IP address).

Returns

The name of this device.

setName

```
public void setName(java.lang.String name)
```

Sets the name of this device (usually, this is the DNS alias of the IP address).

Parameters

name - The name of this device.

getIp

```
public java.lang.String getIp()
```

Gets the IP address of this device.

Returns

The IP address of this device, cannot be NULL.

setIp

```
public void setIp(java.lang.String ip)
```

Sets the IP address of this device.

Parameters

ip - The IP address of this device, cannot be NULL.

isDead

```
public boolean isDead()
```

Returns

True if the device is not currently visible to SANscreen.

setDead

```
public void setDead(boolean dead)
```

Sets if this device is currently visible to SANscreen. A device isn't visible to SANscreen if no active data-source is currently reporting on it.

Parameters

dead - True if the device is not currently visible to SANscreen.

getId

```
public java.lang.String getId()
```

Gets the unique identifier of this object.

Returns

The unique identifier of this object.

setId

```
public void setId(java.lang.String id)
```

Should only be used by WSAPI. Sets the unique identifier of this object.

Parameters

id - The unique identifier for this object.

getModel

```
public java.lang.String getModel()
```

Returns

The model of the tape device.

setModel

```
public void setModel(java.lang.String model)
```

Parameters

model - The model of the tape device.

getSerialNumber

```
public java.lang.String getSerialNumber()
```

Returns

The serial number of the tape device.

setSerialNumber

```
public void setSerialNumber(java.lang.String serialNumber)
```

Parameters

serialNumber - The serial number of the tape device.

getVendor

```
public java.lang.String getVendor()
```

Returns

The vendor of the tape device.

setVendor

```
public void setVendor(java.lang.String vendor)
```

Parameters

vendor - The vendor of the tape device.

TapeResponse

A partial response to a `SANScreenAPIEndPoint.getTapes(APISessionContext, RequestIterator)` method call containing an array of Tape objects.

Constructors

[Constructors](#) | [Methods](#)

```
public TapeResponse()
```

- Initializes a violation object

Methods

[Constructors](#) | [Methods](#)

The following methods are available from this class:

[getRequestIterator](#)

```
public RequestIterator getRequestIterator()
```

Gets the iterator which is used by the client to determine if there are more objects to be fetched in addition to the objects in this response, used by the server to determine which objects to return in subsequent invocations.

[Returns](#)

The request iterator for this response.

[setRequestIterator](#)

```
public void setRequestIterator(RequestIterator iterator)
```

Sets the request iterator to be used with this response. To be used by WSAPI only (for deserializing this response).

[Parameters](#)

iterator - The iterator of this response.

[getSize](#)

```
public int getSize()
```

Gets the number of objects contained in this response. Note that this is not the total number of objects matching a query.

Returns

Number of objects returned in this response.

setSize

```
public void setSize(int size)
```

Sets the number of objects returned in this response. To be used by WSAPI only (for deserializing the response).

Parameters

size - Number of objects in this response.

getTapes

```
public Tape[] getTapes()
```

Returns

An array of tape objects.

setTapes

```
public void setTapes(Tape[] tapes)
```

Parameters

tapes - An array of tape objects

Violation

Constructors

[Constructors | Methods](#)

```
public Violation()
```

- Initializes a violation object

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getHostId

```
public java.lang.String getHostId()
```

setHostId

```
public void setHostId(java.lang.String hostId)
```

getArrayId

```
public java.lang.String getArrayId()
```

setArrayId

```
public void setArrayId(java.lang.String arrayId)
```

getVolumeId

```
public java.lang.String getVolumeId()
```

setVolumeId

```
public void setVolumeId(java.lang.String volumeId)
```

getType

```
public java.lang.String getType()
```


setType

```
public void setType(java.lang.String type)
```

ViolationResponse

Constructors

Constructors | Methods

```
public ViolationResponse()
```

- Initializes a volume response object.

Methods

Constructors | Methods

The following methods are available from this class:

getViolations

```
public Violation[] getViolations()
```

setViolations

```
public void setViolations(Violation[] violations)
```

Volume

Represents a volume in the SAN.

Constructors

Constructors | Methods

```
public Volume()
```

- Initializes a volume object

Methods

Constructors | Methods

The following methods are available from this class:

getId

```
public java.lang.String getId()
```

Gets the unique identifier of this object.

Returns

The unique identifier of this object.

setId

```
public void setId(java.lang.String id)
```

Should only be used by WSAPI. Sets the unique identifier of this object.

Parameters

id - The unique identifier for this object.

getCapacityGB

```
public double getCapacityGB()
```

Returns

The capacity of the volume.

setCapacityGB

```
public void setCapacityGB(double capacityGB)
```

Parameters

capacityGB - The capacity of the volume.

getName

```
public java.lang.String getName()
```

Returns

The name of the volume.

setName

```
public void setName(java.lang.String name)
```

Parameters

name - The name of the volume.

getRawCapacityGB

```
public double getRawCapacityGB()
```

Returns

The raw capacity of the volume.

setRawCapacityGB

```
public void setRawCapacityGB(double rawCapacityGB)
```

Parameters

rawCapacityGB - The raw capacity of the volume.

getRedundancy

```
public java.lang.String getRedundancy()
```

Returns

The volume redundancy.

setRedundancy

```
public void setRedundancy(java.lang.String redundancy)
```

Parameters

redundancy - The volume redundancy.

getType

```
public java.lang.String getType()
```

Returns

The type of volume.

setType

```
public void setType(java.lang.String type)
```

Parameters

type - The volume type.

getStorageID

```
public java.lang.String getStorageID()
```

Returns

The storage id.

setStorageID

```
public void setStorageID(java.lang.String storageID)
```

Parameters

storageID - The storage id.

getDiskType

```
public java.lang.String getDiskType()
```

Returns

The disk type.

setDiskType

```
public void setDiskType(java.lang.String diskType)
```

Parameters

diskType - The disk type.

VolumeMap

Represents mapping between a storage unit volume and a port on the same storage unit. In addition, you can use a volume mask to prevent certain ports from accessing the volume. See, VolumeMask.

Constructors

[Constructors | Methods](#)

```
public VolumeMap()
```

- Initializes a volume map object

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getId

```
public java.lang.String getId()
```

Returns

The unique identifier of this object.

setId

```
public void setId(java.lang.String id)
```

Should only be used by WSAPI. Sets the unique identifier of this object.

Parameters

id - The unique identifier for this object.

getLun

```
public java.lang.String getLun()
```

Returns

The logical unit number used by the host to access this volume.

setLun

```
public void setLun(java.lang.String lun)
```

Should only be used by WSAPI.

Parameters

lun - The logical unit number used by the host to access this volume.

getStoragePortId

```
public java.lang.String getStoragePortId()
```

Returns

The port on the storage device to which the volume is mapped.

setStoragePortId

```
public void setStoragePortId(java.lang.String storagePortId)
```

Should only be used by WSAPI.

Parameters

storagePortId - The port on the storage device to which the volume is mapped.

getProtocolController

```
public java.lang.String getProtocolController()
```

Returns

The protocol controller through which the volume is mapped to the storage port.

setStoragePortId

```
public void setProtocolController(java.lang.String  
protocolController)
```

Should only be used by WSAPI.

Parameters

protocolController - The protocol controller through which the volume is mapped to the storage port.

getVolumeId

```
public java.lang.String getVolumeId()
```

Returns

The ID of the volume being masked to a specified disk storage port.

setVolumeId

```
public void setVolumeId(java.lang.String volumeId)
```

Parameters

volumeId - The ID of the volume being masked to a specified disk storage port.

getStoragePortWwn

```
public java.lang.String getStoragePortWwn()
```

Returns

The World Wide Name for the port to which the volume is mapped.

setStoragePortWwn

```
public void setStoragePortWwn(java.lang.String storagePortWwn)
```

Should only be used by WSAPI.

Parameters

storagePortWwn - The World Wide Name for the port to which the volume is mapped.

isWwnsecurityvalidity

```
public boolean isWwnsecurityvalidity()
```

Returns

When this flag is true, the system validates the WWNs of the ports accessing a volume to ensure they are masked for that volume. The port WWN is stored in the storagePortWwn member in the VolumeMask class. When this flag is false, the system ignores the WWNs of ports access a volume and therefore eliminates the need for masking.

setWwnsecurityvalidity

```
public void setWwnsecurityvalidity(boolean wwnsecurityvalidity)
```

Parameters

wwnsecurityvalidity - When this flag is true, the system validates the WWNs of the ports accessing a volume to ensure they are masked for that volume. The port WWN is stored in the storagePortWwn member in the VolumeMask class. If set to false, the system ignores the WWNs of ports access a volume and therefor eliminates the need for masking.

VolumeMapResponse

- A partial response to the method call.

Constructors

[Constructors](#) | [Methods](#)

```
public VolumeMapResponse()
```

- Initializes a volume map response object

Methods

[Constructors](#) | [Methods](#)

The following methods are available from this class:

getRequestIterator

```
public RequestIterator getRequestIterator()
```

Gets the iterator which is used by the client to determine if there are more objects to be fetched in addition to the objects in this response, used by the server to determine which objects to return in subsequent invocations.

Returns

The request iterator for this response.

setRequestIterator

```
public void setRequestIterator(RequestIterator iterator)
```

Sets the request iterator to be used with this response. To be used by WSAPI only (for deserializing this response).

Parameters

iterator - The iterator of this response.

getSize

```
public int getSize()
```

Gets the number of objects contained in this response. Note that this is not the total number of objects matching a query.

Returns

Number of objects returned in this response.

setSize

```
public void setSize(int size)
```

Sets the number of objects returned in this response. To be used by WSAPI only (for deserializing the response).

Parameters

size - Number of objects in this response.

getVolumeMaps

```
public VolumeMap[] getVolumeMaps()
```

Returns

An array of volume maps.

setVolumeMaps

```
public void setVolumeMaps(VolumeMap[] volumeMaps)
```

Parameters

volumeMaps - An array of volume maps.

VolumeMask

Adds masking between a volume (disk storage port) and a host.

Constructors

Constructors | Methods

```
public VolumeMask()
```

- Initializes a volume mask object

Methods

Constructors | Methods

The following methods are available from this class:

getId

```
public java.lang.String getId()
```

Gets the unique identifier of this object.

Returns

The unique identifier of this object.

setId

```
public void setId(java.lang.String id)
```

Should only be used by WSAPI. Sets the unique identifier of this object.

Parameters

id - The unique identifier for this object.

getInitiatorPortOrNodeWwn

```
public java.lang.String getInitiatorPortOrNodeWwn()
```

Returns

The host port or adapter to which the storage volume is masked.

setInitiatorPortOrNodeWwn

```
public void setInitiatorPortOrNodeWwn(java.lang.String  
initiatorPortOrNodeWwn)
```

Parameters

initiatorPortOrNodeWwn - The host port or adapter to which the storage volume is masked.

getStoragePortWwn

```
public java.lang.String getStoragePortWwn()
```

Returns

The World Wide Name of the storage port.

setStoragePortWwn

```
public void setStoragePortWwn(java.lang.String storagePortWwn)
```

Parameters

storagePortWwn - The World Wide Name of the storage port.

getVolumeId

```
public java.lang.String getVolumeId()
```

Returns

The ID for the volume to be masked to the host adaptor or port.

setVolumeId

```
public void setVolumeId(java.lang.String volumeId)
```

Parameters

volumeId - The identifier for the volume to be masked to the host adaptor or port.

getStoragePortId

```
public java.lang.String getStoragePortId()
```

Returns

The ID of the storage port from which the volume is accessible to the host initiator.

setStoragePortId

```
public void setStoragePortId(java.lang.String storagePortId)
```

Should be used only by WSAPI.

Parameters

storagePortId - The ID of the storage port from which the volume is accessible to the host initiator.

getProtocolController

```
public java.lang.String getProtocolController()
```

Returns

The protocol controller through which the volume is mapped to the storage port.

setProtocolController

```
public void setProtocolController(java.lang.String  
protocolController)
```

Should be used only by WSAPI.

Parameters

protocolController - The protocol controller through which the volume is mapped to the storage port.

VolumeMaskResponse

A partial response to the method call:

[SANscreenAPIEndPoint](#).[getVolumeMasksByStorageArray](#)([APISessionContext](#),
[RequestIterator](#),

- String containing an array of volume masks.

Constructors

Constructors | Methods

```
public VolumeMaskResponse()
```

- Initializes a volume mask response object

Methods

Constructors | Methods

The following methods are available from this class:

getRequestIterator

```
public RequestIterator getRequestIterator()
```

Gets the iterator which is used by the client to determine if there are more objects to be fetched in addition to the objects in this response, used by the server to determine which objects to return in subsequent invocations.

Returns

The request iterator for this response.

setRequestIterator

```
public void setRequestIterator(RequestIterator iterator)
```

To be used by WSAPI only. Sets the request iterator to be used with this response.

Parameters

iterator - The iterator of this response.

getSize

```
public int getSize()
```

Gets the number of objects contained in this response. Note that this is not the total number of objects matching a query.

Returns

Number of objects returned in this response.

setSize

```
public void setSize(int size)
```

To be used by WSAPI only. Sets the number of objects returned in this response.

Parameters

size - Number of objects in this response.

getMasks

```
public VolumeMask[] getMasks()
```

Returns

The masks known to SANscreen.

setMasks

```
public void setViolations(VolumeMask[] masks)
```

Parameters

masks - The masks known to SANscreen.

VolumeResponse

A partial response to two method calls that retrieve volumes known to SANscreen. The response contains an array of Volume objects. It is returned by the following methods:

```
SANscreenAPIEndPoint.getVolumes(APISessionContext, RequestIterator)  
                                and
```

```
SANscreenAPIEndPoint.getVolumesByStorageArray(APISessionContext,  
                                              RequestIteratorBean) .
```

Constructors

[Constructors | Methods](#)

```
public VolumeResponse()  
    – Initializes a volume response object.
```

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getRequestIterator

```
public RequestIterator getRequestIterator()
```

Gets the iterator which is used by the client to determine if there are more objects to be fetched in addition to the objects in this response, used by the server to determine which objects to return in subsequent invocations.

Returns

The request iterator for this response.

setRequestIterator

```
public void setRequestIterator(RequestIterator iterator)
```

Sets the request iterator to be used with this response. To be used by WSAPI only (for deserializing this response).

Parameters

iterator - The iterator of this response.

getSize

```
public int getSize()
```

Gets the number of objects contained in this response. Note that this is not the total number of objects matching a query.

Returns

Number of objects returned in this response.

setSize

```
public void setSize(int size)
```

Sets the number of objects returned in this response. To be used by WSAPI only (for deserializing the response).

Parameters

size - Number of objects in this response.

getVolumes

```
public Volume[] getVolumes()
```

Returns

An array of volumes.

setVolumes

```
public void setVolumes(Volume[] volumes)
```

Parameters

volumes - An array of volumes.

VolumesSynchronizationState

Constructors

[Constructors | Methods](#)

```
public VolumesSynchronizationState()
```

- Initiates a volume synchronization state object.

Methods

[Constructors | Methods](#)

The following methods are available from this class:

getSourceStorageId

```
public java.lang.String getSourceStorageId()
```

setSourceStorageId

```
public void setSourceStorageId(java.lang.String sourceStorageId)
```

getSourceVolumeId

```
public java.lang.String getSourceVolumeId()
```

setSourceVolumeId

```
public void setSourceVolumeId(java.lang.String sourceVolumeId)
```

getTargetStorageId

```
public java.lang.String getTargetStorageId()
```

setTargetStorageId

```
public void setTargetStorageId(java.lang.String targetStorageId)
```

getTargetVolumeId

```
public java.lang.String getTargetVolumeId()
```

setTargetVolumeId

```
public void setTargetVolumeId(java.lang.String targetVolumeId)
```

getVolumesSynchronizationState

```
public java.lang.String getVolumesSynchronizationState()
```

setVolumesSynchronizationState

```
public void setVolumesSynchronizationState(java.lang.String  
volumesSynchronizationState)
```

getDetailedSynchronizationState

```
public int getDetailedSynchronizationState()
```

setDetailedSynchronizationState

```
public void setDetailedSynchronizationState(int  
detailedSynchronizationState)
```

getTimeStamp

```
public long getTimeStamp()
```

setTimeStamp

```
public void setTimeStamp(long timeStamp)
```

VolumesSynchronizationStateResponse

Constructors

Constructors | Methods

```
public VolumesSynchronizationStateResponse()
```

- Initiates the volumes synchronization state response object

Methods

Constructors | Methods

The following methods are available from this class:

getVolumesSynchronizationState

```
public VolumesSynchronizationState[] getVolumesSynchronizationState()
```

setVolumesSynchronizationState

```
public void setVolumesSynchronizationState(VolumesSynchronizationState[]  
volumesSynchronizationState)
```

Sample Code

- ♦ [Java](#) on page 230
- ♦ [Perl](#) on page 252

Java

The following code provides API Examples in Java.

```
import com.NetApp.sanscreen.wsapi.*;
import javax.xml.rpc.Stub;
import java.rmi.RemoteException;
import java.util.*;

public class APIExamples {

    /* Create a service endpoint interface to the SANscreenAPI bean */
    /* The Web service endpoint is defined in the WSDD XML file */
    private SANscreenAPIEndPoint endPoint;

    /* Initialize the endpoint */
    public APIExamples() {
        endPoint = init();
    }

    /* Print the current API and SANscreen version numbers */
    public void printVersions() throws RemoteException {
        System.out.println("endPoint.getAPIVersion() = " + endPoint.getAPIVersion());
        System.out.println("endPoint.getSANscreenVersion() = " + endPoint.getSANscreenVersion());
    }

    /*-----*/
    /* Retrieve and print attributes for all known storage arrays */
    /*-----*/

    public void getAllStorageArrays() throws APIException, RemoteException {

        /* Open a session context. You must open a session context before querying data */
        APISessionContext context = endPoint.openSession("1.0", 200);

        /* Retrieve all storage arrays */
        StorageArrayResponse response = endPoint.getStorageArrays(context, null);
        /* Create an array to hold all the returned storage arrays */
        StorageArray[] storageArrays = response.getArrays();
        System.out.println("");
        System.out.println("Number of storage arrays returned is: " + storageArrays.length);
    }
}
```

```

        System.out.println("Storage Name          Vendor          Model          Raw
Capacity (GB)");
        System.out.println("-----");
        System.out.println("-----");

        /* Print attributes for each storage array in the array of storage arrays
        */
        for (int i = 0; i < storageArrays.length; i++) {
            StorageArray storageArray = storageArrays[i];
            String vendor = storageArray.getVendor() != null ? storageAr-
ray.getVendor() : "NA";
            String model = storageArray.getModel() != null ? storageArray.get-
Model() : "NA";
            System.out.println(storageArray.getName() + "\t\t" + vendor + "\t"
+ model + "\t" + storageArray.getRawCapacityGB());
        }
    }

    /*-----*/
    /* Retrieve and print attributes for all known paths */
    /*-----*/
    public void getAllPaths() throws APIException, RemoteException {

        /* Open a session context. You must open a session context before querying
        data */
        APISessionContext context = endPoint.openSession("1.0", 10000);

        Map<String, StorageArray> arrayIdToArray = prepareArrayMap(context);
        Map<String, Volume> volumeIdToVolume = prepareVolumeMap(context,
arrayIdToArray.keySet());
        Map<String, Host> hostIdToArray = prepareHostMap(context);
        Map<String, Tape> tapeIdToTape = prepareTapeMap(context);
        List<Path> paths = getPaths(context);
        System.out.println("");
        System.out.println("Number of paths is: " + paths.size());
        System.out.println("Host Name          Storage array name          Volume
name");
        System.out.println("-----");

        for (Iterator<Path> iterator = paths.iterator(); iterator.hasNext();) {
            Path path = iterator.next();
            Host host = hostIdToArray.get(path.getHostId());
            String hostName = "";
            if (host != null)
                hostName = host.getName();

