rtslib

API Documentation

August 3, 2011

Contents

1 Package rtslib

Version: 1.99-20110803073816.644eece

Author: Jerome Martin <jxm@risingtidesystems.com>

1.1 Modules

1.2 Variables

Name	Description
__url__	**Value:** `'http://www.risingtidesystems.com'`
__description__	**Value:** `'API for RisingTide Systems generic SCSI target.'`

continued on next page

Name	Description
__package__	**Value:** `'rtslib'`

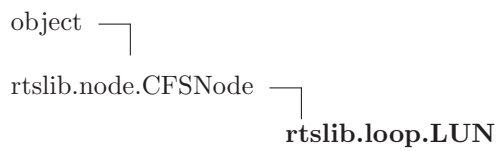
2 Module rtslib.loop

Implements the RTS SAS loopback classes.

2.1 Variables

Name	Description
__package__	**Value:** `'rtslib'`

2.2 Class LUN



This is an interface to RTS Target LUNs in configFS. A LUN is identified by its parent Nexus and LUN index.

2.2.1 Methods

__init__(*self*, *parent_nexus*, *lun*, *storage_object*=`None`, *alias*=`None`)

A LUN object can be instanciated in two ways:

- **Creation mode**: If *storage_object* is specified, the underlying configFS object will be created with that parameter. No LUN with the same *lun* index can pre-exist in the parent Nexus in that mode, or instanciation will fail.
- **Lookup mode**: If *storage_object* is not set, then the LUN will be bound to the existing configFS LUN object of the parent Nexus having the specified *lun* index. The underlying configFS object must already exist in that mode.

Parameters

`parent_nexus`:	The parent Nexus object. *(type=Nexus)*
`lun`:	The LUN index. *(type=0-255)*
`storage_object`:	The storage object to be exported as a LUN. *(type=StorageObject subclass)*
`alias`:	An optional parameter to manually specify the LUN alias. You probably do not need this. *(type=string)*

Return Value

A LUN object.

Overrides: object.__init__

__str__(*self*)

str(x)

Overrides: object.__str__ extit(inherited documentation)

delete(*self*)

If the underlying configFS object does not exists, this method does nothing. If the underlying configFS object exists, this method attempts to delete it.

Overrides: rtslib.node.CFSNode.delete

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

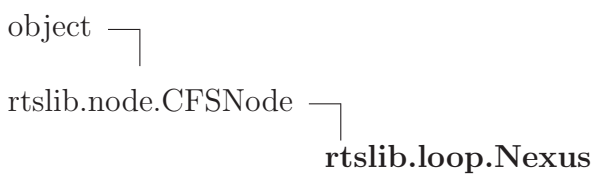
2.2.2 Properties

Name	Description
alua_metadata_path	Get the ALUA metadata directory path for the LUN.
parent_nexus	Get the parent Nexus object.
lun	Get the LUN index as an int.
storage_object	Get the storage object attached to the LUN.
alias	Get the LUN alias.
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

2.2.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

2.3 Class Nexus



This is a an interface to Target Portal Groups in configFS. A Nexus is identified by its parent Target object and its nexus Tag. To a Nexus object is attached a list of NetworkPortals.

2.3.1 Methods

__init__(*self*, *parent_target*, *tag*, *mode*=`'any'`)

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Parameters

`parent_target`: The parent Target object of the Nexus.

(type=Target)

`tag`: The Nexus Tag (TPGT).

(type=int > 0)

`mode`: An optionnal string containing the object creation mode:

- *'any'* means the configFS object will be either looked up or created.
- *'lookup'* means the object MUST already exist configFS.
- *'create'* means the object must NOT already exist in configFS.

(type=string)

Return Value

A Nexus object.

Overrides: object.__init__

__str__(*self*)

str(x)

Overrides: object.__str__ extit(inherited documentation)

delete(*self*)

Recursively deletes a Nexus object. This will delete all attached LUN, and then the Nexus itself.

Overrides: rtslib.node.CFSNode.delete

lun(*self*, *lun*, *storage_object*=`None`, *alias*=`None`)

Same as LUN() but without specifying the parent_nexus.

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(),

set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

2.3.2 Properties

Name	Description
alua_metadata_path	Get the ALUA metadata directory path for the Nexus.
tag	Get the Nexus Tag as an int.
initiator	Get the Nexus initiator address as a string.
parent_target	Get the parent Target object to which the Nexus is attached.
luns	Get the list of LUN objects currently attached to the Nexus.
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

2.3.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

2.4 Class Target

object
rtslib.node.CFSNode
rtslib.loop.Target

This is an interface to loopback SAS Targets in configFS. A Target is identified by its naa SAS address. To a Target is attached a list of Nexus objects.

2.4.1 Methods

__init__(*self*, *naa*=`None`, *mode*=`'any'`)

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Parameters

`naa:` The optionnal Target's address. If no address or an empty address is specified, one will be generated for you.

(type=string)

`mode:` An optionnal string containing the object creation mode:

- *'any'* means the configFS object will be either looked up or created.
- *'lookup'* means the object MUST already exist configFS.
- *'create'* means the object must NOT already exist in configFS.

(type=string)

Return Value

A Target object.

Overrides: object.__init__

__str__(*self*)

str(x)

Overrides: object.__str__ extit(inherited documentation)

delete(*self*)

Recursively deletes a Target object. This will delete all attached Nexus objects and then the Target itself.

Overrides: rtslib.node.CFSNode.delete

nexus(*self*, *tag*, *mode*=`'any'`)

Same as Nexus() but without the parent_target parameter.

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(),

__repr__(), __setattr__(), __sizeof__(), __subclasshook__()

2.4.2 Properties

Name	Description
naa	Get the naa of the Target object as a string.
nexuses	Get the list of Nexus objects currently attached to the Target.
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

2.4.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

3 Module rtslib.node

Implements the base CFSNode class and a few inherited variants.

3.1 Variables

Name	Description
__package__	**Value:** `'rtslib'`

3.2 Class CFSNode

object
└ **rtslib.node.CFSNode**

Known Subclasses: rtslib.loop.LUN, rtslib.loop.Nexus, rtslib.loop.Target, rtslib.root.RTSRoot, rtslib.target.FabricModule, rtslib.target.LUN, rtslib.target.MappedLUN, rtslib.target.NetworkPortal, rtslib.target.NodeACL, rtslib.target.TPG, rtslib.target.Target, rtslib.tcm.Backstore, rtslib.tcm.StorageObj

3.2.1 Methods

__init__(*self*)

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Overrides: object.__init__ extit(inherited documentation)

__nonzero__(*self*)

__str__(*self*)

str(x)

Overrides: object.__str__ extit(inherited documentation)

list_parameters(*self*, *writable*=`None`)

Parameters

`writable`: If None (default), returns all parameters, if True, returns read-write parameters, if False, returns just the read-only parameters.

(type=bool or None)

Return Value

The list of existing RFC-3720 parameter names.

list_attributes(*self*, *writable*=`None`)

Parameters

`writable`: If None (default), returns all attributes, if True, returns read-write attributes, if False, returns just the read-only attributes.

(type=bool or None)

Return Value

A list of existing attribute names as strings.

set_attribute(*self*, *attribute*, *value*)

Sets the value of a named attribute. The attribute must exist in configFS.

Parameters

`attribute`: The attribute's name. It is case-sensitive.

(type=string)

`value`: The attribute's value.

(type=string)

get_attribute(*self*, *attribute*)

Parameters

`attribute`: The attribute's name. It is case-sensitive.

Return Value

The named attribute's value, as a string.

set_parameter(*self*, *parameter*, *value*)

Sets the value of a named RFC-3720 parameter. The parameter must exist in configFS.

Parameters

`parameter:` The RFC-3720 parameter's name. It is case-sensitive.

(type=string)

`value:` The parameter's value.