```

```
String targetName = "";
String volumeName = "";
StorageArray storageArray = arrayIdToArray.get(path.getArrayId());
if (storageArray != null) {
    targetName = storageArray.getName();
    Volume volume = volumeIdToVolume.get(path.getVolumeId());
    if (volume != null)
        volumeName = volume.getName();
} else {
    Tape tape = tapeIdToTape.get(path.getArrayId());
    if (tape != null)
        targetName = tape.getName();
}
System.out.println(hostName + "\t\t" + targetName + "\t" + volume-
Name);
    }
}

private Map<String, Tape> prepareTapeMap(APISessionContext context) throws
APIException, RemoteException {
    Map<String, Tape> tapeIdToTape = new HashMap<String, Tape>();
    RequestIterator requestIterator = null;
    do {
        TapeResponse tapeResponse = endPoint.getTapes(context, requestIt-
erator);
        requestIterator = tapeResponse.getRequestIterator();
        Tape[] tapes = tapeResponse.getTapes();
        for (int i = 0; i < tapes.length; i++) {
            Tape tape = tapes[i];
            tapeIdToTape.put(tape.getId(), tape);
        }
    } while (requestIterator.hasMore());
    return tapeIdToTape;
}

private List<Path> getPaths(APISessionContext context) throws APIExcep-
tion, RemoteException {
    List<Path> pathsList = new LinkedList<Path>();
    RequestIterator requestIterator = null;
    do {
        PathResponse pathResponse = endPoint.getPaths(context, requestIt-
erator);
        requestIterator = pathResponse.getRequestIterator();
        Path[] paths = pathResponse.getPaths();
        pathsList.addAll(Arrays.asList(paths));
    } while (requestIterator.hasMore());
}
```

```
        return pathsList;
    }

    private List<Violation> getViolations(APISessionContext context) throws
APIException, RemoteException {
        List<Violation> violationsList = new LinkedList<Violation>();
        RequestIterator requestIterator = null;
        do {
            ViolationResponse violationResponse = endPoint.getViolations(con-
text, requestIterator);
            requestIterator = violationResponse.getRequestIterator();
            Violation[] violations = violationResponse.getViolations();
            violationsList.addAll(Arrays.asList(violations));
        } while (requestIterator.hasMore());
        return violationsList;
    }

    public void getAllViolations() throws APIException, RemoteException {
        APISessionContext context = endPoint.openSession("1.0", 10000);
        List<Violation> violations = getViolations(context);
        System.out.println("");
        System.out.println("Number of violations is: " + violations.size());
        System.out.println("Host id      Storage id      Volume id
Violation type");
        System.out.println("-----");
        --");
        for (Iterator<Violation> iterator = violations.iterator(); itera-
tor.hasNext();) {
            Violation violation = iterator.next();
            System.out.println(violation.getHostId() + "\t\t" + viola-
tion.getArrayId() + "\t" + violation.getVolumeId() + "\t" + violation.get-
Type());
        }
    }

    private List<Policy> getPolicies(APISessionContext context) throws APIEx-
ception, RemoteException {
        List<Policy> policiesList = new LinkedList<Policy>();
        RequestIterator requestIterator = null;
        do {
            PolicyResponse policyResponse = endPoint.getPolicies(context,
requestIterator);
            requestIterator = policyResponse.getRequestIterator();
            Policy[] violations = policyResponse.getPolicies();
            policiesList.addAll(Arrays.asList(violations));
        } while (requestIterator.hasMore());
    }
```

```
        return policiesList;
    }

    public void getAllPolicies() throws APIException, RemoteException {
        APISessionContext context = endPoint.openSession("1.0", 10000);
        List<Policy> policies = getPolicies(context);
        System.out.println("");
        System.out.println("Number of policies is: " + policies.size());
        System.out.println("Host id      Storage id  Volume id  Redundancy  
HopsNumber  HostPortsRedundancy  VolumeSharing  Owner");
        System.out.println("-----");
        --");
        for (Iterator<Policy> iterator = policies.iterator(); iterator.hasNext();) {
            Policy policy = iterator.next();
            System.out.println(policy.getHostId() + "\t\t" + policy.getArrayId() + "\t" + policy.getVolumeId() +
                               "\t" + policy.getRedundancy() + "\t" + policy.getHostPortsRedundancy() +
                               "\t" + policy.getVolumeSharingScope() + "\t" + policy.getOwner());
        }
    }

    private Map<String, Volume> prepareVolumeMap(APISessionContext context,
        Set<String> storageArrayIds) throws APIException, RemoteException {
        Map<String, Volume> volumeIdToVolume = new HashMap<String, Volume>();
        for (Iterator<String> iterator = storageArrayIds.iterator(); iterator.hasNext();) {
            String storageArrayId = iterator.next();
            RequestIterator requestIterator = null;
            do {
                VolumeResponse volumeResponse = endPoint.getVolumesByStorageArray(context, requestIterator, storageArrayId);
                requestIterator = volumeResponse.getRequestIterator();
                Volume[] volumes = volumeResponse.getVolumes();
                for (int i = 0; i < volumes.length; i++) {
                    Volume volume = volumes[i];
                    volumeIdToVolume.put(volume.getId(), volume);
                }
            } while (requestIterator.hasMore());
        }
        return volumeIdToVolume;
    }
}
```

```
private Map<String, Host> prepareHostMap(APISessionContext context) throws
APIException, RemoteException {
    Map<String, Host> hostIdToHost = new HashMap<String, Host>();
    RequestIterator requestIterator = null;
    do {
        HostResponse hostResponse = endPoint.getHosts(context, requestIt-
erator);
        requestIterator = hostResponse.getRequestIterator();
        Host[] hosts = hostResponse.getHosts();
        for (int i = 0; i < hosts.length; i++) {
            Host host = hosts[i];
            hostIdToHost.put(host.getId(), host);
        }
    } while (requestIterator.hasMore());
    return hostIdToHost;
}

private Map<String, StorageArray> prepareArrayMap(APISessionContext con-
text) throws APIException, RemoteException {
    Map<String, StorageArray> arrayIdToArray = new HashMap<String, Stor-
ageArray>();
    RequestIterator requestIterator = null;
    do {
        StorageArrayResponse arrayResponse = endPoint.getStorageAr-
rays(context, requestIterator);
        requestIterator = arrayResponse.getRequestIterator();
        StorageArray[] arrays = arrayResponse.getArrays();
        for (int i = 0; i < arrays.length; i++) {
            StorageArray array = arrays[i];
            arrayIdToArray.put(array.getId(), array);
        }
    } while (requestIterator.hasMore());
    return arrayIdToArray;
}

private SANscreenAPIEndPoint init() {
    //needed for ssl
    System.setProperty("javax.net.ssl.trustStore", "c:\\Program
Files\\SANscreen\\jboss\\server\\NetApp\\cert\\server.keystore");
    System.setProperty("javax.net.ssl.trustStorePass", "changeit");
    String serviceUrl = "http://localhost/sanscreenapi";
    System.setProperty("javax.xml.soap.MessageFactory", "com.sun.xml.mes-
saging.saaj.soap.ver1_1.SOAPMessageFactory1_1Impl");
    SANscreenAPIEndPoint endPoint = new SANscreenAPIService_Impl().getSAN-
screenAPIEndPointPort();
    ((Stub) endPoint)._setProperty(Stub.ENDPOINT_ADDRESS_PROPERTY, servi-
```

```
ceUrl);
    ((Stub) endPoint)._setProperty(Stub.USERNAME_PROPERTY, "admin");
    ((Stub) endPoint)._setProperty(Stub.PASSWORD_PROPERTY, "NetApp");
    return endPoint;
}

public static void main(String[] args) throws RemoteException, APIException {
    APIExamples examples = new APIExamples();
    examples.printVersions();
    examples.getAllStorageArrays();
    examples.getAllPaths();
}
}
```

CSV Example in Java

```
import com.NetApp.sanscreen.wsapi.*;
import com.sun.xml.rpc.client.*;
import javax.net.ssl.*;

public class CSVExample {
    public static void main(String args[]) throws Exception {
        SANscreenAPIEndPoint endpoint = createEndPoint();
        APISessionContext context = endpoint.openSession("1.0", 10000);
        new csv.GetInventory("inventory-api.csv").printReport(context, endpoint);
        new csv.GetConnectedPorts("connected-api.csv").printReport(context, endpoint);
        new csv.GetReplicas("replicas-api.csv").printReport(context, endpoint);
        new csv.GetVolumes("volumes-api.csv").printReport(context, endpoint);
        endpoint.closeSession(context);
        System.out.println("Done");
    }
    protected static SANscreenAPIEndPoint createEndPoint() {
        TrustManager[] trustAllCerts = new TrustManager[]{
            new X509TrustManager() {
                public java.security.cert.X509Certificate[] getAcceptedIssuers() {
                    return null;
                }
                public void checkClientTrusted(
                    java.security.cert.X509Certificate[] certs, String
                    authType) {
                }
                public void checkServerTrusted(
                    java.security.cert.X509Certificate[] certs, String
                    authType) {
                }
            }
        };
        // Install the all-trusting trust manager
        try {
            SSLContext sc = SSLContext.getInstance("SSL");
            sc.init(null, trustAllCerts, new java.security.SecureRandom());
            HTTPSURLConnection.setDefaultSSLSocketFactory(sc.getSocketFactory());
        } catch (Exception e) {
            //this will prevent the server from throwing the following exception:
            //"Couldn't create SOAP message factory due to exception:
            javax.xml.soap.SOAPException: Failed to create MessageFactory:

```

```
org.jboss.axis.soap.MessageFactoryImpl"
    System.setProperty("javax.xml.soap.MessageFactory", "com.sun.xml.mes-
saging.saaj.soap.ver1_1.SOAPMessageFactory1_1Impl");
    SANSscreenAPIEndPoint port = new SANSscreenAPIService_Impl().getSAN-
screenAPIEndPointPort();
    ((StubBase) port)._setProperty(StubPropertyCon-
stants.ENDPOINT_ADDRESS_PROPERTY, "http://localhost:80/sanscreenapi");
    ((StubBase) port)._setProperty(StubPropertyCon-
stants.USERNAME_PROPERTY, "admin");
    ((StubBase) port)._setProperty(StubPropertyCon-
stants.PASSWORD_PROPERTY, "NetApp");
    return port;
}
}
```

CSV Example

```
package csv;
import java.io.*;

public abstract class CsvExample {
    protected BufferedWriter writer;

    public CsvExample(String fileName) throws IOException {
        writer = new BufferedWriter(new FileWriter(fileName));
        System.out.println("writing results to '"+fileName+"'");
    }

    public void addLine(String ... params) throws IOException {
        StringBuilder builder = new StringBuilder();
        for (int i = 0; i < params.length; i++) {
            Object param = params[i];
            if (param != null) {
                builder.append(param);
            }
            if (i+1 < params.length) {
                builder.append('$');
            }
        }
        builder.append('\n');
        writer.write(builder.toString());
    }
}
```

GetConnectedPorts

```
package csv;

import com.NetApp.sanscreen.wsapi.*;
import java.io.*;
import java.util.*;

public class GetConnectedPorts extends CsvExample {

    public GetConnectedPorts(String fileName) throws IOException {
        super(fileName);
    }

    public void printReport(APISessionContext context, SANscreenAPIEndPoint
endPoint) throws IOException, APIException {
        addLine("Initiator", "Target");

        //first put all the ports in a map
        Map<String, LogicalPort> allPorts = new HashMap<String, Logi-
calPort>();
        for(LogicalPort port: endPoint.getConnectedPorts(context, null).get-
LogicalPorts()) {
            allPorts.put(port.getId(), port);
        }

        for(LogicalPort port: allPorts.values()){
            LogicalPort connectedPort = allPorts.get(port.getConnected-
PortId());
            String initiator = port.getWwn();
            String taget = null;
            if (connectedPort != null) {
                taget = connectedPort.getWwn();
            }else{
                System.out.println("could not find port ["+ port.getConnected-
PortId()+"] connected to port ["+port.getId()+"]");
            }
            addLine(initiator,taget);
        }
        writer.flush();
        writer.close();
    }
}
```

GetInventory

```
package csv;

import com.NetApp.sansscreen.wsapi.*;
import java.io.*;
import java.rmi.*;
import java.util.*;

public class GetInventory extends CsvExample {
    private Map<String, Set<Port>> portsOfNodes = new HashMap<String,
Set<Port>>();
    private Map<String, Set<Port>> portsOfDevices = new HashMap<String,
Set<Port>>();

    public GetInventory(String fileName) throws IOException {
        super(fileName);
    }

    public void printReport(APISessionContext context, SANscreenAPIEndPoint
endPoint) throws APIException, IOException {
        calcPortsMap(endPoint, context);

        addLine("Alias", "Identifier", "DeviceType", "WWNN", "WWPN", "Port-
Name");

        for(StorageArray array:endPoint.getStorageArrays(context, null).getAr-
rays()) {
            addLines(array.getId(), array.getName(), array.getIp(), "Storage",
endPoint, context, true);
        }

        for(Tape tape :endPoint.getTapes(context, null).getTapes()) {
            addLines(tape.getId(), tape.getName(), tape.getIp(), "Tape", end-
Point, context, true);
        }

        for (Host host : endPoint.getHosts(context, null).getHosts()) {
            addLines(host.getId(), host.getName(), host.getIp(), "Host", end-
Point, context, true);
        }

        for (Switch aSwitch : endPoint.getSwitches(context,
null).getSwitches()) {
            addLines( aSwitch.getId(), aSwitch.getName(), aSwitch.getWwn(),
"Switch", endPoint, context, false);
        }
    }
}
```

```

        for (Generic generic : endPoint.getGenerics(context, null).getGenerics()) {
            addLines( generic.getId(), generic.getWwn(), generic.getWwn(),
"Generic Device", endPoint, context, false);
        }

        writer.flush();
        writer.close();
    }

    private void calcPortsMap(SANscreenAPIEndPoint endPoint, APISessionContext
context) throws APIException, RemoteException {
        RequestIterator requestIterator = null;
        while (requestIterator == null || requestIterator.isHasMore()) {
            PortResponse portResponse = endPoint.getPorts(context, requestIt-
erator);
            for(Port port: portResponse.getPorts()){
                String nodeId = port.getNodeId();
                if(nodeId != null){
                    addPort(portsOfNodes, nodeId, port);
                }else{
                    addPort(portsOfDevices, port.getDeviceId(), port);
                }
            }
            requestIterator = portResponse.getRequestIterator();
        }
    }

    private void addPort(Map<String, Set<Port>> map, String nodeOrDeviceId,
Port port) {
        Set<Port> ports = map.get(nodeOrDeviceId);
        if(ports == null){
            ports = new HashSet<Port>();
            map.put(nodeOrDeviceId, ports);
        }
        ports.add(port);
    }

    private void addLines(String deviceId, String deviceAlias, String device-
Identifier, String deviceType, SANscreenAPIEndPoint endPoint, APISessionCon-
text context, boolean isNodeContainer) throws APIException, IOException {
        if(isNodeContainer) {
            NodeResponse nodesOfDevice = endPoint.getNodesOfDevice(context,
null, deviceId);
            for(Node node: nodesOfDevice.getNodes()) {
                for (Port port : portsOfNodes.get(node.getId())) {

```

```
        addLine(deviceAlias, deviceIdentifier, deviceType,
node.getWwn(), port.getWwn(), port.getName());
    }
}
} else {
    for (Port port : portsOfDevices.get(deviceId)) {
        addLine(deviceAlias, deviceIdentifier, deviceType, deviceIdentifier, port.getWwn(), port.getName());
    }
}
}
}
```

GetReplicas

```
package csv;

import com.NetApp.sansscreen.wsapi.*;
import java.io.*;
import java.util.*;

public class GetReplicas extends CsvExample {
    public GetReplicas(String fileName) throws IOException {
        super(fileName);
    }

    public void printReport(APISessionContext context, SANscreenAPIEndPoint
endPoint) throws APIException, IOException {
        addLine("InitiatorStorageIdentifier", "InitiatorStorageName", "Initia-
torStorageSerial", "InitiatorVolume", "TargetStorageIdentifier", "TargetStora-
geName", "TargetStorageSerial", "TargetVolume");

        Map<String, StorageArray> allStorages = new HashMap<String, StorageAr-
ray>();
        for(StorageArray storage: endPoint.getStorageArrays(con-
text, null).getArrays()) {
            allStorages.put(storage.getId(), storage);
        }
        HashSet<SyncInfo> syncHistory = new HashSet<SyncInfo>();
        for(StorageArray storage: allStorages.values()) {
            for(Synchronization sync: endPoint.getSynchronizationsByStorageAr-
ray(context, null, storage.getId()).getSynchronizations()){
                String sourceIP = sync.getSourceStorageArrayIP();
                String sourceName = getName(allStorages, sync.getSourceStorage-
ArrayId());
                String sourceSerial = getSerial(allStorages, sync.getSourceS-
torageArrayId());
                String sourceVolume = sync.getSourceVolumeName();
                String targetIP = sync.getTargetStorageArrayIP();
                String targetName = getName(allStorages, sync.getTargetStorage-
ArrayId());
                String targetSerial = getSerial(allStorages, sync.getTar-
getStorageArrayId());
                String targetVolume = sync.getTargetVolumeName();

                if (notRedundant(syncHistory, sourceIP, sourceVolume, tar-
getIP, targetVolume)) {
                    addLine(sourceIP, sourceName, sourceSerial, sourceVolume, targ-
etIP, targetName, targetSerial, targetVolume);
                }
            }
        }
    }
}
```

```

    }
    writer.flush();
    writer.close();
}

private boolean notRedundant(Set<SyncInfo> syncHistory, String sourceIP,
String sourceVolume, String targetIP, String targetVolume) {
    return syncHistory.add(new SyncInfo(sourceIP, sourceVolume, targetIP,
targetVolume));
}

private String getName(Map<String, StorageArray> allStorages, String stor-
ageArrayId) {
    StorageArray storageArray = allStorages.get(storageArrayId);
    return storageArray != null ? storageArray.getName() : "";
}

private String getSerial(Map<String, StorageArray> allStorages, String
storageArrayId) {
    StorageArray storageArray = allStorages.get(storageArrayId);
    return storageArray != null ? storageArray.getSerialNumber() : "";
}

private static class SyncInfo {
    private String sourceIP;
    private String targetIP;
    private String sourceVolume;
    private String targetVolume;

    public SyncInfo(String sourceIP, String sourceVolume, String targetIP,
String targetVolume) {
        this.sourceIP = sourceIP;
        this.sourceVolume = sourceVolume;
        this.targetIP = targetIP;
        this.targetVolume = targetVolume;
    }

    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;

        final SyncInfo syncInfo = (SyncInfo) o;

        if (sourceIP != null ? !sourceIP.equals(syncInfo.sourceIP) : sync-
Info.sourceIP != null) return false;
        if (sourceVolume != null ? !sourceVolume.equals(sync-
Info.sourceVolume) : syncInfo.sourceVolume != null)
            return false;
    }
}

```

```
        if (targetIP != null ? !targetIP.equals(syncInfo.targetIP) : sync-
Info.targetIP != null) return false;
        if (targetVolume != null ? !targetVolume.equals(syncInfo.tar-
getVolume) : syncInfo.targetVolume != null)
            return false;
        return true;
    }

    public int hashCode() {
        int result;
        result = (sourceIP != null ? sourceIP.hashCode() : 0);
        result = 29 * result + (targetIP != null ? targetIP.hashCode() :
0);
        result = 29 * result + (sourceVolume != null ? sourceVolume.hash-
Code() : 0);
        result = 29 * result + (targetVolume != null ? targetVolume.hash-
Code() : 0);
        return result;
    }
}
```

GetVolumes

```
package csv;

import com.NetApp.sanscreen.wsapi.*;
import java.io.*;
import java.text.*;
import java.util.*;
import java.rmi.*;

public class GetVolumes extends CsvExample {
    public GetVolumes(String fileName) throws IOException {
        super(fileName);
    }

    public void printReport(APISessionContext context, SANscreenAPIEndPoint
endPoint) throws APIException, IOException {
        NumberFormat format = NumberFormat.getNumberInstance();
        format.setMaximumFractionDigits(2);

        Map<String, List<String>> map = calcVolumeIdToHostId(endPoint, con-
text);
        Map<String, WwnSet> hostsPorts = calcHostsPortsList(endPoint, con-
text);

        addLine("ArrayName", "Volume", "Capacity", "Type", "Redundancy",
"WWNs");
        Map<StorageArray, List<Volume>> storages = calcStorageVolumesMap(end-
Point, context);
        for (Map.Entry<StorageArray, List<Volume>> storage: storages.entry-
Set()){
            String ArrayName = storage.getKey().getName();

            for (Volume volume: storage.getValue()) {
                String Volume = volume.getName();
                String Capacity = format.format(volume.getCapacityGB());
                String Type = getType(volume);
                String Redundancy = getRedundancy(volume);

                List<String> hostIds = map.get(volume.getId());
                if (hostIds != null) {
                    for (String hostId : hostIds) {
                        WwnSet wwnSet = hostsPorts.get(hostId);
                        if (wwnSet != null) {
                            addLine(ArrayName, Volume, Capacity, Type, Redun-
dancy, wwnSet.getWwnsList());
                        }
                    }
                }
            }
        }
    }
}
```

```

        }else{
            //host is Unknown
            addLine(ArrayName, Volume, Capacity, Type, Redun-
dancy, "");
        }
    }
} else {
    addLine(ArrayName, Volume, Capacity, Type, Redundancy,
"");
}
}
}

writer.flush();
writer.close();

}

private List<Path> getAllPaths(SANscreenAPIEndPoint endPoint, APISession-
Context context) throws APIException, RemoteException {
    LinkedList<Path> list = new LinkedList<Path>();
    RequestIterator requestIterator = null;
    while(requestIterator == null || requestIterator.isHasMore()){
        PathResponse pathResponse = endPoint.getPathResponse(context, requestIt-
erator);
        list.addAll(Arrays.asList(pathResponse.getPathResponse()));
        requestIterator = pathResponse.getRequestIterator();
    }
    return list;
}

private Map<StorageArray, List<Volume>> calcStorageVolumesMap(SAN-
screenAPIEndPoint endPoint, APISessionContext context) throws APIException,
RemoteException {
    Map<String, StorageArray> storages = new HashMap<String, StorageAr-
ray>();
    for(StorageArray storageArray: endPoint.getStorageArrays(context,
null).getArrays()){
        storages.put(storageArray.getId(), storageArray);
    }

    HashMap<StorageArray, List<Volume>> map = new HashMap<StorageArray,
List<Volume>>();
    for(Volume volume: getAllVolumes(endPoint, context)){
        String storageID = volume.getStorageID();
        StorageArray storageArray = storages.get(storageID);

```

```

        List<Volume> volumes = map.get(storageArray);
        if(volumes == null){
            volumes = new ArrayList<Volume>();
            map.put(storageArray, volumes);
        }
        volumes.add(volume);
    }
    return map;
}

private List<Volume> getAllVolumes(SANscreenAPIEndPoint endPoint, APISessionContext context) throws APIException, RemoteException {
    LinkedList<Volume> list = new LinkedList<Volume>();
    RequestIterator requestIterator = null;
    while(requestIterator == null || requestIterator.isHasMore()){
        VolumeResponse pathResponse = endPoint.getVolumes(context, requestIterator);
        list.addAll(Arrays.asList(pathResponse.getVolumes()));
        requestIterator = pathResponse.getRequestIterator();
    }
    return list;
}

private static String getType(Volume volume) {
    String type = volume.getType();
    if (type != null && type.length() > 0) {
        //looking for comma to remove the ", Meta" and leave only the Volume R1 or R2
        int indexOfComma = type.indexOf(',');
        return indexOfComma != -1 ? type.substring(0, indexOfComma) :
type;
    } else {
        return "N/A";
    }
}

private static String getRedundancy(Volume volume) {
    String redundancy = volume.getRedundancy();
    if (redundancy != null && redundancy.length() > 0) {
        String rv = redundancy.replace(", ", ";");
        if (rv.startsWith("[") ) {
            rv = rv.substring(1);
        }
        if (rv.endsWith("]")) {
            rv = rv.substring(0, rv.length() - 1);
        }
        return rv;
    }
}

```

```
    } else {  
        return "N/A";  
    }  
}
```

```
private Map<String, List<String>> calcVolumeIdToHostId(SANscreenAPIEndPoint endPoint, APISessionContext context) throws APIException, RemoteException {  
    List<Path> paths = getAllPaths(endPoint, context);  
    HashMap<String, List<String>> map = new HashMap<String, List<String>>();  
    for(Path path: paths){  
        String volumeId = path.getVolumeId();  
        List<String> hosts = map.get(volumeId);  
        if(hosts == null){  
            hosts = new ArrayList<String>();  
            map.put(volumeId, hosts);  
        }  
        hosts.add(path.getHostId());  
    }  
    return map;  
}
```

```
private Map<String, WwnSet> calcHostsPortsList(SANscreenAPIEndPoint endPoint, APISessionContext context) throws APIException, RemoteException {  
    HashMap<String, WwnSet> map = new HashMap<String, WwnSet>();  
  
    for(Port port: getAllPortsOfHost(endPoint, context)){  
        String wwn = port.getWwn();  
        String hostId = port.getDeviceId();  
        WwnSet wwnSet = map.get(hostId);  
        if(wwnSet == null){  
            wwnSet = new WwnSet();  
            map.put(hostId, wwnSet);  
        }  
        wwnSet.add(wwn);  
    }  
    return map;  
}
```

```
private LinkedList<Port> getAllPortsOfHost(SANscreenAPIEndPoint endPoint, APISessionContext context) throws APIException, RemoteException {  
    LinkedList<Port> list = new LinkedList<Port>();  
    RequestIterator requestIterator = null;  
    while (requestIterator == null || requestIterator.isHasMore()) {
```

```

        PortResponse portResponse = endPoint.getPortsOfDeviceType(context,
requestIterator, "Host");
        list.addAll(Arrays.asList(portResponse.getPorts()));
        requestIterator = portResponse.getRequestIterator();
    }
    return list;
}

private static class WwnSet extends TreeSet<String>{
    private String wwnsList = null;
    public String getWwnsList() {
        if(wwnsList== null){
            StringBuilder wwnsbuilder = new StringBuilder();
            for (Iterator<String> wwnIterator = iterator(); wwnItera-
tor.hasNext();) {
                wwnsbuilder.append(wwnIterator.next());
                if (wwnIterator.hasNext()) {
                    wwnsbuilder.append(';');
                }
            }
            wwnsList = wwnsbuilder.toString();
        }
        return wwnsList;
    }
}
}
}

```

Perl

Note

Perl examples are for reference purposes only and are not actively supported by NetApp.

APISessionContext Package

```
package APISessionContext;

use warnings;
use strict;
use SOAPParam;
our @ISA = qw(SOAPParam);

####

# APISessionContext

# I've decided to model this as one SOAPParam containing several child
# SOAP::Data objects. If you parse the results from a connect call,
# it will

# fill in the objects as necessary. I've overloaded
SOAPParam::getSoap() to

# drop these child objects in the value for the SOAP::Data object
right

# before deserializing.

sub new
{
    my ( $class ) = @_;

    my $self = $class->SUPER::new(
        "APISessionContext_1",
        "tns:APISessionContext");

    # Build a temporary hash of parameter name to SOAP::Data objects to
    aid
```

```
$self->{_childHash} = ();
$self->{_childHash}->{"clientVersion"} = new SOAP::Data(
    type => "string",
    name => "clientVersion");

    $self->{_childHash}->{"id"} = new SOAP::Data(
    type => "string",
    name => "id");

$self->{_childHash}->{"limit"} = new SOAP::Data(
    type => "int",
    name => "limit");

$self->{_childHash}->{"time"} = new SOAP::Data(
    type => "long",
    name => "time");

bless $self, ref($class) || $class;
return $self;
}

# parseResult
# Takes a SOAP::SOM data representation. An object that can be
# searched using
# XPath. We will parse the results of a openSession call so that we
# can reuse
# this context object in subsequent calls
sub parseResult
{
    my ($self, $som) = @_;

    $self->clientVersion($som->valueof("//result/clientVersion"));
```



```
$self->id($som->valueof("//result/id"));
$self->limit($som->valueof("//result/limit"));
$self->time($som->valueof("//result/time"));
}

# clientVersion
# The getter and setter method for _childHash->{"clientVersion"}
SOAP::Data
# object
sub clientVersion
{
my ($self, $clientVersion) = @_;
    $self->{_childHash}->{"clientVersion"}->value($clientVersion) if
defined $clientVersion;
return $self->{_childHash}->{"clientVersion"}->value();
}

# id
# The getter and setter method for _childHash->{"id"} SOAP::Data
# object
sub id
{
my ($self, $id) = @_;
    $self->{_childHash}->{"id"}->value($id) if defined $id;
return $self->{_childHash}->{"id"}->value();
}

# limit
# The getter and setter method for _childHash->{"limit"} SOAP::Data
# object
sub limit
{
my ($self, $limit) = @_;
    $self->{_childHash}->{"limit"}->value($limit) if defined $limit;
return $self->{_childHash}->{"limit"}->value();
}
```

```
# time
# The getter and setter method for _childHash->{"time"} SOAP::Data
# object
sub time
{
my ($self, $time) = @_;
    $self->{_childHash}->{"time"}->value($time) if defined $time;
return $self->{_childHash}->{"time"}->value();
}

# getSoap
# The overloaded function that drops the values of our childHash, in
# this case
# SOAP::Data objects into the encapsulating SOAP::Data object. The
# SOAP xml
# representation will be <Parent><Child1>value<Child2
sub getSOAP
{
my ( $self ) = @_;
my @valueList = values %{$self->{_childHash}};

$self->value(\@valueList);

return $self->SUPER::getSOAP();
}
1;
```

Iterator Package

```

package Iterator;
use warnings;
use strict;
use SOAPParam;
our @ISA = qw(SOAPParam);
sub new
{
my ( $class ) = @_;

my $self = $class->SUPER::new(
    "RequestIterator_2",
    "tns:RequestIterator");

$self->{_childHash} = ();
$self->{_childHash}->{"hasMore"} = new SOAP::Data(
    type => "boolean",
    name => "hasMore");

$self->{_childHash}->{"nextKey"} = new SOAP::Data(
    type => "string",
    name => "nextKey");

$self->{_childHash}->{"methodName"} = new SOAP::Data(
    type => "string",
    name => "methodName");

bless $self, ref($class) || $class;
return $self;
}

```

```

sub parseResult
{
my ($self, $som) = @_;

my $retValue = new Iterator();

$retValue->hasMore($som->valueof("//result/requestIterator/
hasMore"));

$retValue->nextKey($som->valueof("//result/requestIterator/
nextKey"));

$retValue->methodName($som->valueof("//result/requestIterator/
methodName"));

return $retValue;
}

sub methodName
{
my ( $self, $methodName) = @_;

$self->{_childHash}->{"methodName"}->value($methodName) if defined
$methodName;

return $self->{_childHash}->{"methodName"}->value();
}

sub hasMore
{
my ( $self, $hasMore) = @_;

$self->{_childHash}->{"hasMore"}->value($hasMore) if defined
$hasMore;

return $self->{_childHash}->{"hasMore"}->value();
}

sub nextKey
{
my ( $self, $nextKey) = @_;

$self->{_childHash}->{"nextKey"}->value($nextKey) if defined
$nextKey;

return $self->{_childHash}->{"nextKey"}->value();
}

```

```
}

# getSoap

# The overloaded function that drops the values of our childHash, in
# this case

# SOAP::Data objects into the encapsulating SOAP::Data object. The
# SOAP xml
# representation will be <Parent><Child1>value<Child2

sub getSOAP
{
my ( $self ) = @_;
my @valueList = values %{$self->{_childHash}};

$self->value(\@valueList);

return $self->SUPER::getSOAP();
}

1;
```

APIVersion Package

```
package APIVersion;

use warnings;
use strict;
use SOAPParam;

our @ISA = qw(SOAPParam);

sub new
{
    my ( $class, $apiVersion ) = @_;

    my $self = $class->SUPER::new(
        "String_1",
        "string",
        $apiVersion);

    $self->{_apiVersion} = $apiVersion;

    bless $self, ref($class) || $class;
    return $self;
}

sub apiVersion {
    my ( $self, $apiVersion ) = @_;
    if(defined $apiVersion)
    {
        $self->{_apiVersion} = $apiVersion;
        $self->value($apiVersion);
    }

    return $self->{_apiVersion};
}

1;
```

Sample Scripts

- ♦ [Computing Total Storage Per Array](#) on page 261
- ♦ [Finding Disconnected Hosts](#) on page 265
- ♦ [Get Paths](#) on page 269
- ♦ [Get Storage](#) on page 272
- ♦ [Host Package](#) on page 275
- ♦ [Iterator Package](#) on page 256
- ♦ [Path Package](#) on page 278
- ♦ [Perl Examples Requirements](#) on page 281
- ♦ [SOAPParam Package](#) on page 282
- ♦ [StorageArray Package](#) on page 285
- ♦ [Timeout Package](#) on page 284
- ♦ [Volume Package](#) on page 290

Computing Total Storage Per Array

```
# Note that you would never want to actually do this, as storages have
a

# capacity attribute on them. But for the purposes of example, the
script below

# will look up all the volume on storage devices and figure out the
total amount

# of capacity available.

#!/usr/bin/perl -w

use warnings;
use strict;
use SOAP::Lite;
use APIVersion;
use Timeout;
use APISessionContext;
use StorageArray;
use Iterator;
use Volume;

# This is to get standard output to flush on write.
$| = 1;

# First instantiate the SOAPLite object.
# Provide the namespace, which in our case is:
#      http://wsapi.sansscreen.NetApp.com/SANscreenAPI
# and the actual server to connect to: https://<yourserver>/
sansscreenapi.

# Note that using the https in the URL requires some version of SSL
installed on

# the system. On windows boxes this can be supplied by using SSLeay
my $Soap= new SOAP::Lite
ns => 'http://wsapi.sansscreen.NetApp.com/SANscreenAPI',
proxy => 'https://localhost/sansscreenapi',
```



```
on_debug => sub{print@_;},
on_fault => sub { my($soap, $res) = @_;
    die ref $res ? $res->faultstring : $soap->transport->status,
    "\n";
};

# We want to 'overload' getting the login information. It expects a
# map with a
# name that points to a password
sub SOAP::Transport::HTTP::Client::get_basic_credentials {
    return 'guest' => '';
}

# SOAPLite uses SOAP::Data objects to build up the request. Included
# in the
# examples directory will be a copy of the APIVersion and Timeout
# objects.
# This simplifies the instantiation of the SOAP::Data objects
my $SOAPApiVersion = new APIVersion("1.0")->getSOAP();
my $SOAPTimeOut = new Timeout(1000)->getSOAP();
# Now our first request.
my $somResponse = $Soap->openSession($SOAPApiVersion,
    $SOAPTimeOut);

# Our response, the "somResponse" is stored in SOAP::Som format. This
# object is an object you can feed XPath queries into to get the
# actual
# contents of the response. To simplify the parsing of this XML, im
# going to
# feed it into the parse function of an APISessionContext object, also
# found in the examples directory. This context is what we're going to
# be passing a
# round to let the server know which session we are.
my $APIContext = new APISessionContext();
$APIContext->parseResult($somResponse);
# We can print out the values of the API Context after parse, just to
# signify
# a succesful connection
```

```
print("Client Version: " . $APIContext->clientVersion() . "\n");
print("ID: " . $APIContext->id() . "\n");
print("Limit: " . $APIContext->limit() . "\n");
print("Time: " . $APIContext->time() . "\n\n\n");
my @allStorageObjects;

# We need to build an empty request iterator to seed the call
my $iterObject = new SOAPParam( "RequestIterator_2",
    "tns:RequestIterator");

do
{
    my $storageArraysResponse = $Soap->getStorageArrays($APIContext->
        >getSOAP(), $iterObject->getSOAP());
    $iterObject = Iterator->parseResult($storageArraysResponse);
    my @currStorageArrayObjects = StorageArray->
        >parseResult($storageArraysResponse);

    push @allStorageObjects, @currStorageArrayObjects;
} while($iterObject->hasMore() eq "true");
foreach my $storageArrayObject (@allStorageObjects)
{
    # We need to build an empty request iterator to seed the call
    $iterObject = new SOAPParam( "RequestIterator_2",
        "tns:RequestIterator");
    print $storageArrayObject->name() . "\n\n";
    my $size = 0;
    my @allVolumeObjects;
    do
    {
        my $arrayRequester = new SOAP::Data(
            value => $storageArrayObject->id(),
            name => "String_3",
            type => "string");
        my $volumeResponse = $Soap->getVolumesByStorageArray($APIContext->
            >getSOAP(), $iterObject->getSOAP(), $arrayRequester );
```

```
$iterObject = Iterator->parseResult($volumeResponse);  
my @currVolumeObjects= Volume->parseResult($volumeResponse);  
  
push @allVolumeObjects, @currVolumeObjects;  
  
} while($iterObject->hasMore() eq "true");  
  
foreach my $volume (@allVolumeObjects){  
    $size += $volume->capacityGB();  
}  
  
print "Size in GB as returned by summing the volume sizes: ". $size .  
"\n\n\n";  
  
}
```

Finding Disconnected Hosts

```
# This script is an example of how to get hosts that do not have any
paths
# associated with them.
#!/usr/bin/perl -w
use warnings;
use strict;
use SOAP::Lite;
use APIVersion;
use Timeout;
use APISessionContext;
use Iterator;
use Path;
use Host;

# This is perl deep magic to get standard output to flush on write.
$| = 1;

# First instantiate the SOAPLite object.
# Provide the namespace, which in our case is:
#      http://wsapi.sanscreen.NetApp.com/SANscreenAPI
# and the actual server to connect to: https://<yourserver>/
# sansscreenapi.
# Note that using the https in the URL requires some version of SSL
# installed on
# the system. On windows boxes this can be supplied by using SSLeay
my $Soap= new SOAP::Lite
ns => 'http://wsapi.sanscreen.NetApp.com/SANscreenAPI',
proxy => 'https://localhost/sansscreenapi',
on_debug => sub{print@_;},
on_fault => sub { my($soap, $res) = @_;
    die ref $res ? $res->faultstring : $soap->transport->status,
    "\n";
```

```
};

# We want to 'overload' getting the login information. It expects a
map with a

# name that points to a password

sub SOAP::Transport::HTTP::Client::get_basic_credentials {

    return 'guest' => '';

}

# SOAPLite uses SOAP::Data objects to build up the request. Included
in the

# examples directory will be a copy of the APIVersion and Timeout
objects.

# This simplifies the instantiation of the SOAP::Data objects
my $SOAPApiVersion = new APIVersion("1.0")->getSOAP();
my $SOAPTimeOut = new Timeout(1000)->getSOAP();

# Now our first request.
my $somResponse = $Soap->openSession($SOAPApiVersion,
    $SOAPTimeOut);

# Our response, the "somResponse" is stored in SOAP::Som format. This
# object is an object you can feed XPath queries into to get the
actual

# contents of the response. To simplify the parsing of this XML, im
going to

# feed it into the parse function of an APISessionContext object, also
# found in the examples directory. This context is what we're going to
be passing a

# round to let the server know which session we are.
my $APIContext = new APISessionContext();
$APIContext->parseResult($somResponse);

# We can print out the values of the API Context after parse, just to
signify

# a succesful connection
print("Client Version: " . $APIContext->clientVersion() . "\n");
print("ID: " . $APIContext->id() . "\n");
print("Limit: " . $APIContext->limit() . "\n");
```

```
print("Time: " . $APIContext->time() . "\n\n\n");
my %allHostsInPaths;

# We need to build an empty request iterator to seed the call
my $iterObject = new SOAPParam( "RequestIterator_2",
    "tns:RequestIterator");

# First we retrieve all the path objects
my (@currPathObjects, @allPathObjects);
print("Finding paths...\n");
do
{
    my $pathResponse = $Soap->getPaths($APIContext->getSOAP(),
        $iterObject->getSOAP());
    $iterObject = Iterator->parseResult($pathResponse);
    my @currPathObjects = Path->parseResult($pathResponse);

    push @allPathObjects, @currPathObjects;
    print(".");
} while($iterObject->hasMore() eq "true");
print("\n");

# Now go through the pats we retrieved and put them into a hash
object, there
# by eliminating duplicates
for my $pathObject (@allPathObjects)
{
    $allHostsInPaths{$pathObject->hostId()} = 1;
}

# Now retrieve all the hosts. If a host is not in our hash, then it is
# disconnected
my (@currHostObjects, @allHostObjects);
$iterObject = new SOAPParam( "RequestIterator_2",
    "tns:RequestIterator");
print("Finding all hosts...\n");
do
```

```
{
my $hostResponse = $Soap->getHosts($APIContext->getSOAP(),
    $iterObject->getSOAP());
$iterObject = Iterator->parseResult($hostResponse);
my @currHostObjects = Host->parseResult($hostResponse);

push @allHostObjects, @currHostObjects;
print(".");
} while($iterObject->hasMore() eq "true");
print("\n");
my @disconnectedHosts;
for my $hostObject(@allHostObjects)
{
    if(!defined $allHostsInPaths{$hostObject->id()})
    {
        push @disconnectedHosts, $hostObject;
        $hostObject->dump();
        print("\n\n");
    }
}
```

Get Paths

```
#!/usr/bin/perl -w
use warnings;
use strict;
use SOAP::Lite;
use APIVersion;
use Timeout;
use APISessionContext;
use StorageArray;
use Iterator;
use Path;

# This is perl deep magic to get standard output to flush on write.
$| = 1;

# First instantiate the SOAPLite object.
# Provide the namespace, which in our case is:
#      http://wsapi.sanscreen.NetApp.com/SANscreenAPI
# and the actual server to connect to: https://<yourserver>/
# sansscreenapi.
# Note that using the https in the URL requires some version of SSL
# installed on
# the system. On windows boxes this can be supplied by using SSLeay
my $Soap= new SOAP::Lite
ns => 'http://wsapi.sanscreen.NetApp.com/SANscreenAPI',
proxy => 'https://localhost/sansscreenapi',
on_fault => sub { my($soap, $res) = @_;
    die ref $res ? $res->faultstring : $soap->transport->status,
    "\n";
};

# We want to 'overload' getting the login information. It expects a
# map with a
# name that points to a password
sub SOAP::Transport::HTTP::Client::get_basic_credentials {
```



```
    return 'guest' => '';
}

# SOAPLite uses SOAP::Data objects to build up the request. Included
in the

# examples directory will be a copy of the APIVersion and Timeout
objects.

# This simplifies the instantiation of the SOAP::Data objects
my $SOAPApiVersion = new APIVersion("1.0")->getSOAP();
my $SOAPTimeOut = new Timeout(1000)->getSOAP();

# Now our first request.
my $somResponse = $Soap->openSession($SOAPApiVersion,
    $SOAPTimeOut);

# Our response, the "somResponse" is stored in SOAP::Som format. This
# object is an object you can feed XPath queries into to get the
actual

# contents of the response. To simplify the parsing of this XML, im
going to

# feed it into the parse function of an APISessionContext object, also
# found in the examples directory. This context is what we're going to
be passing a

# round to let the server know which session we are.
my $APIContext = new APISessionContext();
$APIContext->parseResult($somResponse);

# We can print out the values of the API Context after parse, just to
signify

# a succesful connection
print("Client Version: " . $APIContext->clientVersion() . "\n");
print("ID: " . $APIContext->id() . "\n");
print("Limit: " . $APIContext->limit() . "\n");
print("Time: " . $APIContext->time() . "\n\n\n");

# This shows how to retrieve the paths arrays that SANSscreen knows
about

# normally the second command of this function will be an iterator
object,

# but we'll just retrieve all of them in one batch now.
```

```
my (@currPathObjects, @allPathObjects);

# We need to build an empty request iterator to seed the call
my $iterObject = new SOAPParam( "RequestIterator_2",
    "tns:RequestIterator");

do
{
    my $pathResponse = $Soap->getPaths($APIContext->getSOAP(),
        $iterObject->getSOAP());
    $iterObject = Iterator->parseResult($pathResponse);
    my @currPathObjects = Path->parseResult($pathResponse);

    push @allPathObjects, @currPathObjects;
    print(".");
} while($iterObject->hasMore() eq "true");

# Now print them out
for my $pathObject (@allPathObjects)
{
    $pathObject->dump();
    print("\n\n");
}
```