(type=string)

get_parameter(*self*, *parameter*)

Parameters

`parameter:` The RFC-3720 parameter's name. It is case-sensitive.

(type=string)

Return Value

The named parameter value as a string.

delete(*self*)

If the underlying configFS object does not exists, this method does nothing. If the underlying configFS object exists, this method attempts to delete it.

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

3.2.2 Properties

Name	Description
path	Get the configFS object path.
exists	Is True as long as the underlying configFS object exists. If the underlying configFS objects gets deleted either by calling the delete() method, or by any other means, it will be False.
is_fresh	Is True if the underlying configFS object has been created when instanciating this particular object. Is False if this object instanciation just looked up the underlying configFS object.
Inherited from object __class__	

3.2.3 Class Variables

Name	Description
spec_dir	**Value:** `'/var/target/fabric'`
configfs_dir	**Value:** `'/sys/kernel/config/target'`
alua_metadata_dir	**Value:** `'/var/target/alua/iSCSI'`

4 Module rtslib.root

Implements the RTSRoot class.

4.1 Variables

Name	Description
__package__	**Value:** `'rtslib'`

4.2 Class RTSRoot

object
└ rtslib.node.CFSNode
  └ **rtslib.root.RTSRoot**

This is an interface to the root of the configFS object tree. Is allows one to start browsing Target and Backstore objects, as well as helper methods to return arbitrary objects from the configFS tree.

```
>>> import rtslib.root as root
>>> rtsroot = root.RTSRoot()
>>> rtsroot.path
'/sys/kernel/config/target'
>>> rtsroot.exists
True
>>> rtsroot.targets # doctest: +ELLIPSIS
[...]
>>> rtsroot.backstores # doctest: +ELLIPSIS
```

```
[...]
>>> rtsroot.tpgs # doctest: +ELLIPSIS
[...]
>>> rtsroot.storage_objects # doctest: +ELLIPSIS
[...]
>>> rtsroot.network_portals # doctest: +ELLIPSIS
[...]
```

4.2.1 Methods

__init__(*self*)

Instanciate an RTSRoot object. Basically checks for configfs setup and base kernel modules (tcm)

Overrides: object.__init__

__str__(*self*)

str(x)

Overrides: object.__str__ extit(inherited documentation)

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), delete(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

4.2.2 Properties

Name	Description
backstores	Get the list of Backstore objects.
targets	Get the list of Target objects.
tpgs	Get the list of all the existing TPG objects.
node_acls	Get the list of all the existing NodeACL objects.
network_portals	Get the list of all the existing Network Portal objects.
storage_objects	Get the list of all the existing Storage objects.
luns	Get the list of all existing LUN objects.

continued on next page

Name	Description
fabric_modules	Get the list of all FabricModule objects.
loaded_fabric_modules	Get the list of all loaded FabricModule objects.
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

4.2.3 Class Variables

Name	Description
target_core_mod	**Value:** `'target_core_mod'`
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

5 Module rtslib.target

Implements the RTS generic Target fabric classes.

5.1 Variables

Name	Description
__package__	**Value:** `'rtslib'`

5.2 Class FabricModule

object
└ rtslib.node.CFSNode
└ **rtslib.target.FabricModule**

This is an interface to RTS Target Fabric Modules. It can load/unload modules, provide information about them and handle the configfs housekeeping. It uses module configuration files in /var/target/fabric/*.spec. After instanciation, whether or not the fabric module is loaded and

5.2.1 Methods

__init__(*self*, *name*)

Instanciate a FabricModule object, according to the provided name.

Parameters

`name`: the name of the FabricModule object. It must match an existing target fabric module specfile (name.spec).

(type=str)

Overrides: object.__init__

has_feature(*self*, *feature*)

Whether or not this FabricModule has a certain feature.

load(*self*, *yield_steps*=`False`)

Attempt to load the target fabric kernel module as defined in the specfile.

Parameters

`yield_steps`: Whether or not to yield an (action, taken, desc) tuple at each step: action is either 'load_module' or 'create_cfs_group', 'taken' is a bool indicating whether the action was taken (if needed) or not, and desc is a text description of the step suitable for logging.