Get Storage

```
#!/usr/bin/perl -w
use warnings;
use strict;
use SOAP::Lite;
use APIVersion;
use Timeout;
use APISessionContext;
use StorageArray;

# This is perl deep magic to get standard output to flush on write.
$| = 1;

# First instantiate the SOAPLite object.
# Provide the namespace, which in our case is:
#      http://wsapi.sanscreen.NetApp.com/SANscreenAPI
# and the actual server to connect to: https://<yourserver>/
# sansscreenapi.
# Note that using the https in the URL requires some version of SSL
# installed on
# the system. On windows boxes this can be supplied by using SSLeay
my $Soap= new SOAP::Lite
ns => 'http://wsapi.sanscreen.NetApp.com/SANscreenAPI',
proxy => 'https://localhost/sansscreenapi',
on_fault => sub { my($soap, $res) = @_;
    die ref $res ? $res->faultstring : $soap->transport->status,
    "\n";
};

# We want to 'overload' getting the login information. It expects a
# map with a
# name that points to a password
sub SOAP::Transport::HTTP::Client::get_basic_credentials {
    return 'guest' => '';
```

```
}

# SOAPLite uses SOAP::Data objects to build up the request. Included
in the

# examples directory will be a copy of the APIVersion and Timeout
objects.

# This simplifies the instantiation of the SOAP::Data objects
my $SOAPApiVersion = new APIVersion("1.0")->getSOAP();
my $SOAPTimeOut = new Timeout(1000)->getSOAP();

# Now our first request.
my $somResponse = $Soap->openSession($SOAPApiVersion,
    $SOAPTimeOut);

# Our response, the "somResponse" is stored in SOAP::Som format. This
# object is an object you can feed XPath queries into to get the
actual

# contents of the response. To simplify the parsing of this XML, im
going to

# feed it into the parse function of an APISessionContext object, also
# found in the examples directory. This context is what we're going to
be passing a

# round to let the server know which session we are.
my $APIContext = new APISessionContext();
$APIContext->parseResult($somResponse);

# We can print out the values of the API Context after parse, just to
signify

# a succesful connection
print("Client Version: " . $APIContext->clientVersion() . "\n");
print("ID: " . $APIContext->id() . "\n");
print("Limit: " . $APIContext->limit() . "\n");
print("Time: " . $APIContext->time() . "\n\n\n");

# This shows how to retrieve the storages arrays that SANScreen knows
about

# normally the second command of this function will be an iterator
object,

# but we'll just retrieve all of them in one batch now.
```

```
my $storageArraysResponse = $Soap->getStorageArrays($APIContext->getSOAP());

my @storageArrayObjects = StorageArray->parseResult($storageArraysResponse);

# Now print them out
for my $storageArrayObject (@storageArrayObjects)
{
    $storageArrayObject->dump();
    print("\n\n");
}
```

Host Package

```
package Host;
use warnings;
use strict;
sub new
{
my ( $class,
    $id,
        $dead,
        $ip,
        $name )= @_;

    my $self = {
        _id => $id,
        _dead => $dead,
        _ip => $ip,
        _name => $name};

    bless $self, ref($class) || $class;
    return $self;
}
# This method is intended to be called statically to retrieve the
sub parseResult
{
my ($self, $somResponse) = @_;
my @retValue = ();

if($somResponse->valueof("//result/size") eq "0")
{
return @retValue;
}
```

```
my @hostResponses = $somResponse->valueof("//result/hosts");
    for my $hostResponse(@hostResponses)
    {
        my $host = new Host();

        $host->id($hostResponse->{id});
        $host->dead($hostResponse->{dead});
        $host->ip($hostResponse->{ip});
        $host->name($hostResponse->{name});

        push @retValue, $host;
    }

    return @retValue;
}

sub id
{
    my ($self, $id) = @_;
    $self->{_id} = $id if defined $id;
    return $self->{_id};
}

sub dead
{
    my ($self, $dead) = @_;
    $self->{_dead} = $dead if defined $dead;
    return $self->{_dead};
}

sub ip
{
    my ($self, $ip) = @_;
    $self->{_ip} = $ip if defined $ip;
```

```
return $self->{_ip};
}
sub name
{
my ($self, $name) = @_;
$self->{_name} = $name if defined $name;
return $self->{_name};
}
sub dump()
{
my ($self) = @_;

if(defined $self->id()) { print("id: " . $self->id() . "\n");}
if(defined $self->dead()) { print("dead: " . $self->dead() . "\n");}
if(defined $self->ip()) { print("ip: " . $self->ip() . "\n");}
if(defined $self->name()) { print("name: " . $self->name() . "\n");}
}
1;
```


Path Package

```
package Path;
use warnings;
use strict;
sub new
{
my ( $class,
    $id,
        $arrayId,
        $hostId,
        $volumeId )= @_;

    my $self = {
        _id => $id,
        _arrayId => $arrayId,
        _hostId => $hostId,
        _volumeId => $volumeId};

    bless $self, ref($class) || $class;
    return $self;
}
# This method is intended to be called statically to retrieve the
sub parseResult
{
my ($self, $somResponse) = @_;
my @retValue = ();

if($somResponse->valueof("//result/size") eq "0")
{
return @retValue;
}
```

```
my @pathResponses = $somResponse->valueof("//result/paths");
    for my $pathResponse(@pathResponses)
    {
        my $path = new Path();

        $path->id($pathResponse->{id});
        $path->arrayId($pathResponse->{arrayId});
        $path->hostId($pathResponse->{hostId});
        $path->volumeId($pathResponse->{volumeId});

        push @retValue, $path;
    }

    return @retValue;
}

sub id
{
    my ($self, $id) = @_;
    $self->{_id} = $id if defined $id;
    return $self->{_id};
}

sub arrayId
{
    my ($self, $arrayId) = @_;
    $self->{_arrayId} = $arrayId if defined $arrayId;
    return $self->{_arrayId};
}

sub hostId
{
    my ($self, $hostId) = @_;
    $self->{_hostId} = $hostId if defined $hostId;
```

```
return $self->{_hostId};
}
sub volumeId
{
my ($self, $volumeId) = @_;
$self->{_volumeId} = $volumeId if defined $volumeId;
return $self->{_volumeId};
}
sub dump()
{
my ($self) = @_;

if(defined $self->id()) { print("id: " . $self->id() . "\n");}
if(defined $self->arrayId()) { print("arrayId: " . $self->arrayId() .
"\n");}
if(defined $self->hostId()) { print("hostId: " . $self->hostId() .
"\n");}
if(defined $self->volumeId()) { print("volumeId: " . $self-
>volumeId() . "\n");}
}
1;
```

Perl Examples Requirements

Requirements

In order for the examples to work, you'll need to download the following libraries (and all their dependencies) from cpan:

- ◆ Ssleay - SSL library for windows
- ◆ SOAP::LITE - Basic implementation of SOAP

Examples

- ◆ getStorage.pl - Queries localhost for storage devices without using iterations.
- ◆ getPaths.pl - Queries localhost for all paths using iterations.
- ◆ computeTotalStoragePerArray.pl - Iterates through all the volumes of all the storage-devices and sums up their size to figure out the capacity of each storage.
- ◆ findDisconnectedHosts.pl - Compares the list of paths and the list of hosts, printing out any hosts that are not in a path.

Client side API

- ◆ APISessionContext.pm
- ◆ APIVersion.pm
- ◆ Host.pm
- ◆ Iterator.pm
- ◆ Path.pm
- ◆ SOAPParam.pm
- ◆ StorageArray.pm
- ◆ Timeout.pm
- ◆ Volume.pm

SOAPParam Package

```
package SOAPParam;

use warnings;
use strict;

sub new {
    my ( $class,
        $name,
        $type,
        $value ) = @_;

    my $self = {
        _name => $name,
        _type => $type,
        _value => $value
    };

    bless $self, ref($class) || $class;
    return $self;
}

sub name {
    my ( $self, $name ) = @_;
    $self->{_name} = $name if defined($name);
    return $self->{_name};
}

sub type {
    my ( $self, $type ) = @_;
    $self->{_type} = $type if defined($type);
    return $self->{_type};
}

sub value {
    my ( $self, $value ) = @_;
```

```
$self->{_value} = $value if defined($value);  
return $self->{_value};  
}  
  
sub getSOAP  
{  
  my ( $self ) = @_;  
  return SOAP::Data->new(  
    name => $self->{_name},  
    type => $self->{_type},  
    value => $self->{_value});  
}  
1;
```

Timeout Package

```
package Timeout;

use warnings;
use strict;
use SOAPParam;

our @ISA = qw(SOAPParam);

sub new
{
    my ( $class, $timeout ) = @_;

    my $self = $class->SUPER::new(
        "int_2",
        "int",
        $timeout);

    $self->{_timeout} = $timeout;

    bless $self, ref($class) || $class;
    return $self;
}

sub timeout {
    my ( $self, $timeout ) = @_;
    if(defined $timeout)
    {
        $self->{_timeout} = $timeout;
        $self->value($timeout);
    }

    return $self->{_timeout};
}

1;
```

StorageArray Package

```
package StorageArray;

use warnings;
use strict;

sub new
{
    my ( $class,
        $id,
        $ip,
        $name,
        $capacityGB,
        $firmware,
        $model,
        $rawCapacityGB,
        $serialNumber,
        $vendor )= @_;

    my $self = {
        _id => $id,
        _ip => $ip,
        _name => $name,
        _capacityGB => $capacityGB,
        _firmware => $firmware,
        _model => $model,
        _rawCapacityGB => $rawCapacityGB,
        _serialNumber => $serialNumber,
        _vendor => $vendor};

    bless $self, ref($class) || $class;
    return $self;
}
```



```
# This method is intended to be called statically to retrieve the
sub parseResult
{
my ($self, $somResponse) = @_;
my @retValue = ();

if($somResponse->valueof("//result/size") eq "0")
{
return @retValue;
}

my @storageArraysResponse = $somResponse->valueof("//result/arrays");
for my $storageArrayResponse(@storageArraysResponse)
{
my $storageArray = new StorageArray();

$storageArray->id($storageArrayResponse->{id});
$storageArray->ip($storageArrayResponse->{ip});
$storageArray->name($storageArrayResponse->{name});
$storageArray->capacityGB($storageArrayResponse->{capacityGB});
$storageArray->firmware($storageArrayResponse->{firmware});
$storageArray->model($storageArrayResponse->{model});
$storageArray->rawCapacityGB($storageArrayResponse->{rawCapacityGB});
$storageArray->serialNumber($storageArrayResponse->{serialNumber});
$storageArray->vendor($storageArrayResponse->{vendor});

push @retValue, $storageArray;
}

return @retValue;
}

sub id
```

```
{
my ($self, $id) = @_;
$self->{_id} = $id if defined $id;
return $self->{_id};
}

sub ip
{
my ($self, $ip) = @_;
$self->{_ip} = $ip if defined $ip;
return $self->{_ip};
}

sub name()
{
my ($self, $name) = @_;
$self->{_name} = $name if defined $name;
return $self->{_name};
}

sub capacityGB()
{
my ($self, $capacityGB) = @_;
$self->{_capacityGB} = $capacityGB if defined $capacityGB;
return $self->{_capacityGB};
}

sub firmware()
{
my ($self, $firmware) = @_;
$self->{_firmware} = $firmware if defined $firmware;
return $self->{_firmware};
}

sub model()
{
my ($self, $model) = @_;
```

```
$self->{_model} = $model if defined $model;
return $self->{_model};
}

sub rawCapacityGB()
{
my ($self, $rawCapacityGB) = @_;
$self->{_rawCapacityGB} = $rawCapacityGB if defined $rawCapacityGB;
return $self->{_rawCapacityGB};
}

sub serialNumber()
{
my ($self, $serialNumber) = @_;
$self->{_serialNumber} = $serialNumber if defined $serialNumber;
return $self->{_serialNumber};
}

sub vendor()
{
my ($self, $vendor) = @_;
$self->{_vendor} = $vendor if defined $vendor;
return $self->{_vendor};
}

sub dump()
{
my ($self) = @_;

if(defined $self->id()) { print("id: " . $self->id() . "\n");}
if(defined $self->ip()) { print("ip: " . $self->ip() . "\n");}
if(defined $self->name()) { print("name: " . $self->name() . "\n");}
if(defined $self->capacityGB()) { print("capacityGB: " . $self->capacityGB() . "\n");}
if(defined $self->firmware()) { print("firmware: " . $self->firmware() . "\n");}
```

```
if(defined $self->model()) { print("model: " . $self->model() .
"\n");}

if(defined $self->rawCapacityGB()) { print("rawCapacityGB: " . $self-
>rawCapacityGB() . "\n");}

if(defined $self->serialNumber()) { print("serialNumber: " . $self-
>serialNumber() . "\n");}

if(defined $self->vendor()) { print("vendor: " . $self->vendor() .
"\n");}

}

1;
```

Volume Package

```
package Volume;

use warnings;
use strict;

sub new
{
    my ( $class,
        $id,
        $capacityGB,
        $name,
        $rawCapacityGB,
        $redundancy,
        $storageID,
        $type ) = @_;

    my $self = {
        _id => $id,
        _capacityGB => $capacityGB,
        _name => $name,
        _rawCapacityGB => $rawCapacityGB,
        _redundancy => $redundancy,
        _storageID => $storageID,
        _type => $type };

    bless $self, ref($class) || $class;
    return $self;
}

# This method is intended to be called statically to retrieve the
sub parseResult
{
    my ($self, $somResponse) = @_;
```

```
my @retValue = ();

if($somResponse->valueof("//result/size") eq "0")
{
return @retValue;
}

my @volumeResponses = $somResponse->valueof("//result/volumes");
for my $volumeResponse(@volumeResponses)
{
my $volume = new Volume();

$volume->class($volumeResponse->{class});
$volume->id($volumeResponse->{id});
$volume->capacityGB($volumeResponse->{capacityGB});
$volume->name($volumeResponse->{name});
$volume->rawCapacityGB($volumeResponse->{rawCapacityGB});
$volume->redundancy($volumeResponse->{redundancy});
$volume->storageID($volumeResponse->{storageID});
$volume->type ($volumeResponse->{type});

push @retValue, $volume;
}

return @retValue;
}

sub class()
{
my ($self, $class ) = @_;
$self->{_class} = $class if defined $class;
return $self->{_class};
}
```

```
sub id()
{
my ($self, $id ) = @_;
$self->{_id} = $id if defined $id;
return $self->{_id};
}

sub capacityGB()
{
my ($self, $capacityGB ) = @_;
$self->{_capacityGB} = $capacityGB if defined $capacityGB;
return $self->{_capacityGB};
}

sub name()
{
my ($self, $name ) = @_;
$self->{_name} = $name if defined $name;
return $self->{_name};
}

sub rawCapacityGB()
{
my ($self, $rawCapacityGB ) = @_;
$self->{_rawCapacityGB} = $rawCapacityGB if defined $rawCapacityGB;
return $self->{_rawCapacityGB};
}

sub redundancy()
{
my ($self, $redundancy ) = @_;
$self->{_redundancy} = $redundancy if defined $redundancy;
return $self->{_redundancy};
}
```

```
sub storageID()
{
my ($self, $storageID ) = @_;
$self->{_storageID} = $storageID if defined $storageID;
return $self->{_storageID};
}

sub type()
{
my ($self, $type ) = @_;
$self->{_type} = $type if defined $type;
return $self->{_type};
}

sub dump()
{
my ($self) = @_;

if(defined $self->class()) { print("class: " . $self->class() .
"\n"); }

if(defined $self->id()) { print("id: " . $self->id() . "\n"); }

if(defined $self->capacityGB()) { print("capacityGB: " . $self->
capacityGB() . "\n"); }

if(defined $self->name()) { print("name: " . $self->name() . "\n"); }

if(defined $self->rawCapacityGB()) { print("rawCapacityGB: " . $self->
rawCapacityGB() . "\n"); }

if(defined $self->redundancy()) { print("redundancy: " . $self->
redundancy() . "\n"); }

if(defined $self->storageID()) { print("storageID: " . $self->
storageID() . "\n"); }

if(defined $self->type()) { print("type: " . $self->type() . "\n"); }
}

1;
```


Appendix A: WSDL File

WSDL is an XML-based service description on how to communicate using web services. WSDL defines services as collections of network endpoints, or ports. WSDL specification provides an XML format for documents for this purpose.

The abstract definition of ports and messages is separated from their concrete use or instance, allowing the reuse of these definitions. A port is defined by associating a network address with a reusable binding, and a collection of ports define a service.

Messages are abstract descriptions of the data being exchanged, and port types are abstract collections of supported operations. The concrete protocol and data format specifications for a particular port type constitutes a reusable binding, where the messages and operations are then bound to a concrete network protocol and message format. In this way, WSDL describes the public interface to the web service.

The following data is the WSDL specification in XML format:

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="SANSscreenAPIService" targetNamespace="http://
wsapi.sansscreen.NetApp.com/SANSscreenAPI" xmlns:tns="http://
wsapi.sansscreen.NetApp.com/SANSscreenAPI" xmlns:xsd="http://
www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xml-
soap.org/wsdl/soap/" xmlns="http://schemas.xmlsoap.org/wsdl/">
  <types>
    <schema targetNamespace="http://wsapi.sansscreen.NetApp.com/
SANSscreenAPI" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:soap11-enc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:tns="http://wsapi.sansscreen.NetApp.com/SANSscreenAPI"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance">
      <complexType name="addHostToApplication">
        <sequence>
          <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
          <element name="String_2" nillable="true" type="string"/>
          <element name="String_3" nillable="true" type="string"/>
        </sequence>
      </complexType>
    </schema>
  </types>
</definitions>
```

```

        <element name="String_4" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="APISessionContext">
        <sequence>
            <element name="clientVersion" nillable="true"
type="string"/>
            <element name="id" nillable="true" type="string"/>
            <element name="limit" type="int"/>
            <element name="time" type="long"/></sequence></complex-
Type>
    <complexType name="addHostToApplicationResponse">
        <sequence>
            <element name="result" type="boolean"/></sequence></com-
plexType>
    <complexType name="APIBadRequestException">
        <complexContent>
            <extension base="tns:APIException">
                <sequence/></extension></complexContent></complexType>
    <complexType name="APIException">
        <sequence>
            <element name="message" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="closeSession">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/></sequence></complexType>
    <complexType name="closeSessionResponse">
        <sequence>
            <element name="result" type="boolean"/></sequence></com-
plexType>
    <complexType name="findObjectByWwn">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="RequestIterator">
        <sequence>
            <element name="hasMore" type="boolean"/>

```

```

        <element name="methodName" nillable="true"
type="string"/>
        <element name="nextKey" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="findObjectByWwnResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Device-
Base"/></sequence></complexType>
    <complexType name="DeviceBase">
        <sequence>
            <element name="id" nillable="true" type="string"/>
            <element name="endTime" type="long"/>
            <element name="startTime" type="long"/>
            <element name="dead" type="boolean"/>
            <element name="ip" nillable="true" type="string"/>
            <element name="name" nillable="true" type="string"/>
            <element name="objectType" nillable="true"
type="string"/></sequence></complexType>
    <complexType name="Switch">
        <complexContent>
            <extension base="tns:DeviceBase">
                <sequence>
                    <element name="model" nillable="true" type="string"/
>
                    <element name="status" nillable="true"
type="string"/>
                    <element name="vendor" nillable="true"
type="string"/>
                    <element name="wwn" nillable="true" type="string"/
></sequence></extension></complexContent></complexType>
    <complexType name="StorageArray">
        <complexContent>
            <extension base="tns:DeviceBase">
                <sequence>
                    <element name="capacityGB" type="double"/>
                    <element name="microcodeVersion" nillable="true"
type="string"/>
                    <element name="model" nillable="true" type="string"/
>
                    <element name="rawCapacityGB" type="double"/>
                    <element name="serialNumber" nillable="true"

```

```

type="string"/>
    <element name="vendor" nillable="true"
type="string"/></sequence></extension></complexContent></complex-
Type>
    <complexType name="Host">
        <complexContent>
            <extension base="tns:DeviceBase">
                <sequence/></extension></complexContent></complexType>
    <complexType name="Tape">
        <complexContent>
            <extension base="tns:DeviceBase">
                <sequence>
                    <element name="model" nillable="true" type="string"/
>
                    <element name="serialNumber" nillable="true"
type="string"/>
                    <element name="vendor" nillable="true"
type="string"/></sequence></extension></complexContent></complex-
Type>
    <complexType name="findObjectsByIp">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="findObjectsByIpResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Device-
Response"/></sequence></complexType>
    <complexType name="DeviceResponse">
        <sequence>
            <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
            <element name="size" type="int"/>
            <element maxOccurs="unbounded" minOccurs="0"
name="devices" nillable="true" type="tns:DeviceBase"/></
sequence></complexType>
    <complexType name="findObjectsByName">
        <sequence>

```

```

        <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
        <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
        <element name="String_3" nillable="true" type="string"/
></sequence></complexType>
        <complexType name="findObjectsByNameResponse">
        <sequence>
        <element name="result" nillable="true" type="tns:Device-
Response"/></sequence></complexType>
        <complexType name="getAPIVersion">
        <sequence/></complexType>
        <complexType name="getAPIVersionResponse">
        <sequence>
        <element name="result" nillable="true" type="string"/></
sequence></complexType>
        <complexType name="getAllApplications">
        <sequence>
        <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
        <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/></sequence></complexType>
        <complexType name="getAllApplicationsResponse">
        <sequence>
        <element name="result" nillable="true" type="tns:Appli-
cationResponse"/></sequence></complexType>
        <complexType name="ApplicationResponse">
        <sequence>
        <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
        <element name="size" type="int"/>
        <element maxOccurs="unbounded" minOccurs="0"
name="applications" nillable="true" type="tns:Application"/></
sequence></complexType>
        <complexType name="Application">
        <sequence>
        <element name="id" nillable="true" type="string"/>
        <element name="businessUnit" nillable="true"
type="string"/>
        <element maxOccurs="unbounded" minOccurs="0" name="hos-
tIds" nillable="true" type="string"/>

```

```

        <element name="ignoreSharing" type="boolean"/>
        <element name="name" nillable="true" type="string"/>
        <element name="priority" nillable="true" type="string"/>
    ></sequence></complexType>
    <complexType name="getApplication">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/>
        ></sequence></complexType>
    <complexType name="getApplicationResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Appli-
cationResponse"/></sequence></complexType>
    <complexType name="getApplicationsOfHost">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/>
        ></sequence></complexType>
    <complexType name="getApplicationsOfHostResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Appli-
cationResponse"/></sequence></complexType>
    <complexType name="getConnectedPorts">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/></sequence></complexType>
    <complexType name="getConnectedPortsResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Logi-
calPortResponse"/></sequence></complexType>
    <complexType name="LogicalPortResponse">
        <sequence>
            <element name="requestIterator" nillable="true"

```

```

type="tns:RequestIterator"/>
    <element name="size" type="int"/>
    <element maxOccurs="unbounded" minOccurs="0" name="logicalPorts" nillable="true" type="tns:LogicalPort"/></sequence></complexType>
    <complexType name="LogicalPort">
        <complexContent>
            <extension base="tns:Port">
                <sequence>
                    <element name="physicalPortId" nillable="true"
type="string"/></sequence></extension></complexContent></complexType>
    <complexType name="Port">
        <sequence>
            <element name="id" nillable="true" type="string"/>
            <element name="connectedPortId" nillable="true"
type="string"/>
            <element name="dead" type="boolean"/>
            <element name="deviceId" nillable="true" type="string"/>
            <element name="name" nillable="true" type="string"/>
            <element name="nodeId" nillable="true" type="string"/>
            <element name="speed" type="int"/>
            <element name="state" nillable="true" type="string"/>
            <element name="status" nillable="true" type="string"/>
            <element name="wwn" nillable="true" type="string"/></sequence></complexType>
    <complexType name="getDRPathsByVolume">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/>
        </sequence></complexType>
    <complexType name="getDRPathsByVolumeResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:DRPath-Response"/></sequence></complexType>
    <complexType name="DRPathResponse">
        <sequence>
            <element name="requestIterator" nillable="true"

```

```

type="tns:RequestIterator"/>
    <element name="size" type="int"/>
    <element maxOccurs="unbounded" minOccurs="0"
name="drPaths" nillable="true" type="tns:DRPath"/></sequence></
complexType>
    <complexType name="DRPath">
        <sequence>
            <element name="id" nillable="true" type="string"/>
            <element name="hostId" nillable="true" type="string"/>
            <element name="remoteHostId" nillable="true"
type="string"/>
            <element name="remoteReplicaVolumeId" nillable="true"
type="string"/>
            <element name="remoteStorageId" nillable="true"
type="string"/>
            <element name="replicaVolumeId" nillable="true"
type="string"/>
            <element name="secondaryReplicaVolumeId" nillable="true"
type="string"/>
            <element name="storageId" nillable="true" type="string"/
>
            <element name="volumeId" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getDevice">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="String_2" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getDeviceResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Device-
Base"/></sequence></complexType>
    <complexType name="getDeviceGroup">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="String_2" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getDeviceGroupResponse">
        <sequence>

```

```

        <element name="result" nillable="true" type="tns:Device-
Group"/></sequence></complexType>
    <complexType name="DeviceGroup">
        <sequence>
            <element name="id" nillable="true" type="string"/>
            <element name="endTime" type="long"/>
            <element name="startTime" type="long"/>
            <element name="name" nillable="true" type="string"/>
            <element name="storageManagementId" nillable="true"
type="string"/>
            <element name="storageManagementName" nillable="true"
type="string"/>
            <element name="type" nillable="true" type="string"/></
sequence></complexType>
    <complexType name="getDeviceGroupVolumes">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getDeviceGroupVolumesResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Volume-
Response"/></sequence></complexType>
    <complexType name="VolumeResponse">
        <sequence>
            <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
            <element name="size" type="int"/>
            <element maxOccurs="unbounded" minOccurs="0" name="vol-
umes" nillable="true" type="tns:Volume"/></sequence></complex-
Type>
    <complexType name="Volume">
        <sequence>
            <element name="id" nillable="true" type="string"/>
            <element name="endTime" type="long"/>
            <element name="startTime" type="long"/>
            <element name="capacityGB" type="double"/>
            <element name="diskSpeed" nillable="true" type="string"/

```

```

>
    <element name="diskType" nillable="true" type="string"/>
    <element name="name" nillable="true" type="string"/>
    <element name="rawCapacityGB" type="double"/>
    <element name="redundancy" nillable="true"
type="string"/>
    <element name="storageID" nillable="true" type="string"/>
>
    <element name="type" nillable="true" type="string"/>
    <element name="virtual" type="boolean"/></sequence></
complexType>
    <complexType name="getDeviceGroups">
    <sequence>
    <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
    <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/></sequence></complexType>
    <complexType name="getDeviceGroupsResponse">
    <sequence>
    <element name="result" nillable="true" type="tns:Device-
GroupResponse"/></sequence></complexType>
    <complexType name="DeviceGroupResponse">
    <sequence>
    <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
    <element name="size" type="int"/>
    <element maxOccurs="unbounded" minOccurs="0"
name="deviceGroups" nillable="true" type="tns:DeviceGroup"/></
sequence></complexType>
    <complexType name="getDeviceGroupsByStorage">
    <sequence>
    <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
    <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
    <element name="String_3" nillable="true" type="string"/>
></sequence></complexType>
    <complexType name="getDeviceGroupsByStorageResponse">
    <sequence>
    <element name="result" nillable="true" type="tns:Device-
GroupResponse"/></sequence></complexType>

```

```

    <complexType name="getDeviceGroupsByVolume">
      <sequence>
        <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
        <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
        <element name="String_3" nillable="true" type="string"/>
      </sequence></complexType>
    <complexType name="getDeviceGroupsByVolumeResponse">
      <sequence>
        <element name="result" nillable="true" type="tns:Device-
GroupResponse"/></sequence></complexType>
    <complexType name="getGeneric">
      <sequence>
        <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
        <element name="String_2" nillable="true" type="string"/>
      </sequence></complexType>
    <complexType name="getGenericResponse">
      <sequence>
        <element name="result" nillable="true"
type="tns:Generic"/></sequence></complexType>
    <complexType name="Generic">
      <sequence>
        <element name="id" nillable="true" type="string"/>
        <element name="dead" type="boolean"/>
        <element name="wwn" nillable="true" type="string"/></
sequence></complexType>
    <complexType name="getGenerics">
      <sequence>
        <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
        <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/></sequence></complexType>
    <complexType name="getGenericsResponse">
      <sequence>
        <element name="result" nillable="true" type="tns:Gener-
icResponse"/></sequence></complexType>
    <complexType name="GenericResponse">
      <sequence>
        <element name="requestIterator" nillable="true"

```

```

type="tns:RequestIterator"/>
    <element name="size" type="int"/>
    <element maxOccurs="unbounded" minOccurs="0"
name="generics" nillable="true" type="tns:Generic"/></sequence></
complexType>
    <complexType name="getHost">
    <sequence>
    <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
    <element name="String_2" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getHostResponse">
    <sequence>
    <element name="result" nillable="true" type="tns:Host"/
></sequence></complexType>
    <complexType name="getHostPolicy">
    <sequence>
    <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
    <element name="String_2" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getHostPolicyResponse">
    <sequence>
    <element name="result" nillable="true" type="tns:Pol-
icy"/></sequence></complexType>
    <complexType name="Policy">
    <sequence>
    <element name="id" nillable="true" type="string"/>
    <element name="arrayId" nillable="true" type="string"/>
    <element name="hopsNumber" type="int"/>
    <element name="hostId" nillable="true" type="string"/>
    <element name="hostPortsRedundancy" type="int"/>
    <element name="owner" nillable="true" type="string"/>
    <element name="redundancy" nillable="true"
type="string"/>
    <element name="volumeId" nillable="true" type="string"/>
    <element name="volumeSharingScope" nillable="true"
type="string"/></sequence></complexType>
    <complexType name="getHosts">
    <sequence>
    <element name="APISessionContext_1" nillable="true"

```

```

type="tns:APISessionContext"/>
    <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/></sequence></complexType>
    <complexType name="getHostsResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:HostRe-
sponse"/></sequence></complexType>
    <complexType name="HostResponse">
        <sequence>
            <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
            <element name="size" type="int"/>
            <element maxOccurs="unbounded" minOccurs="0"
name="hosts" nillable="true" type="tns:Host"/></sequence></com-
plexType>
    <complexType name="getHostsOfApplication">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getHostsOfApplicationResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:HostRe-
sponse"/></sequence></complexType>
    <complexType name="getLogicalPortsByLogicalSwitch">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getLogicalPortsByLogicalSwitchResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Logi-
calPortResponse"/></sequence></complexType>
    <complexType name="getLogicalPortsByPhysicalPort">
        <sequence>

```

```

        <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
        <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
        <element name="String_3" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getLogicalPortsByPhysicalPortResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Logi-
calPortResponse"/></sequence></complexType>
    <complexType name="getLogicalPortsByPhysicalSwitch">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getLogicalPortsByPhysicalSwitch-
Response">
        <sequence>
            <element name="result" nillable="true" type="tns:Logi-
calPortResponse"/></sequence></complexType>
    <complexType name="getLogicalSwitchesByPhysicalSwitch">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getLogicalSwitchesByPhysicalSwitch-
Response">
        <sequence>
            <element name="result" nillable="true" type="tns:Switch-
Response"/></sequence></complexType>
    <complexType name="SwitchResponse">
        <sequence>
            <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
            <element name="size" type="int"/>

```

```

        <element maxOccurs="unbounded" minOccurs="0"
name="switches" nillable="true" type="tns:Switch"/></sequence></
complexType>
    <complexType name="getNodesOfDevice">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/>
        </sequence></complexType>
    <complexType name="getNodesOfDeviceResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:NodeRe-
sponse"/></sequence></complexType>
    <complexType name="NodeResponse">
        <sequence>
            <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
            <element name="size" type="int"/>
            <element maxOccurs="unbounded" minOccurs="0"
name="nodes" nillable="true" type="tns:Node"/></sequence></com-
plexType>
    <complexType name="Node">
        <sequence>
            <element name="id" nillable="true" type="string"/>
            <element name="dead" type="boolean"/>
            <element name="deviceId" nillable="true" type="string"/>
            <element name="wwn" nillable="true" type="string"/></
sequence></complexType>
    <complexType name="getPathPolicy">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="Path_2" nillable="true" type="tns:Path"/>
        </sequence></complexType>
    <complexType name="Path">
        <sequence>
            <element name="id" nillable="true" type="string"/>
            <element name="endTime" type="long"/>
            <element name="startTime" type="long"/>

```

```

        <element name="arrayId" nillable="true" type="string"/>
        <element name="hostId" nillable="true" type="string"/>
        <element name="minimumNumberOfHops" type="int"/>
        <element name="numberOfFabrics" type="int"/>
        <element name="numberOfHostPorts" type="int"/>
        <element name="numberOfStoragePorts" type="int"/>
        <element name="spf" type="boolean"/>
        <element name="volumeId" nillable="true" type="string"/>
    ></sequence></complexType>
    <complexType name="getPathPolicyResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Pol-
icity"/></sequence></complexType>
    <complexType name="getPaths">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/></sequence></complexType>
    <complexType name="getPathsResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Path-
Response"/></sequence></complexType>
    <complexType name="PathResponse">
        <sequence>
            <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
            <element name="size" type="int"/>
            <element maxOccurs="unbounded" minOccurs="0"
name="paths" nillable="true" type="tns:Path"/></sequence></com-
plexType>
    <complexType name="getPathsByHost">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/>
        ></sequence></complexType>
    <complexType name="getPathsByHostResponse">
        <sequence>

```

```

        <element name="result" nillable="true" type="tns:Path-
Response"/></sequence></complexType>
    <complexType name="getPhysicalPortByLogicalPort">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/>
        </sequence></complexType>
    <complexType name="getPhysicalPortByLogicalPortResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:PortRe-
sponse"/></sequence></complexType>
    <complexType name="PortResponse">
        <sequence>
            <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
            <element name="size" type="int"/>
            <element maxOccurs="unbounded" minOccurs="0"
name="ports" nillable="true" type="tns:Port"/></sequence></com-
plexType>
    <complexType name="getPolicies">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/></sequence></complexType>
    <complexType name="getPoliciesResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Policy-
Response"/></sequence></complexType>
    <complexType name="PolicyResponse">
        <sequence>
            <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
            <element name="size" type="int"/>
            <element maxOccurs="unbounded" minOccurs="0" name="pol-
icies" nillable="true" type="tns:Policy"/></sequence></complex-
Type>
    <complexType name="getPort">

```

```

        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="String_2" nillable="true" type="string"/
></sequence></complexType>
        <complexType name="getPortResponse">
            <sequence>
                <element name="result" nillable="true" type="tns:Port"/
></sequence></complexType>
        <complexType name="getPorts">
            <sequence>
                <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
                <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/></sequence></complexType>
        <complexType name="getPortsResponse">
            <sequence>
                <element name="result" nillable="true" type="tns:PortRe-
sponse"/></sequence></complexType>
        <complexType name="getPortsOfDevice">
            <sequence>
                <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
                <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
                <element name="String_3" nillable="true" type="string"/
></sequence></complexType>
        <complexType name="getPortsOfDeviceResponse">
            <sequence>
                <element name="result" nillable="true" type="tns:PortRe-
sponse"/></sequence></complexType>
        <complexType name="getPortsOfDeviceType">
            <sequence>
                <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
                <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
                <element name="String_3" nillable="true" type="string"/
></sequence></complexType>
        <complexType name="getPortsOfDeviceTypeResponse">
            <sequence>

```

```

        <element name="result" nillable="true" type="tns:PortRe-
sponse"/></sequence></complexType>
    <complexType name="getPortsOfNode">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getPortsOfNodeResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:PortRe-
sponse"/></sequence></complexType>
    <complexType name="getRecentDevice">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="String_2" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getRecentDeviceResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Device-
Base"/></sequence></complexType>
    <complexType name="getRecentHost">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="String_2" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getRecentHostResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Host"/
></sequence></complexType>
    <complexType name="getRecentStorageArray">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="String_2" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getRecentStorageArrayResponse">

```

```

        <sequence>
            <element name="result" nillable="true" type="tns:StorageArray"/></sequence></complexType>
        <complexType name="getRecentSwitch">
            <sequence>
                <element name="APISessionContext_1" nillable="true" type="tns:APISessionContext"/>
                <element name="String_2" nillable="true" type="string"/>
            </sequence></complexType>
        <complexType name="getRecentSwitchResponse">
            <sequence>
                <element name="result" nillable="true" type="tns:Switch"/></sequence></complexType>
        <complexType name="getRecentTape">
            <sequence>
                <element name="APISessionContext_1" nillable="true" type="tns:APISessionContext"/>
                <element name="String_2" nillable="true" type="string"/>
            </sequence></complexType>
        <complexType name="getRecentTapeResponse">
            <sequence>
                <element name="result" nillable="true" type="tns:Tape"/>
            </sequence></complexType>
        <complexType name="getRecentVolume">
            <sequence>
                <element name="APISessionContext_1" nillable="true" type="tns:APISessionContext"/>
                <element name="String_2" nillable="true" type="string"/>
            </sequence></complexType>
        <complexType name="getRecentVolumeResponse">
            <sequence>
                <element name="result" nillable="true" type="tns:Volume"/></sequence></complexType>
        <complexType name="getSANscreenVersion">
            <sequence/></complexType>
        <complexType name="getSANscreenVersionResponse">
            <sequence>
                <element name="result" nillable="true" type="string"/></sequence></complexType>
        <complexType name="getStorageArray">
            <sequence>

```

```

        <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
        <element name="String_2" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getStorageArrayResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Stor-
ageArray"/></sequence></complexType>
    <complexType name="getStorageArrays">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/></sequence></complexType>
    <complexType name="getStorageArraysResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Stor-
ageArrayResponse"/></sequence></complexType>
    <complexType name="StorageArrayResponse">
        <sequence>
            <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
            <element name="size" type="int"/>
            <element maxOccurs="unbounded" minOccurs="0"
name="arrays" nillable="true" type="tns:StorageArray"/></
sequence></complexType>
    <complexType name="getSwitch">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="String_2" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getSwitchResponse">
        <sequence>
            <element name="result" nillable="true"
type="tns:Switch"/></sequence></complexType>
    <complexType name="getSwitches">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"

```

```

type="tns:RequestIterator"/></sequence></complexType>
  <complexType name="getSwitchesResponse">
    <sequence>
      <element name="result" nillable="true" type="tns:Switch-
Response"/></sequence></complexType>
  <complexType name="getSynchronizationsByStorageArray">
    <sequence>
      <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
      <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
      <element name="String_3" nillable="true" type="string"/
></sequence></complexType>
  <complexType name="getSynchronizationsByStorageArrayRe-
sponse">
    <sequence>
      <element name="result" nillable="true" type="tns:Syn-
chronizationResponse"/></sequence></complexType>
  <complexType name="SynchronizationResponse">
    <sequence>
      <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
      <element name="size" type="int"/>
      <element maxOccurs="unbounded" minOccurs="0" name="syn-
chronizations" nillable="true" type="tns:Synchronization"/></
sequence></complexType>
  <complexType name="Synchronization">
    <sequence>
      <element name="id" nillable="true" type="string"/>
      <element name="sourceStorageArrayIP" nillable="true"
type="string"/>
      <element name="sourceStorageArrayId" nillable="true"
type="string"/>
      <element name="sourceVolumeName" nillable="true"
type="string"/>
      <element name="storageId" nillable="true" type="string"/
>
      <element name="synchronizationMode" nillable="true"
type="string"/>
      <element name="targetStorageArrayIP" nillable="true"
type="string"/>

```

```

        <element name="targetStorageArrayId" nillable="true"
type="string"/>
        <element name="targetVolumeName" nillable="true"
type="string"/></sequence></complexType>
    <complexType name="getTape">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="String_2" nillable="true" type="string"/>
        </sequence></complexType>
    <complexType name="getTapeResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Tape"/>
        </sequence></complexType>
    <complexType name="getTapes">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/></sequence></complexType>
    <complexType name="getTapesResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:TapeRe-
sponse"/></sequence></complexType>
    <complexType name="TapeResponse">
        <sequence>
            <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
            <element name="size" type="int"/>
            <element maxOccurs="unbounded" minOccurs="0"
name="tapes" nillable="true" type="tns:Tape"/></sequence></com-
plexType>
    <complexType name="getViolations">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/></sequence></complexType>
    <complexType name="getViolationsResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Viola-

```

```

tionResponse"/></sequence></complexType>
    <complexType name="ViolationResponse">
        <sequence>
            <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
            <element name="size" type="int"/>
            <element maxOccurs="unbounded" minOccurs="0" name="vio-
lations" nillable="true" type="tns:Violation"/></sequence></com-
plexType>
    <complexType name="Violation">
        <sequence>
            <element name="id" nillable="true" type="string"/>
            <element name="endTime" type="long"/>
            <element name="startTime" type="long"/>
            <element name="arrayId" nillable="true" type="string"/>
            <element name="hostId" nillable="true" type="string"/>
            <element name="type" nillable="true" type="string"/>
            <element name="volumeId" nillable="true" type="string"/>
        </sequence></complexType>
    <complexType name="getViolationsByHost">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/>
        </sequence></complexType>
    <complexType name="getViolationsByHostResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Viola-
tionResponse"/></sequence></complexType>
    <complexType name="getVolume">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="String_2" nillable="true" type="string"/>
        </sequence></complexType>
    <complexType name="getVolumeResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Vol-
ume"/></sequence></complexType>

```

```

    <complexType name="getVolumeMapsByStorageArray">
      <sequence>
        <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
        <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
        <element name="String_3" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getVolumeMapsByStorageArrayResponse">
      <sequence>
        <element name="result" nillable="true" type="tns:Vol-
umeMapResponse"/></sequence></complexType>
    <complexType name="VolumeMapResponse">
      <sequence>
        <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
        <element name="size" type="int"/>
        <element maxOccurs="unbounded" minOccurs="0" name="vol-
umeMaps" nillable="true" type="tns:VolumeMap"/></sequence></com-
plexType>
    <complexType name="VolumeMap">
      <sequence>
        <element name="id" nillable="true" type="string"/>
        <element name="lun" nillable="true" type="string"/>
        <element name="protocolController" nillable="true"
type="string"/>
        <element name="storagePortId" nillable="true"
type="string"/>
        <element name="storagePortWwn" nillable="true"
type="string"/>
        <element name="volumeId" nillable="true" type="string"/>
        <element name="wwnsecurityvalidity" type="boolean"/></
sequence></complexType>
    <complexType name="getVolumeMasksByStorageArray">
      <sequence>
        <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
        <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
        <element name="String_3" nillable="true" type="string"/
></sequence></complexType>

```