(type=bool)

Raises

`RTSLibError` For failure to load kernel module and/or create configfs group.

is_valid_wwn(*self*, *wwn*)

Checks whether or not the provided WWN is valid for this fabric module according to the spec file.

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), delete(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

5.2.2 Properties

Name	Description
discovery_userid	Set or get the initiator discovery userid.
discovery_password	Set or get the initiator discovery password.
discovery_mutual_userid	Set or get the mutual discovery userid.
discovery_mutual_password	Set or get the mutual discovery password.
discovery_enable_auth	Set or get the discovery enable_auth flag.
targets	Get the list of target objects.
version	Get the fabric module version string.
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

5.2.3 Class Variables

Name	Description
version_attributes	**Value:** `set(['lio_version', 'version'])`
discovery_auth_attributes	**Value:** `set(['discovery_auth'])`
target_names_excludes	**Value:** `set(['discovery_auth', 'lio_version', 'version'])`
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

5.3 Class LUN

object
└
rtslib.node.CFSNode
└
rtslib.target.LUN

This is an interface to RTS Target LUNs in configFS. A LUN is identified by its parent TPG and LUN index.

5.3.1 Methods

__init__(*self*, *parent_tpg*, *lun*, *storage_object*=`None`, *alias*=`None`)

A LUN object can be instanciated in two ways:

- **Creation mode**: If *storage_object* is specified, the underlying configFS object will be created with that parameter. No LUN with the same *lun* index can pre-exist in the parent TPG in that mode, or instanciation will fail.
- **Lookup mode**: If *storage_object* is not set, then the LUN will be bound to the existing configFS LUN object of the parent TPG having the specified *lun* index. The underlying configFS object must already exist in that mode.

Parameters

`parent_tpg`: The parent TPG object.
(type=TPG)

`lun`: The LUN index.
(type=0-255)

`storage_object`: The storage object to be exported as a LUN.
(type=StorageObject subclass)

`alias`: An optional parameter to manually specify the LUN alias. You probably do not need this.
(type=string)

Return Value

A LUN object.

Overrides: object.__init__

delete(*self*)

If the underlying configFS object does not exists, this method does nothing. If the underlying configFS object exists, this method attempts to delete it along with all MappedLUN objects referencing that LUN.

Overrides: rtslib.node.CFSNode.delete

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

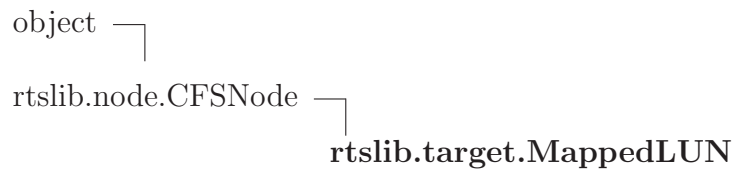
5.3.2 Properties

Name	Description
alua_metadata_path	Get the ALUA metadata directory path for the LUN.
parent_tpg	Get the parent TPG object.
lun	Get the LUN index as an int.
storage_object	Get the storage object attached to the LUN.
alias	Get the LUN alias.
mapped_luns	List all MappedLUN objects referencing this LUN.
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

5.3.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

5.4 Class MappedLUN



This is an interface to RTS Target Mapped LUNs. A MappedLUN is a mapping of a TPG LUN to a specific initiator node, and is part of a NodeACL. It allows the initiator to actually access the TPG LUN if ACLs are enabled for the TPG. The initial TPG LUN will then be seen by the initiator node as the MappedLUN.

5.4.1 Methods

__init__(*self*, *parent_nodeacl*, *mapped_lun*, *tpg_lun*=`None`, *write_protect*=`None`)

A MappedLUN object can be instanciated in two ways:

- **Creation mode**: If *tpg_lun* is specified, the underlying configFS object will be created with that parameter. No MappedLUN with the same *mapped_lun* index can pre-exist in the parent NodeACL in that mode, or instanciation will fail.
- **Lookup mode**: If *tpg_lun* is not set, then the MappedLUN will be bound to the existing configFS MappedLUN object of the parent NodeACL having the specified *mapped_lun* index. The underlying configFS object must already exist in that mode.