```

    <complexType name="getVolumeMasksByStorageArrayResponse">
      <sequence>
        <element name="result" nillable="true" type="tns:Volume-
MaskResponse"/></sequence></complexType>
    <complexType name="VolumeMaskResponse">
      <sequence>
        <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
        <element name="size" type="int"/>
        <element maxOccurs="unbounded" minOccurs="0"
name="masks" nillable="true" type="tns:VolumeMask"/></sequence></
complexType>
    <complexType name="VolumeMask">
      <sequence>
        <element name="id" nillable="true" type="string"/>
        <element name="initiatorPortOrNodeWwn" nillable="true"
type="string"/>
        <element name="protocolController" nillable="true"
type="string"/>
        <element name="storagePortId" nillable="true"
type="string"/>
        <element name="storagePortWwn" nillable="true"
type="string"/>
        <element name="volumeId" nillable="true" type="string"/>
      </sequence></complexType>
    <complexType name="getVolumes">
      <sequence>
        <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
        <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/></sequence></complexType>
    <complexType name="getVolumesResponse">
      <sequence>
        <element name="result" nillable="true" type="tns:Volume-
Response"/></sequence></complexType>
    <complexType name="getVolumesByStorageArray">
      <sequence>
        <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
        <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>

```

```

        <element name="String_3" nillable="true" type="string"/
></sequence></complexType>
    <complexType name="getVolumesByStorageArrayResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Volume-
Response"/></sequence></complexType>
    <complexType name="getVolumesSynchStatesByDRPath">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
            <element name="String_3" nillable="true" type="string"/>
            <element name="long_4" type="long"/></sequence></com-
plexType>
    <complexType name="getVolumesSynchStatesByDRPathResponse">
        <sequence>
            <element name="result" nillable="true" type="tns:Vol-
umesSynchronizationStateResponse"/></sequence></complexType>
    <complexType name="VolumesSynchronizationStateResponse">
        <sequence>
            <element name="requestIterator" nillable="true"
type="tns:RequestIterator"/>
            <element name="size" type="int"/>
            <element maxOccurs="unbounded" minOccurs="0" name="vol-
umesSynchronizationState" nillable="true" type="tns:VolumesSyn-
chronizationState"/></sequence></complexType>
    <complexType name="VolumesSynchronizationState">
        <sequence>
            <element name="id" nillable="true" type="string"/>
            <element name="detailedSynchronizationState" type="int"/
>
            <element name="sourceStorageId" nillable="true"
type="string"/>
            <element name="sourceVolumeId" nillable="true"
type="string"/>
            <element name="targetStorageId" nillable="true"
type="string"/>
            <element name="targetVolumeId" nillable="true"
type="string"/>
            <element name="timeStamp" type="long"/>

```

```

        <element name="volumesSynchronizationState" nill-
able="true" type="string"/></sequence></complexType>
        <complexType name="getVolumesSynchStatesBySourceVolume">
            <sequence>
                <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
                <element name="RequestIterator_2" nillable="true"
type="tns:RequestIterator"/>
                <element name="String_3" nillable="true" type="string"/>
                <element name="long_4" type="long"/></sequence></com-
plexType>
        <complexType name="getVolumesSynchStatesBySourceVolumeRe-
sponse">
            <sequence>
                <element name="result" nillable="true"
type="tns:VolumesSynchronizationStateResponse"/></sequence></com-
plexType>
        <complexType name="openSession">
            <sequence>
                <element name="String_1" nillable="true" type="string"/>
                <element name="int_2" type="int"/></sequence></complex-
Type>
        <complexType name="openSessionResponse">
            <sequence>
                <element name="result" nillable="true" type="tns:APISes-
sionContext"/></sequence></complexType>
        <complexType name="removeHostFromApplication">
            <sequence>
                <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
                <element name="String_2" nillable="true" type="string"/>
                <element name="String_3" nillable="true" type="string"/>
                <element name="String_4" nillable="true" type="string"/
></sequence></complexType>
        <complexType name="removeHostFromApplicationResponse">
            <sequence>
                <element name="result" type="boolean"/></sequence></com-
plexType>
        <complexType name="removeHostPolicies">
            <sequence>
                <element name="APISessionContext_1" nillable="true"

```

```

type="tns:APISessionContext"/>
    <element name="String_2" nillable="true" type="string"/>
</sequence></complexType>
    <complexType name="removeHostPoliciesResponse">
        <sequence>
            <element name="result" type="boolean"/></sequence></com-
plexType>
    <complexType name="removePolicies">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element maxOccurs="unbounded" minOccurs="0"
name="arrayOfString_2" nillable="true" type="string"/></
sequence></complexType>
    <complexType name="removePoliciesResponse">
        <sequence>
            <element name="result" type="boolean"/></sequence></com-
plexType>
    <complexType name="setApplication">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element name="String_2" nillable="true" type="string"/>
            <element name="String_3" nillable="true" type="string"/>
            <element name="String_4" nillable="true" type="string"/>
            <element name="boolean_5" type="boolean"/></sequence></
complexType>
    <complexType name="setApplicationResponse">
        <sequence>
            <element name="result" type="boolean"/></sequence></com-
plexType>
    <complexType name="setHostsPolicy">
        <sequence>
            <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
            <element maxOccurs="unbounded" minOccurs="0"
name="arrayOfString_2" nillable="true" type="string"/>
            <element name="String_3" nillable="true" type="string"/>
            <element name="int_4" type="int"/>
            <element name="int_5" type="int"/>
            <element name="String_6" nillable="true" type="string"/>

```

```

></sequence></complexType>
  <complexType name="setHostsPolicyResponse">
    <sequence>
      <element name="result" type="boolean"/></sequence></com-
plexType>
  <complexType name="setPathsPolicy">
    <sequence>
      <element name="APISessionContext_1" nillable="true"
type="tns:APISessionContext"/>
      <element maxOccurs="unbounded" minOccurs="0"
name="arrayOfPath_2" nillable="true" type="tns:Path"/>
      <element name="String_3" nillable="true" type="string"/>
      <element name="int_4" type="int"/>
      <element name="int_5" type="int"/>
      <element name="String_6" nillable="true" type="string"/>
    </sequence></complexType>
  <complexType name="setPathsPolicyResponse">
    <sequence>
      <element name="result" type="boolean"/></sequence></com-
plexType>
    <element name="addHostToApplication" type="tns:addHostToAp-
plication"/>
    <element name="addHostToApplicationResponse"
type="tns:addHostToApplicationResponse"/>
    <element name="APIException" type="tns:APIException"/>
    <element name="APIBadRequestException" type="tns:APIBa-
dRequestException"/>
    <element name="closeSession" type="tns:closeSession"/>
    <element name="closeSessionResponse" type="tns:closeSes-
sionResponse"/>
    <element name="findObjectByWwn" type="tns:findObjectByWwn"/>
  >
    <element name="findObjectByWwnResponse" type="tns:findOb-
jectByWwnResponse"/>
    <element name="findObjectsByIp" type="tns:findObjectsByIp"/>
  >
    <element name="findObjectsByIpResponse" type="tns:findOb-
jectsByIpResponse"/>
    <element name="findObjectsByName" type="tns:findObjectsBy-
Name"/>
    <element name="findObjectsByNameResponse" type="tns:findOb-

```

```

jectsByNameResponse"/>
    <element name="getAPIVersion" type="tns:getAPIVersion"/>
    <element name="getAPIVersionResponse" type="tns:getAPIVer-
sionResponse"/>
    <element name="getAllApplications" type="tns:getAllApplica-
tions"/>
    <element name="getAllApplicationsResponse" type="tns:getAl-
lApplicationsResponse"/>
    <element name="getApplication" type="tns:getApplication"/>
    <element name="getApplicationResponse" type="tns:getAppli-
cationResponse"/>
    <element name="getApplicationsOfHost" type="tns:getApplica-
tionsOfHost"/>
    <element name="getApplicationsOfHostResponse"
type="tns:getApplicationsOfHostResponse"/>
    <element name="getConnectedPorts" type="tns:getConnected-
Ports"/>
    <element name="getConnectedPortsResponse" type="tns:getCon-
nectedPortsResponse"/>
    <element name="getDRPathsByVolume" type="tns:getDRPathsBy-
Volume"/>
    <element name="getDRPathsByVolumeResponse" type="tns:getDR-
PathsByVolumeResponse"/>
    <element name="getDevice" type="tns:getDevice"/>
    <element name="getDeviceResponse" type="tns:getDeviceRe-
sponse"/>
    <element name="getDeviceGroup" type="tns:getDeviceGroup"/>
    <element name="getDeviceGroupResponse" type="tns:getDevice-
GroupResponse"/>
    <element name="getDeviceGroupVolumes" type="tns:getDevice-
GroupVolumes"/>
    <element name="getDeviceGroupVolumesResponse"
type="tns:getDeviceGroupVolumesResponse"/>
    <element name="getDeviceGroups" type="tns:getDeviceGroups"/
>
    <element name="getDeviceGroupsResponse" type="tns:getDe-
viceGroupsResponse"/>
    <element name="getDeviceGroupsByStorage" type="tns:getDe-
viceGroupsByStorage"/>
    <element name="getDeviceGroupsByStorageResponse"
type="tns:getDeviceGroupsByStorageResponse"/>

```

```

        <element name="getDeviceGroupsByVolume" type="tns:getDe-
viceGroupsByVolume"/>
        <element name="getDeviceGroupsByVolumeResponse"
type="tns:getDeviceGroupsByVolumeResponse"/>
        <element name="getGeneric" type="tns:getGeneric"/>
        <element name="getGenericResponse" type="tns:getGenericRe-
sponse"/>
        <element name="getGenerics" type="tns:getGenerics"/>
        <element name="getGenericsResponse" type="tns:getGenerics-
Response"/>
        <element name="getHost" type="tns:getHost"/>
        <element name="getHostResponse" type="tns:getHostResponse"/
>
        <element name="getHostPolicy" type="tns:getHostPolicy"/>
        <element name="getHostPolicyResponse" type="tns:getHostPol-
icyResponse"/>
        <element name="getHosts" type="tns:getHosts"/>
        <element name="getHostsResponse" type="tns:getHostsRe-
sponse"/>
        <element name="getHostsOfApplication" type="tns:getHosts0-
fApplication"/>
        <element name="getHostsOfApplicationResponse"
type="tns:getHostsOfApplicationResponse"/>
        <element name="getLogicalPortsByLogicalSwitch"
type="tns:getLogicalPortsByLogicalSwitch"/>
        <element name="getLogicalPortsByLogicalSwitchResponse"
type="tns:getLogicalPortsByLogicalSwitchResponse"/>
        <element name="getLogicalPortsByPhysicalPort"
type="tns:getLogicalPortsByPhysicalPort"/>
        <element name="getLogicalPortsByPhysicalPortResponse"
type="tns:getLogicalPortsByPhysicalPortResponse"/>
        <element name="getLogicalPortsByPhysicalSwitch"
type="tns:getLogicalPortsByPhysicalSwitch"/>
        <element name="getLogicalPortsByPhysicalSwitchResponse"
type="tns:getLogicalPortsByPhysicalSwitchResponse"/>
        <element name="getLogicalSwitchesByPhysicalSwitch"
type="tns:getLogicalSwitchesByPhysicalSwitch"/>
        <element name="getLogicalSwitchesByPhysicalSwitchResponse"
type="tns:getLogicalSwitchesByPhysicalSwitchResponse"/>
        <element name="getNodesOfDevice" type="tns:getNodesOfDe-
vice"/>

```

```

        <element name="getNodesOfDeviceResponse" type="tns:get-
NodesOfDeviceResponse"/>
        <element name="getPathPolicy" type="tns:getPathPolicy"/>
        <element name="getPathPolicyResponse" type="tns:getPathPol-
icyResponse"/>
        <element name="getPaths" type="tns:getPaths"/>
        <element name="getPathsResponse" type="tns:getPathsRe-
sponse"/>
        <element name="getPathsByHost" type="tns:getPathsByHost"/>
        <element name="getPathsByHostResponse" type="tns:getPaths-
ByHostResponse"/>
        <element name="getPhysicalPortByLogicalPort" type="tns:get-
PhysicalPortByLogicalPort"/>
        <element name="getPhysicalPortByLogicalPortResponse"
type="tns:getPhysicalPortByLogicalPortResponse"/>
        <element name="getPolicies" type="tns:getPolicies"/>
        <element name="getPoliciesResponse" type="tns:getPolicies-
Response"/>
        <element name="getPort" type="tns:getPort"/>
        <element name="getPortResponse" type="tns:getPortResponse"/>
    >
        <element name="getPorts" type="tns:getPorts"/>
        <element name="getPortsResponse" type="tns:getPortsRe-
sponse"/>
        <element name="getPortsOfDevice" type="tns:getPortsOfDe-
vice"/>
        <element name="getPortsOfDeviceResponse" type="tns:getPort-
sOfDeviceResponse"/>
        <element name="getPortsOfDeviceType" type="tns:getPorts-
OfDeviceType"/>
        <element name="getPortsOfDeviceTypeResponse" type="tns:get-
PortsOfDeviceTypeResponse"/>
        <element name="getPortsOfNode" type="tns:getPortsOfNode"/>
        <element name="getPortsOfNodeResponse" type="tns:getPorts-
OfNodeResponse"/>
        <element name="getRecentDevice" type="tns:getRecentDevice"/>
    >
        <element name="getRecentDeviceResponse" type="tns:getRe-
centDeviceResponse"/>
        <element name="getRecentHost" type="tns:getRecentHost"/>
        <element name="getRecentHostResponse" type="tns:getRec-

```

```

entHostResponse"/>
    <element name="getRecentStorageArray" type="tns:getRecentStorageArray"/>
    <element name="getRecentStorageArrayResponse" type="tns:getRecentStorageArrayResponse"/>
    <element name="getRecentSwitch" type="tns:getRecentSwitch"/>
>
    <element name="getRecentSwitchResponse" type="tns:getRecentSwitchResponse"/>
    <element name="getRecentTape" type="tns:getRecentTape"/>
    <element name="getRecentTapeResponse" type="tns:getRecentTapeResponse"/>
    <element name="getRecentVolume" type="tns:getRecentVolume"/>
>
    <element name="getRecentVolumeResponse" type="tns:getRecentVolumeResponse"/>
    <element name="getSANSscreenVersion" type="tns:getSANSscreenVersion"/>
    <element name="getSANSscreenVersionResponse" type="tns:getSANSscreenVersionResponse"/>
    <element name="getStorageArray" type="tns:getStorageArray"/>
>
    <element name="getStorageArrayResponse" type="tns:getStorageArrayResponse"/>
    <element name="getStorageArrays" type="tns:getStorageArrays"/>
    <element name="getStorageArraysResponse" type="tns:getStorageArraysResponse"/>
    <element name="getSwitch" type="tns:getSwitch"/>
    <element name="getSwitchResponse" type="tns:getSwitchResponse"/>
    <element name="getSwitches" type="tns:getSwitches"/>
    <element name="getSwitchesResponse" type="tns:getSwitchesResponse"/>
    <element name="getSynchronizationsByStorageArray" type="tns:getSynchronizationsByStorageArray"/>
    <element name="getSynchronizationsByStorageArrayResponse" type="tns:getSynchronizationsByStorageArrayResponse"/>
    <element name="getTape" type="tns:getTape"/>
    <element name="getTapeResponse" type="tns:getTapeResponse"/>
>

```

```

        <element name="getTapes" type="tns:getTapes"/>
        <element name="getTapesResponse" type="tns:getTapesRe-
sponse"/>
        <element name="getViolations" type="tns:getViolations"/>
        <element name="getViolationsResponse" type="tns:getViola-
tionsResponse"/>
        <element name="getViolationsByHost" type="tns:getViola-
tionsByHost"/>
        <element name="getViolationsByHostResponse" type="tns:get-
ViolationsByHostResponse"/>
        <element name="getVolume" type="tns:getVolume"/>
        <element name="getVolumeResponse" type="tns:getVolumeRe-
sponse"/>
        <element name="getVolumeMapsByStorageArray"
type="tns:getVolumeMapsByStorageArray"/>
        <element name="getVolumeMapsByStorageArrayResponse"
type="tns:getVolumeMapsByStorageArrayResponse"/>
        <element name="getVolumeMasksByStorageArray"
type="tns:getVolumeMasksByStorageArray"/>
        <element name="getVolumeMasksByStorageArrayResponse"
type="tns:getVolumeMasksByStorageArrayResponse"/>
        <element name="getVolumes" type="tns:getVolumes"/>
        <element name="getVolumesResponse" type="tns:getVolumesRe-
sponse"/>
        <element name="getVolumesByStorageArray" type="tns:getVol-
umesByStorageArray"/>
        <element name="getVolumesByStorageArrayResponse"
type="tns:getVolumesByStorageArrayResponse"/>
        <element name="getVolumesSynchStatesByDRPath"
type="tns:getVolumesSynchStatesByDRPath"/>
        <element name="getVolumesSynchStatesByDRPathResponse"
type="tns:getVolumesSynchStatesByDRPathResponse"/>
        <element name="getVolumesSynchStatesBySourceVolume"
type="tns:getVolumesSynchStatesBySourceVolume"/>
        <element name="getVolumesSynchStatesBySourceVolumeResponse"
type="tns:getVolumesSynchStatesBySourceVolumeResponse"/>
        <element name="openSession" type="tns:openSession"/>
        <element name="openSessionResponse" type="tns:openSession-
Response"/>
        <element name="removeHostFromApplication" type="tns:remove-
HostFromApplication"/>

```

```

        <element name="removeHostFromApplicationResponse"
type="tns:removeHostFromApplicationResponse"/>
        <element name="removeHostPolicies" type="tns:removeHostPol-
icies"/>
        <element name="removeHostPoliciesResponse"
type="tns:removeHostPoliciesResponse"/>
        <element name="removePolicies" type="tns:removePolicies"/>
        <element name="removePoliciesResponse" type="tns:removePol-
iciesResponse"/>
        <element name="setApplication" type="tns:setApplication"/>
        <element name="setApplicationResponse" type="tns:setAppli-
cationResponse"/>
        <element name="setHostsPolicy" type="tns:setHostsPolicy"/>
        <element name="setHostsPolicyResponse" type="tns:setHost-
sPolicyResponse"/>
        <element name="setPathsPolicy" type="tns:setPathsPolicy"/>
        <element name="setPathsPolicyResponse" type="tns:setPath-
sPolicyResponse"/></schema>
</types>
<message name="SANSscreenAPIEndPoint_getDeviceResponse">
    <part name="result" element="tns:getDeviceResponse"/>
</message>
<message name="SANSscreenAPIEndPoint_addHostToApplication">
    <part name="parameters" element="tns:addHostToApplication"/>
</message>
<message name="SANSscreenAPIEndPoint_getPortsOfDevice">
    <part name="parameters" element="tns:getPortsOfDevice"/>
</message>
<message name="SANSscreenAPIEndPoint_getViolations">
    <part name="parameters" element="tns:getViolations"/>
</message>
<message name="SANSscreenAPIEndPoint_getDeviceGroups">
    <part name="parameters" element="tns:getDeviceGroups"/>
</message>
<message name="SANSscreenAPIEndPoint_getPortsResponse">
    <part name="result" element="tns:getPortsResponse"/>
</message>
<message name="SANSscreenAPIEndPoint_getStorageArraysResponse">
    <part name="result" element="tns:getStorageArraysResponse"/>
</message>
<message name="SANSscreenAPIEndPoint_getPathsResponse">

```

```

    <part name="result" element="tns:getPathsResponse"/>
</message>
<message name="SANscreenAPIEndPoint_getPathsByHostResponse">
    <part name="result" element="tns:getPathsByHostResponse"/>
</message>
<message name="SANscreenAPIEndPoint_getSwitchesResponse">
    <part name="result" element="tns:getSwitchesResponse"/>
</message>
<message name="SANscreenAPIEndPoint_findObjectsByIpResponse">
    <part name="result" element="tns:findObjectsByIpResponse"/>
</message>
<message name="SANscreenAPIEndPoint_getRecentVolume">
    <part name="parameters" element="tns:getRecentVolume"/>
</message>
<message name="SANscreenAPIEndPoint_getRecentSwitchResponse">
    <part name="result" element="tns:getRecentSwitchResponse"/>
</message>
<message name="SANscreenAPIEndPoint_getNodesOfDevice">
    <part name="parameters" element="tns:getNodesOfDevice"/>
</message>
<message name="SANscreenAPIEndPoint_getPorts">
    <part name="parameters" element="tns:getPorts"/>
</message>
<message name="SANscreenAPIEndPoint_getVolumesByStorageArray">
    <part name="parameters" element="tns:getVolumesByStorageAr-
ray"/>
</message>
<message name="SANscreenAPIEndPoint_getVolumesResponse">
    <part name="result" element="tns:getVolumesResponse"/>
</message>
<message
name="SANscreenAPIEndPoint_getVolumeMapsByStorageArray">
    <part name="parameters" element="tns:getVolumeMapsByStorage-
Array"/>
</message>
<message name="SANscreenAPIEndPoint_getHostPolicy">
    <part name="parameters" element="tns:getHostPolicy"/>
</message>
<message name="SANscreenAPIEndPoint_getVolume">
    <part name="parameters" element="tns:getVolume"/>
</message>

```

```

    <message
name="SANscreenAPIEndPoint_getRecentStorageArrayResponse">
    <part name="result" element="tns:getRecentStorageArrayRe-
sponse"/>
</message>
<message name="SANscreenAPIEndPoint_getAPIVersion">
    <part name="parameters" element="tns:getAPIVersion"/>
</message>
<message name="APIException">
    <part name="APIException" element="tns:APIException"/>
</message>
<message name="SANscreenAPIEndPoint_getHostsOfApplication">
    <part name="parameters" element="tns:getHostsOfApplication"/>
</message>
<message
name="SANscreenAPIEndPoint_getVolumesSynchStatesByDRPath">
    <part name="parameters" element="tns:getVolumesSynchStatesBy-
DRPath"/>
</message>
<message name="SANscreenAPIEndPoint_getGenerics">
    <part name="parameters" element="tns:getGenerics"/>
</message>
<message name="SANscreenAPIEndPoint_getDeviceGroupsByVolume">
    <part name="parameters" element="tns:getDeviceGroupsByVol-
ume"/>
</message>
<message name="SANscreenAPIEndPoint_getApplication">
    <part name="parameters" element="tns:getApplication"/>
</message>
<message name="SANscreenAPIEndPoint_getRecentVolumeResponse">
    <part name="result" element="tns:getRecentVolumeResponse"/>
</message>
<message name="SANscreenAPIEndPoint_getPortsOfNode">
    <part name="parameters" element="tns:getPortsOfNode"/>
</message>
<message
name="SANscreenAPIEndPoint_getDRPathsByVolumeResponse">
    <part name="result" element="tns:getDRPathsByVolumeResponse"/
>
</message>
<message

```

```

name="SANscreenAPIEndPoint_addHostToApplicationResponse">
  <part name="result" element="tns:addHostToApplicationRe-
sponse"/>
</message>
<message name="SANscreenAPIEndPoint_setApplicationResponse">
  <part name="result" element="tns:setApplicationResponse"/>
</message>
<message name="SANscreenAPIEndPoint_getDevice">
  <part name="parameters" element="tns:getDevice"/>
</message>
<message
name="SANscreenAPIEndPoint_getSANscreenVersionResponse">
  <part name="result" element="tns:getSANscreenVersionRe-
sponse"/>
</message>
<message name="SANscreenAPIEndPoint_getRecentTape">
  <part name="parameters" element="tns:getRecentTape"/>
</message>
<message name="SANscreenAPIEndPoint_closeSession">
  <part name="parameters" element="tns:closeSession"/>
</message>
<message name="SANscreenAPIEndPoint_getPathPolicy">
  <part name="parameters" element="tns:getPathPolicy"/>
</message>
<message name="SANscreenAPIEndPoint_getSwitch">
  <part name="parameters" element="tns:getSwitch"/>
</message>
<message name="SANscreenAPIEndPoint_getStorageArrays">
  <part name="parameters" element="tns:getStorageArrays"/>
</message>
<message name="SANscreenAPIEndPoint_getVolumes">
  <part name="parameters" element="tns:getVolumes"/>
</message>
<message name="SANscreenAPIEndPoint_getSwitches">
  <part name="parameters" element="tns:getSwitches"/>
</message>
<message name="SANscreenAPIEndPoint_getTape">
  <part name="parameters" element="tns:getTape"/>
</message>
<message
name="SANscreenAPIEndPoint_getVolumeMasksByStorageArray">

```

```

        <part name="parameters" element="tns:getVolumeMasksByStorage-
Array"/>
    </message>
    <message name="SANscreenAPIEndPoint_getRecentDeviceResponse">
        <part name="result" element="tns:getRecentDeviceResponse"/>
    </message>
    <message name="SANscreenAPIEndPoint_getAPIVersionResponse">
        <part name="result" element="tns:getAPIVersionResponse"/>
    </message>
    <message
name="SANscreenAPIEndPoint_removeHostPoliciesResponse">
        <part name="result" element="tns:removeHostPoliciesResponse"/>
    >
    </message>
    <message name="SANscreenAPIEndPoint_getApplicationResponse">
        <part name="result" element="tns:getApplicationResponse"/>
    </message>
    <message name="SANscreenAPIEndPoint_closeSessionResponse">
        <part name="result" element="tns:closeSessionResponse"/>
    </message>
    <message
name="SANscreenAPIEndPoint_getPhysicalPortByLogicalPort">
        <part name="parameters" element="tns:getPhysicalPortByLogi-
calPort"/>
    </message>
    <message name="SANscreenAPIEndPoint_findObjectByWwn">
        <part name="parameters" element="tns:findObjectByWwn"/>
    </message>
    <message name="SANscreenAPIEndPoint_getPathPolicyResponse">
        <part name="result" element="tns:getPathPolicyResponse"/>
    </message>
    <message name="SANscreenAPIEndPoint_findObjectsByName">
        <part name="parameters" element="tns:findObjectsByName"/>
    </message>
    <message name="SANscreenAPIEndPoint_setPathsPolicy">
        <part name="parameters" element="tns:setPathsPolicy"/>
    </message>
    <message name="SANscreenAPIEndPoint_getHostPolicyResponse">
        <part name="result" element="tns:getHostPolicyResponse"/>
    </message>
    <message

```

```

name="SANscreenAPIEndPoint_getAllApplicationsResponse">
  <part name="result" element="tns:getAllApplicationsResponse"/>
</message>
<message name="SANscreenAPIEndPoint_getPaths">
  <part name="parameters" element="tns:getPaths"/>
</message>
<message name="SANscreenAPIEndPoint_getPortsOfDeviceResponse">
  <part name="result" element="tns:getPortsOfDeviceResponse"/>
</message>
<message name="SANscreenAPIEndPoint_getViolationsResponse">
  <part name="result" element="tns:getViolationsResponse"/>
</message>
<message name="SANscreenAPIEndPoint_getStorageArrayResponse">
  <part name="result" element="tns:getStorageArrayResponse"/>
</message>
<message
name="SANscreenAPIEndPoint_getApplicationsOfHostResponse">
  <part name="result" element="tns:getApplicationsOfHostRe-
sponse"/>
</message>
<message
name="SANscreenAPIEndPoint_getLogicalPortsByPhysicalSwitchRespons
e">
  <part name="result" element="tns:getLogicalPortsByPhysi-
calSwitchResponse"/>
</message>
<message name="SANscreenAPIEndPoint_removeHostPolicies">
  <part name="parameters" element="tns:removeHostPolicies"/>
</message>
<message
name="SANscreenAPIEndPoint_removeHostFromApplicationResponse">
  <part name="result" element="tns:removeHostFromApplicationRe-
sponse"/>
</message>
<message name="SANscreenAPIEndPoint_setHostsPolicy">
  <part name="parameters" element="tns:setHostsPolicy"/>
</message>
<message
name="SANscreenAPIEndPoint_getDeviceGroupsByStorageResponse">
  <part name="result" element="tns:getDeviceGroupsByStorageRe-

```

```
sponse"/>
  </message>
  <message name="SANscreenAPIEndPoint_getRecentDevice">
    <part name="parameters" element="tns:getRecentDevice"/>
  </message>
  <message
name="SANscreenAPIEndPoint_getLogicalPortsByPhysicalPort">
    <part name="parameters" element="tns:getLogicalPortsByPhysi-
calPort"/>
  </message>
  <message name="SANscreenAPIEndPoint_getHostsResponse">
    <part name="result" element="tns:getHostsResponse"/>
  </message>
  <message
name="SANscreenAPIEndPoint_getVolumesSynchStatesByDRPathResponse"
>
    <part name="result" element="tns:getVolumesSynchStatesBy-
DRPathResponse"/>
  </message>
  <message name="SANscreenAPIEndPoint_openSession">
    <part name="parameters" element="tns:openSession"/>
  </message>
  <message name="SANscreenAPIEndPoint_getTapesResponse">
    <part name="result" element="tns:getTapesResponse"/>
  </message>
  <message
name="SANscreenAPIEndPoint_getHostsOfApplicationResponse">
    <part name="result" element="tns:getHostsOfApplicationRe-
sponse"/>
  </message>
  <message
name="SANscreenAPIEndPoint_getLogicalPortsByPhysicalSwitch">
    <part name="parameters" element="tns:getLogicalPortsByPhysi-
calSwitch"/>
  </message>
  <message name="SANscreenAPIEndPoint_getGenericResponse">
    <part name="result" element="tns:getGenericResponse"/>
  </message>
  <message name="SANscreenAPIEndPoint_getDeviceGroupResponse">
    <part name="result" element="tns:getDeviceGroupResponse"/>
  </message>
```

```

<message name="SANscreenAPIEndPoint_getTapes">
  <part name="parameters" element="tns:getTapes"/>
</message>
<message name="SANscreenAPIEndPoint_getVolumeResponse">
  <part name="result" element="tns:getVolumeResponse"/>
</message>
<message name="SANscreenAPIEndPoint_removeHostFromApplication">
  <part name="parameters" element="tns:removeHostFromApplica-
tion"/>
</message>
<message name="SANscreenAPIEndPoint_getPathsByHost">
  <part name="parameters" element="tns:getPathsByHost"/>
</message>
<message
name="SANscreenAPIEndPoint_getLogicalPortsByPhysicalPortResponse"
>
  <part name="result" element="tns:getLogicalPortsByPhysi-
calPortResponse"/>
</message>
<message name="SANscreenAPIEndPoint_getHost">
  <part name="parameters" element="tns:getHost"/>
</message>
<message
name="SANscreenAPIEndPoint_getLogicalSwitchesByPhysicalSwitchResp
onse">
  <part name="result" element="tns:getLogicalSwitchesByPhysi-
calSwitchResponse"/>
</message>
<message
name="SANscreenAPIEndPoint_getVolumesSynchStatesBySourceVolumeRes
ponse">
  <part name="result" element="tns:getVolumesSynchStatesBy-
SourceVolumeResponse"/>
</message>
<message name="SANscreenAPIEndPoint_getHosts">
  <part name="parameters" element="tns:getHosts"/>
</message>
<message name="SANscreenAPIEndPoint_getDeviceGroup">
  <part name="parameters" element="tns:getDeviceGroup"/>
</message>
<message name="SANscreenAPIEndPoint_getDeviceGroupsByStorage">

```

```

        <part name="parameters" element="tns:getDeviceGroupsByStorage"/>
    </message>
    <message name="SANscreenAPIEndPoint_getDeviceGroupVolumes">
        <part name="parameters" element="tns:getDeviceGroupVolumes"/>
    </message>
    <message
name="SANscreenAPIEndPoint_getVolumesSynchStatesBySourceVolume">
        <part name="parameters" element="tns:getVolumesSynchStatesBySourceVolume"/>
    </message>
    <message name="SANscreenAPIEndPoint_findObjectsByNameResponse">
        <part name="result" element="tns:findObjectsByNameResponse"/>
    </message>
    <message name="SANscreenAPIEndPoint_getRecentSwitch">
        <part name="parameters" element="tns:getRecentSwitch"/>
    </message>
    <message name="SANscreenAPIEndPoint_getGeneric">
        <part name="parameters" element="tns:getGeneric"/>
    </message>
    <message name="SANscreenAPIEndPoint_getConnectedPorts">
        <part name="parameters" element="tns:getConnectedPorts"/>
    </message>
    <message name="SANscreenAPIEndPoint_getNodesOfDeviceResponse">
        <part name="result" element="tns:getNodesOfDeviceResponse"/>
    </message>
    <message
name="SANscreenAPIEndPoint_getPhysicalPortByLogicalPortResponse">
        <part name="result" element="tns:getPhysicalPortByLogicalPortResponse"/>
    </message>
    <message name="SANscreenAPIEndPoint_getDeviceGroupsResponse">
        <part name="result" element="tns:getDeviceGroupsResponse"/>
    </message>
    <message
name="SANscreenAPIEndPoint_getVolumeMasksByStorageArrayResponse">
        <part name="result" element="tns:getVolumeMasksByStorageArrayResponse"/>
    </message>
    <message name="SANscreenAPIEndPoint_findObjectByWwnResponse">
        <part name="result" element="tns:findObjectByWwnResponse"/>

```

```

    </message>
    <message
name="SANscreenAPIEndPoint_getVolumeMapsByStorageArrayResponse">
    <part name="result" element="tns:getVolumeMapsByStorageArray-
Response"/>
    </message>
    <message name="SANscreenAPIEndPoint_getAllApplications">
    <part name="parameters" element="tns:getAllApplications"/>
    </message>
    <message name="SANscreenAPIEndPoint_getGenericsResponse">
    <part name="result" element="tns:getGenericsResponse"/>
    </message>
    <message name="SANscreenAPIEndPoint_getDRPathsByVolume">
    <part name="parameters" element="tns:getDRPathsByVolume"/>
    </message>
    <message name="SANscreenAPIEndPoint_getRecentHost">
    <part name="parameters" element="tns:getRecentHost"/>
    </message>
    <message name="SANscreenAPIEndPoint_getHostResponse">
    <part name="result" element="tns:getHostResponse"/>
    </message>
    <message
name="SANscreenAPIEndPoint_getLogicalSwitchesByPhysicalSwitch">
    <part name="parameters" element="tns:getLogicalSwitchesBy-
PhysicalSwitch"/>
    </message>
    <message name="SANscreenAPIEndPoint_getConnectedPortsResponse">
    <part name="result" element="tns:getConnectedPortsResponse"/>
    </message>
    <message name="SANscreenAPIEndPoint_getPolicies">
    <part name="parameters" element="tns:getPolicies"/>
    </message>
    <message name="SANscreenAPIEndPoint_getPoliciesResponse">
    <part name="result" element="tns:getPoliciesResponse"/>
    </message>
    <message name="SANscreenAPIEndPoint_getPortsOfNodeResponse">
    <part name="result" element="tns:getPortsOfNodeResponse"/>
    </message>
    <message name="SANscreenAPIEndPoint_setPathsPolicyResponse">
    <part name="result" element="tns:setPathsPolicyResponse"/>
    </message>

```

```

<message name="SANscreenAPIEndPoint_removePoliciesResponse">
  <part name="result" element="tns:removePoliciesResponse"/>
</message>
<message name="SANscreenAPIEndPoint_openSessionResponse">
  <part name="result" element="tns:openSessionResponse"/>
</message>
<message name="SANscreenAPIEndPoint_getPortResponse">
  <part name="result" element="tns:getPortResponse"/>
</message>
<message
name="SANscreenAPIEndPoint_getDeviceGroupsByVolumeResponse">
  <part name="result" element="tns:getDeviceGroupsByVolumeRe-
sponse"/>
</message>
<message
name="SANscreenAPIEndPoint_getViolationsByHostResponse">
  <part name="result" element="tns:getViolationsByHostRe-
sponse"/>
</message>
<message name="SANscreenAPIEndPoint_getTapeResponse">
  <part name="result" element="tns:getTapeResponse"/>
</message>
<message name="SANscreenAPIEndPoint_getApplicationsOfHost">
  <part name="parameters" element="tns:getApplicationsOfHost"/>
</message>
<message
name="SANscreenAPIEndPoint_getPortsOfDeviceTypeResponse">
  <part name="result" element="tns:getPortsOfDeviceTypeRe-
sponse"/>
</message>
<message name="SANscreenAPIEndPoint_getRecentHostResponse">
  <part name="result" element="tns:getRecentHostResponse"/>
</message>
<message name="APIBadRequestException">
  <part name="APIBadRequestException" element="tns:APIBa-
dRequestException"/>
</message>
<message
name="SANscreenAPIEndPoint_getDeviceGroupVolumesResponse">
  <part name="result" element="tns:getDeviceGroupVolumesRe-
sponse"/>

```

```

</message>
<message name="SANscreenAPIEndPoint_getRecentTapeResponse">
  <part name="result" element="tns:getRecentTapeResponse"/>
</message>
<message
name="SANscreenAPIEndPoint_getLogicalPortsByLogicalSwitchResponse
">
  <part name="result" element="tns:getLogicalPortsByLogi-
calSwitchResponse"/>
</message>
<message name="SANscreenAPIEndPoint_removePolicies">
  <part name="parameters" element="tns:removePolicies"/>
</message>
<message
name="SANscreenAPIEndPoint_getVolumesByStorageArrayResponse">
  <part name="result" element="tns:getVolumesByStorageArrayRe-
sponse"/>
</message>
<message name="SANscreenAPIEndPoint_getSwitchResponse">
  <part name="result" element="tns:getSwitchResponse"/>
</message>
<message
name="SANscreenAPIEndPoint_getSynchronizationsByStorageArray">
  <part name="parameters" element="tns:getSynchronizationsBy-
StorageArray"/>
</message>
<message name="SANscreenAPIEndPoint_findObjectsByIp">
  <part name="parameters" element="tns:findObjectsByIp"/>
</message>
<message name="SANscreenAPIEndPoint_getPortsOfDeviceType">
  <part name="parameters" element="tns:getPortsOfDeviceType"/>
</message>
<message name="SANscreenAPIEndPoint_setApplication">
  <part name="parameters" element="tns:setApplication"/>
</message>
<message name="SANscreenAPIEndPoint_getRecentStorageArray">
  <part name="parameters" element="tns:getRecentStorageArray"/>
</message>
<message name="SANscreenAPIEndPoint_getPort">
  <part name="parameters" element="tns:getPort"/>
</message>

```

```

    <message
name="SANscreenAPIEndPoint_getSynchronizationsByStorageArrayResponse">
    <part name="result" element="tns:getSynchronizationsByStorageArrayResponse"/>
</message>
<message name="SANscreenAPIEndPoint_getStorageArray">
    <part name="parameters" element="tns:getStorageArray"/>
</message>
<message
name="SANscreenAPIEndPoint_getLogicalPortsByLogicalSwitch">
    <part name="parameters" element="tns:getLogicalPortsByLogicalSwitch"/>
</message>
<message name="SANscreenAPIEndPoint_getSANscreenVersion">
    <part name="parameters" element="tns:getSANscreenVersion"/>
</message>
<message name="SANscreenAPIEndPoint_getViolationsByHost">
    <part name="parameters" element="tns:getViolationsByHost"/>
</message>
<message name="SANscreenAPIEndPoint_setHostsPolicyResponse">
    <part name="result" element="tns:setHostsPolicyResponse"/>
</message>
<portType name="SANscreenAPIEndPoint">
    <operation name="addHostToApplication">
        <input message="tns:SANscreenAPIEndPoint_addHostToApplication"/>
        <output message="tns:SANscreenAPIEndPoint_addHostToApplicationResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBadRequestException"/>
    </operation>
    <operation name="closeSession">
        <input message="tns:SANscreenAPIEndPoint_closeSession"/>
        <output message="tns:SANscreenAPIEndPoint_closeSessionResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBadRequestException"/>
    </operation>

```

```

        <operation name="findObjectByWwn">
            <input message="tns:SANSscreenAPIEndPoint_findObjectByWwn"/>
            <output mes-
sage="tns:SANSscreenAPIEndPoint_findObjectByWwnResponse"/>
            <fault name="APIException" message="tns:APIException"/>
            <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
        </operation>
        <operation name="findObjectsByIp">
            <input message="tns:SANSscreenAPIEndPoint_findObjectsByIp"/>
            <output mes-
sage="tns:SANSscreenAPIEndPoint_findObjectsByIpResponse"/>
            <fault name="APIException" message="tns:APIException"/>
            <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
        </operation>
        <operation name="findObjectsByName">
            <input mes-
sage="tns:SANSscreenAPIEndPoint_findObjectsByName"/>
            <output mes-
sage="tns:SANSscreenAPIEndPoint_findObjectsByNameResponse"/>
            <fault name="APIException" message="tns:APIException"/>
            <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
        </operation>
        <operation name="getAPIVersion">
            <input message="tns:SANSscreenAPIEndPoint_getAPIVersion"/>
            <output mes-
sage="tns:SANSscreenAPIEndPoint_getAPIVersionResponse"/>
        </operation>
        <operation name="getAllApplications">
            <input mes-
sage="tns:SANSscreenAPIEndPoint_getAllApplications"/>
            <output mes-
sage="tns:SANSscreenAPIEndPoint_getAllApplicationsResponse"/>
            <fault name="APIException" message="tns:APIException"/>
            <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
        </operation>
        <operation name="getApplication">
            <input message="tns:SANSscreenAPIEndPoint_getApplication"/>

```

```

        <output mes-
sage="tns:SANscreenAPIEndPoint_getApplicationResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getApplicationsOfHost">
        <input mes-
sage="tns:SANscreenAPIEndPoint_getApplicationsOfHost"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getApplicationsOfHostResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getConnectedPorts">
        <input mes-
sage="tns:SANscreenAPIEndPoint_getConnectedPorts"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getConnectedPortsResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getDRPathsByVolume">
        <input mes-
sage="tns:SANscreenAPIEndPoint_getDRPathsByVolume"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getDRPathsByVolumeResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getDevice">
        <input message="tns:SANscreenAPIEndPoint_getDevice"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getDeviceResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>

```

```

    <operation name="getDeviceGroup">
      <input message="tns:SANscreenAPIEndPoint_getDeviceGroup"/>
      <output mes-
sage="tns:SANscreenAPIEndPoint_getDeviceGroupResponse"/>
      <fault name="APIException" message="tns:APIException"/>
      <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getDeviceGroupVolumes">
      <input mes-
sage="tns:SANscreenAPIEndPoint_getDeviceGroupVolumes"/>
      <output mes-
sage="tns:SANscreenAPIEndPoint_getDeviceGroupVolumesResponse"/>
      <fault name="APIException" message="tns:APIException"/>
      <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getDeviceGroups">
      <input message="tns:SANscreenAPIEndPoint_getDeviceGroups"/>
      <output mes-
sage="tns:SANscreenAPIEndPoint_getDeviceGroupsResponse"/>
      <fault name="APIException" message="tns:APIException"/>
      <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getDeviceGroupsByStorage">
      <input mes-
sage="tns:SANscreenAPIEndPoint_getDeviceGroupsByStorage"/>
      <output mes-
sage="tns:SANscreenAPIEndPoint_getDeviceGroupsByStorageResponse"/
>
      <fault name="APIException" message="tns:APIException"/>
      <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getDeviceGroupsByVolume">
      <input mes-
sage="tns:SANscreenAPIEndPoint_getDeviceGroupsByVolume"/>
      <output mes-
sage="tns:SANscreenAPIEndPoint_getDeviceGroupsByVolumeResponse"/>
      <fault name="APIException" message="tns:APIException"/>

```

```

        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getGeneric">
        <input message="tns:SANscreenAPIEndPoint_getGeneric"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getGenericResponse"/>
        <fault name="APIException" message="tns:APIException"/>
    </operation>
    <operation name="getGenerics">
        <input message="tns:SANscreenAPIEndPoint_getGenerics"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getGenericsResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getHost">
        <input message="tns:SANscreenAPIEndPoint_getHost"/>
        <output message="tns:SANscreenAPIEndPoint_getHostResponse"/>
    >
        <fault name="APIException" message="tns:APIException"/>
    </operation>
    <operation name="getHostPolicy">
        <input message="tns:SANscreenAPIEndPoint_getHostPolicy"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getHostPolicyResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getHosts">
        <input message="tns:SANscreenAPIEndPoint_getHosts"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getHostsResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getHostsOfApplication">
        <input mes-

```

```

sage="tns:SANscreenAPIEndPoint_getHostsOfApplication"/>
    <output mes-
sage="tns:SANscreenAPIEndPoint_getHostsOfApplicationResponse"/>
    <fault name="APIException" message="tns:APIException"/>
    <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getLogicalPortsByLogicalSwitch">
    <input mes-
sage="tns:SANscreenAPIEndPoint_getLogicalPortsByLogicalSwitch"/>
    <output mes-
sage="tns:SANscreenAPIEndPoint_getLogicalPortsByLogicalSwitchResp
onse"/>
    <fault name="APIException" message="tns:APIException"/>
    <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getLogicalPortsByPhysicalPort">
    <input mes-
sage="tns:SANscreenAPIEndPoint_getLogicalPortsByPhysicalPort"/>
    <output mes-
sage="tns:SANscreenAPIEndPoint_getLogicalPortsByPhysicalPortRespo
nse"/>
    <fault name="APIException" message="tns:APIException"/>
    <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getLogicalPortsByPhysicalSwitch">
    <input mes-
sage="tns:SANscreenAPIEndPoint_getLogicalPortsByPhysicalSwitch"/>
    <output mes-
sage="tns:SANscreenAPIEndPoint_getLogicalPortsByPhysicalSwitchRes
ponse"/>
    <fault name="APIException" message="tns:APIException"/>
    <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getLogicalSwitchesByPhysicalSwitch">
    <input mes-
sage="tns:SANscreenAPIEndPoint_getLogicalSwitchesByPhysicalSwitch
"/>

```

```

        <output mes-
sage="tns:SANscreenAPIEndPoint_getLogicalSwitchesByPhysicalSwitch
Response"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getNodesOfDevice">
        <input message="tns:SANscreenAPIEndPoint_getNodesOfDevice"/>
    >
        <output mes-
sage="tns:SANscreenAPIEndPoint_getNodesOfDeviceResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getPathPolicy">
        <input message="tns:SANscreenAPIEndPoint_getPathPolicy"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getPathPolicyResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getPaths">
        <input message="tns:SANscreenAPIEndPoint_getPaths"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getPathsResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getPathsByHost">
        <input message="tns:SANscreenAPIEndPoint_getPathsByHost"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getPathsByHostResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getPhysicalPortByLogicalPort">

```

```

        <input mes-
sage="tns:SANscreenAPIEndPoint_getPhysicalPortByLogicalPort"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getPhysicalPortByLogicalPortRespon
se"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getPolicies">
        <input message="tns:SANscreenAPIEndPoint_getPolicies"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getPoliciesResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getPort">
        <input message="tns:SANscreenAPIEndPoint_getPort"/>
        <output message="tns:SANscreenAPIEndPoint_getPortResponse"/
>
        <fault name="APIException" message="tns:APIException"/>
    </operation>
    <operation name="getPorts">
        <input message="tns:SANscreenAPIEndPoint_getPorts"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getPortsResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getPortsOfDevice">
        <input message="tns:SANscreenAPIEndPoint_getPortsOfDevice"/
>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getPortsOfDeviceResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getPortsOfDeviceType">

```

```

        <input mes-
sage="tns:SANscreenAPIEndPoint_getPortsOfDeviceType"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getPortsOfDeviceTypeResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getPortsOfNode">
        <input message="tns:SANscreenAPIEndPoint_getPortsOfNode"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getPortsOfNodeResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getRecentDevice">
        <input message="tns:SANscreenAPIEndPoint_getRecentDevice"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getRecentDeviceResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getRecentHost">
        <input message="tns:SANscreenAPIEndPoint_getRecentHost"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getRecentHostResponse"/>
        <fault name="APIException" message="tns:APIException"/>
    </operation>
    <operation name="getRecentStorageArray">
        <input mes-
sage="tns:SANscreenAPIEndPoint_getRecentStorageArray"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getRecentStorageArrayResponse"/>
        <fault name="APIException" message="tns:APIException"/>
    </operation>
    <operation name="getRecentSwitch">
        <input message="tns:SANscreenAPIEndPoint_getRecentSwitch"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getRecentSwitchResponse"/>

```

```

        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getRecentTape">
        <input message="tns:SANSscreenAPIEndPoint_getRecentTape"/>
        <output mes-
sage="tns:SANSscreenAPIEndPoint_getRecentTapeResponse"/>
        <fault name="APIException" message="tns:APIException"/>
    </operation>
    <operation name="getRecentVolume">
        <input message="tns:SANSscreenAPIEndPoint_getRecentVolume"/>
        <output mes-
sage="tns:SANSscreenAPIEndPoint_getRecentVolumeResponse"/>
        <fault name="APIException" message="tns:APIException"/>
    </operation>
    <operation name="getSANSscreenVersion">
        <input mes-
sage="tns:SANSscreenAPIEndPoint_getSANSscreenVersion"/>
        <output mes-
sage="tns:SANSscreenAPIEndPoint_getSANSscreenVersionResponse"/>
    </operation>
    <operation name="getStorageArray">
        <input message="tns:SANSscreenAPIEndPoint_getStorageArray"/>
        <output mes-
sage="tns:SANSscreenAPIEndPoint_getStorageArrayResponse"/>
        <fault name="APIException" message="tns:APIException"/>
    </operation>
    <operation name="getStorageArrays">
        <input message="tns:SANSscreenAPIEndPoint_getStorageArrays"/
>
        <output mes-
sage="tns:SANSscreenAPIEndPoint_getStorageArraysResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getSwitch">
        <input message="tns:SANSscreenAPIEndPoint_getSwitch"/>
        <output mes-
sage="tns:SANSscreenAPIEndPoint_getSwitchResponse"/>

```

```

        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getSwitches">
        <input message="tns:SANscreenAPIEndPoint_getSwitches"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getSwitchesResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getSynchronizationsByStorageArray">
        <input mes-
sage="tns:SANscreenAPIEndPoint_getSynchronizationsByStorageArray"
/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getSynchronizationsByStorageArrayR
esponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getTape">
        <input message="tns:SANscreenAPIEndPoint_getTape"/>
        <output message="tns:SANscreenAPIEndPoint_getTapeResponse"/
>
        <fault name="APIException" message="tns:APIException"/>
    </operation>
    <operation name="getTapes">
        <input message="tns:SANscreenAPIEndPoint_getTapes"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getTapesResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getViolations">
        <input message="tns:SANscreenAPIEndPoint_getViolations"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getViolationsResponse"/>

```

```

        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getViolationsByHost">
        <input mes-
sage="tns:SANscreenAPIEndPoint_getViolationsByHost"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getViolationsByHostResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getVolume">
        <input message="tns:SANscreenAPIEndPoint_getVolume"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getVolumeResponse"/>
        <fault name="APIException" message="tns:APIException"/>
    </operation>
    <operation name="getVolumeMapsByStorageArray">
        <input mes-
sage="tns:SANscreenAPIEndPoint_getVolumeMapsByStorageArray"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getVolumeMapsByStorageArrayRespons
e"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getVolumeMasksByStorageArray">
        <input mes-
sage="tns:SANscreenAPIEndPoint_getVolumeMasksByStorageArray"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getVolumeMasksByStorageArrayRespon
se"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getVolumes">
        <input message="tns:SANscreenAPIEndPoint_getVolumes"/>

```

```

        <output mes-
sage="tns:SANscreenAPIEndPoint_getVolumesResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getVolumesByStorageArray">
        <input mes-
sage="tns:SANscreenAPIEndPoint_getVolumesByStorageArray"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getVolumesByStorageArrayResponse"/
>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getVolumesSynchStatesByDRPath">
        <input mes-
sage="tns:SANscreenAPIEndPoint_getVolumesSynchStatesByDRPath"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getVolumesSynchStatesByDRPathRespo
nse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="getVolumesSynchStatesBySourceVolume">
        <input mes-
sage="tns:SANscreenAPIEndPoint_getVolumesSynchStatesBySourceVolum
e"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_getVolumesSynchStatesBySourceVolum
eResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="openSession">
        <input message="tns:SANscreenAPIEndPoint_openSession"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_openSessionResponse"/>

```

```

        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="removeHostFromApplication">
        <input mes-
sage="tns:SANscreenAPIEndPoint_removeHostFromApplication"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_removeHostFromApplicationResponse"
/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="removeHostPolicies">
        <input mes-
sage="tns:SANscreenAPIEndPoint_removeHostPolicies"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_removeHostPoliciesResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="removePolicies">
        <input message="tns:SANscreenAPIEndPoint_removePolicies"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_removePoliciesResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="setApplication">
        <input message="tns:SANscreenAPIEndPoint_setApplication"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_setApplicationResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="setHostsPolicy">
        <input message="tns:SANscreenAPIEndPoint_setHostsPolicy"/>

```

```

        <output mes-
sage="tns:SANscreenAPIEndPoint_setHostsPolicyResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
    <operation name="setPathsPolicy">
        <input message="tns:SANscreenAPIEndPoint_setPathsPolicy"/>
        <output mes-
sage="tns:SANscreenAPIEndPoint_setPathsPolicyResponse"/>
        <fault name="APIException" message="tns:APIException"/>
        <fault name="APIBadRequestException" message="tns:APIBa-
dRequestException"/>
    </operation>
</portType>
<binding name="SANscreenAPIEndPointBinding" type="tns:SAN-
screenAPIEndPoint">
    <soap:binding style="document" transport="http://schemas.xml-
soap.org/soap/http"/>
    <operation name="addHostToApplication">
        <soap:operation soapAction=""/>
        <input>
            <soap:body use="literal"/>
        </input>
        <output>
            <soap:body use="literal"/>
        </output>
        <fault name="APIException">
            <soap:fault name="APIException" use="literal"/>
        </fault>
        <fault name="APIBadRequestException">
            <soap:fault name="APIBadRequestException" use="literal"/>
        </fault>
    </operation>
    <operation name="closeSession">
        <soap:operation soapAction=""/>
        <input>
            <soap:body use="literal"/>
        </input>
        <output>
            <soap:body use="literal"/>

```

```
</output>
<fault name="APIException">
  <soap:fault name="APIException" use="literal"/>
</fault>
<fault name="APIBadRequestException">
  <soap:fault name="APIBadRequestException" use="literal"/>
</fault>
</operation>
<operation name="findObjectByWwn">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="findObjectsByIp">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="findObjectsByName">
  <soap:operation soapAction=""/>
  <input>
```

```
        <soap:body use="literal"/>
    </input>
    <output>
        <soap:body use="literal"/>
    </output>
    <fault name="APIException">
        <soap:fault name="APIException" use="literal"/>
    </fault>
    <fault name="APIBadRequestException">
        <soap:fault name="APIBadRequestException" use="literal"/>
    </fault>
</operation>
<operation name="getAPIVersion">
    <soap:operation soapAction=""/>
    <input>
        <soap:body use="literal"/>
    </input>
    <output>
        <soap:body use="literal"/>
    </output>
</operation>
<operation name="getAllApplications">
    <soap:operation soapAction=""/>
    <input>
        <soap:body use="literal"/>
    </input>
    <output>
        <soap:body use="literal"/>
    </output>
    <fault name="APIException">
        <soap:fault name="APIException" use="literal"/>
    </fault>
    <fault name="APIBadRequestException">
        <soap:fault name="APIBadRequestException" use="literal"/>
    </fault>
</operation>
<operation name="getApplication">
    <soap:operation soapAction=""/>
    <input>
        <soap:body use="literal"/>
    </input>
```

```
<output>
  <soap:body use="literal"/>
</output>
<fault name="APIException">
  <soap:fault name="APIException" use="literal"/>
</fault>
<fault name="APIBadRequestException">
  <soap:fault name="APIBadRequestException" use="literal"/>
</fault>
</operation>
<operation name="getApplicationsOfHost">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getConnectedPorts">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getDRPathsByVolume">
```

```
<soap:operation soapAction=""/>
<input>
  <soap:body use="literal"/>
</input>
<output>
  <soap:body use="literal"/>
</output>
<fault name="APIException">
  <soap:fault name="APIException" use="literal"/>
</fault>
<fault name="APIBadRequestException">
  <soap:fault name="APIBadRequestException" use="literal"/>
</fault>
</operation>
<operation name="getDevice">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getDeviceGroup">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
```

```

        <soap:fault name="APIBadRequestException" use="literal"/>
    </fault>
</operation>
<operation name="getDeviceGroupVolumes">
    <soap:operation soapAction=""/>
    <input>
        <soap:body use="literal"/>
    </input>
    <output>
        <soap:body use="literal"/>
    </output>
    <fault name="APIException">
        <soap:fault name="APIException" use="literal"/>
    </fault>
    <fault name="APIBadRequestException">
        <soap:fault name="APIBadRequestException" use="literal"/>
    </fault>
</operation>
<operation name="getDeviceGroups">
    <soap:operation soapAction=""/>
    <input>
        <soap:body use="literal"/>
    </input>
    <output>
        <soap:body use="literal"/>
    </output>
    <fault name="APIException">
        <soap:fault name="APIException" use="literal"/>
    </fault>
    <fault name="APIBadRequestException">
        <soap:fault name="APIBadRequestException" use="literal"/>
    </fault>
</operation>
<operation name="getDeviceGroupsByStorage">
    <soap:operation soapAction=""/>
    <input>
        <soap:body use="literal"/>
    </input>
    <output>
        <soap:body use="literal"/>
    </output>

```

```
<fault name="APIException">
  <soap:fault name="APIException" use="literal"/>
</fault>
<fault name="APIBadRequestException">
  <soap:fault name="APIBadRequestException" use="literal"/>
</fault>
</operation>
<operation name="getDeviceGroupsByVolume">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getGeneric">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
</operation>
<operation name="getGenerics">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
```

```
</output>
<fault name="APIException">
  <soap:fault name="APIException" use="literal"/>
</fault>
<fault name="APIBadRequestException">
  <soap:fault name="APIBadRequestException" use="literal"/>
</fault>
</operation>
<operation name="getHost">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
</operation>
<operation name="getHostPolicy">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getHosts">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
```

```
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getHostsOfApplication">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getLogicalPortsByLogicalSwitch">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getLogicalPortsByPhysicalPort">
  <soap:operation soapAction=""/>
```

```
<input>
  <soap:body use="literal"/>
</input>
<output>
  <soap:body use="literal"/>
</output>
<fault name="APIException">
  <soap:fault name="APIException" use="literal"/>
</fault>
<fault name="APIBadRequestException">
  <soap:fault name="APIBadRequestException" use="literal"/>
</fault>
</operation>
<operation name="getLogicalPortsByPhysicalSwitch">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getLogicalSwitchesByPhysicalSwitch">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
```

```
    </fault>
  </operation>
  <operation name="getNodesOfDevice">
    <soap:operation soapAction=""/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
    <fault name="APIException">
      <soap:fault name="APIException" use="literal"/>
    </fault>
    <fault name="APIBadRequestException">
      <soap:fault name="APIBadRequestException" use="literal"/>
    </fault>
  </operation>
  <operation name="getPathPolicy">
    <soap:operation soapAction=""/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
    <fault name="APIException">
      <soap:fault name="APIException" use="literal"/>
    </fault>
    <fault name="APIBadRequestException">
      <soap:fault name="APIBadRequestException" use="literal"/>
    </fault>
  </operation>
  <operation name="getPaths">
    <soap:operation soapAction=""/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
    <fault name="APIException">
```

```
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getPathsByHost">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getPhysicalPortByLogicalPort">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getPolicies">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
```

```
<output>
  <soap:body use="literal"/>
</output>
<fault name="APIException">
  <soap:fault name="APIException" use="literal"/>
</fault>
<fault name="APIBadRequestException">
  <soap:fault name="APIBadRequestException" use="literal"/>
</fault>
</operation>
<operation name="getPort">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
</operation>
<operation name="getPorts">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getPortsOfDevice">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
```

```
</input>
<output>
  <soap:body use="literal"/>
</output>
<fault name="APIException">
  <soap:fault name="APIException" use="literal"/>
</fault>
<fault name="APIBadRequestException">
  <soap:fault name="APIBadRequestException" use="literal"/>
</fault>
</operation>
<operation name="getPortsOfDeviceType">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getPortsOfNode">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
```

```
<operation name="getRecentDevice">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getRecentHost">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
</operation>
<operation name="getRecentStorageArray">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
</operation>
<operation name="getRecentSwitch">
  <soap:operation soapAction=""/>
```

```
<input>
  <soap:body use="literal"/>
</input>
<output>
  <soap:body use="literal"/>
</output>
<fault name="APIException">
  <soap:fault name="APIException" use="literal"/>
</fault>
<fault name="APIBadRequestException">
  <soap:fault name="APIBadRequestException" use="literal"/>
</fault>
</operation>
<operation name="getRecentTape">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
</operation>
<operation name="getRecentVolume">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
</operation>
<operation name="getSANSscreenVersion">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
```

```
</input>
<output>
  <soap:body use="literal"/>
</output>
</operation>
<operation name="getStorageArray">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
</operation>
<operation name="getStorageArrays">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getSwitch">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
```

```
        <soap:fault name="APIException" use="literal"/>
    </fault>
    <fault name="APIBadRequestException">
        <soap:fault name="APIBadRequestException" use="literal"/>
    </fault>
</operation>
<operation name="getSwitches">
    <soap:operation soapAction=""/>
    <input>
        <soap:body use="literal"/>
    </input>
    <output>
        <soap:body use="literal"/>
    </output>
    <fault name="APIException">
        <soap:fault name="APIException" use="literal"/>
    </fault>
    <fault name="APIBadRequestException">
        <soap:fault name="APIBadRequestException" use="literal"/>
    </fault>
</operation>
<operation name="getSynchronizationsByStorageArray">
    <soap:operation soapAction=""/>
    <input>
        <soap:body use="literal"/>
    </input>
    <output>
        <soap:body use="literal"/>
    </output>
    <fault name="APIException">
        <soap:fault name="APIException" use="literal"/>
    </fault>
    <fault name="APIBadRequestException">
        <soap:fault name="APIBadRequestException" use="literal"/>
    </fault>
</operation>
<operation name="getTape">
    <soap:operation soapAction=""/>
    <input>
        <soap:body use="literal"/>
    </input>
```

```
<output>
  <soap:body use="literal"/>
</output>
<fault name="APIException">
  <soap:fault name="APIException" use="literal"/>
</fault>
</operation>
<operation name="getTapes">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getViolations">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getViolationsByHost">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
```

```

    </input>
    <output>
      <soap:body use="literal"/>
    </output>
    <fault name="APIException">
      <soap:fault name="APIException" use="literal"/>
    </fault>
    <fault name="APIBadRequestException">
      <soap:fault name="APIBadRequestException" use="literal"/>
    </fault>
  </operation>
  <operation name="getVolume">
    <soap:operation soapAction=""/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
    <fault name="APIException">
      <soap:fault name="APIException" use="literal"/>
    </fault>
  </operation>
  <operation name="getVolumeMapsByStorageArray">
    <soap:operation soapAction=""/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
    <fault name="APIException">
      <soap:fault name="APIException" use="literal"/>
    </fault>
    <fault name="APIBadRequestException">
      <soap:fault name="APIBadRequestException" use="literal"/>
    </fault>
  </operation>
  <operation name="getVolumeMasksByStorageArray">
    <soap:operation soapAction=""/>
    <input>

```

```
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getVolumes">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getVolumesByStorageArray">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
```

```

</operation>
<operation name="getVolumesSynchStatesByDRPath">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="getVolumesSynchStatesBySourceVolume">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="openSession">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>

```

```
</fault>
<fault name="APIBadRequestException">
  <soap:fault name="APIBadRequestException" use="literal"/>
</fault>
</operation>
<operation name="removeHostFromApplication">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="removeHostPolicies">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="removePolicies">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
```

```
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="setApplication">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="setHostsPolicy">
  <soap:operation soapAction=""/>
  <input>
    <soap:body use="literal"/>
  </input>
  <output>
    <soap:body use="literal"/>
  </output>
  <fault name="APIException">
    <soap:fault name="APIException" use="literal"/>
  </fault>
  <fault name="APIBadRequestException">
    <soap:fault name="APIBadRequestException" use="literal"/>
  </fault>
</operation>
<operation name="setPathsPolicy">
  <soap:operation soapAction=""/>
```

```
<input>
  <soap:body use="literal"/>
</input>
<output>
  <soap:body use="literal"/>
</output>
<fault name="APIException">
  <soap:fault name="APIException" use="literal"/>
</fault>
<fault name="APIBadRequestException">
  <soap:fault name="APIBadRequestException" use="literal"/>
</fault>
</operation>
</binding>
<service name="SANscreenAPIService">
  <port name="SANscreenAPIEndPointPort" binding="tns:SAN-
screenAPIEndPointBinding">
    <soap:address location="http://NIR:80/sanscreenapi"/>
  </port>
</service>
</definitions>
```