Parameters

`mapped_lun`: The mapped LUN index.

(type=int)

`tpg_lun`: The TPG LUN index to map, or directly a LUN object that belong to the same TPG as the parent NodeACL.

(type=int or LUN)

`write_protect`: The write-protect flag value, defaults to False (write-protection disabled).

(type=bool)

Overrides: object.__init__

delete(*self*)

Delete the MappedLUN.

Overrides: rtslib.node.CFSNode.delete

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

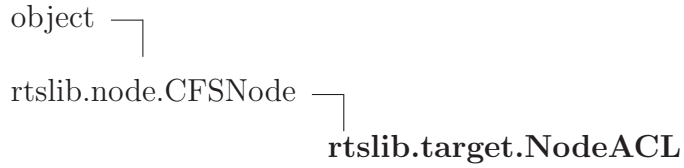
5.4.2 Properties

Name	Description
mapped_lun	Get the integer MappedLUN mapped_lun index.
parent_nodeacl	Get the parent NodeACL object.
write_protect	Get or set the boolean write protection.
tpg_lun	Get the TPG LUN object the MappedLUN is pointing at.
node_wwn	Get the wwn of the node for which the TPG LUN is mapped.
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

5.4.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

5.5 Class NodeACL



This is an interface to node ACLs in configFS. A NodeACL is identified by the initiator node wwn and parent TPG.

5.5.1 Methods

__init__(*self*, *parent_tpg*, *node_wwn*, *mode*='any')

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Parameters

`parent_tpg`: The parent TPG object.

(type=TPG)

`node_wwn`: The wwn of the initiator node for which the ACL is created.

(type=string)

`mode`: An optionnal string containing the object creation mode:

- *'any'* means the configFS object will be either looked up or created.
- *'lookup'* means the object MUST already exist configFS.
- *'create'* means the object must NOT already exist in configFS.

(type=string)

Return Value

A NodeACL object.

Overrides: object.__init__

has_feature(*self*, *feature*)

Whether or not this NodeACL has a certain feature.

delete(*self*)

Delete the NodeACL, including all MappedLUN objects. If the underlying configFS object does not exist, this method does nothing.

Overrides: rtslib.node.CFSNode.delete

mapped_lun(*self*, *mapped_lun*, *tpg_lun*=None, *write_protect*=None)

Same as MappedLUN() but without the parent_nodeacl parameter.

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

5.5.2 Properties

Name	Description
chap_userid	Set or get the initiator CHAP auth userid.
chap_password	Set or get the initiator CHAP auth password.
chap_mutual_userid	Set or get the mutual CHAP auth userid.
chap_mutual_password	Set or get the mutual CHAP password.
tcq_depth	Set or get the TCQ depth for the initiator sessions matching this NodeACL.
parent_tpg	Get the parent TPG object.
node_wwn	Get the node wwn.
authenticate_target	Get the boolean authenticate target flag.
mapped_luns	Get the list of all MappedLUN objects in this NodeACL.
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

5.5.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

5.6 Class NetworkPortal

This is an interface to NetworkPortals in configFS. A NetworkPortal is identified by its IP and port, but here we also require the parent TPG, so instance objects represent both the NetworkPortal and its association to a TPG. This is necessary to get path information in order to create the portal in the proper configFS hierarchy.

5.6.1 Methods

__init__(*self*, *parent_tpg*, *ip_address*, *port*, *mode='any'*)

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Parameters

`parent_tpg`: The parent TPG object.

(type=TPG)

`ip_address`: The ipv4 IP address of the NetworkPortal.

(type=string)

`port`: The NetworkPortal TCP/IP port.

(type=int)

`mode`: An optionnal string containing the object creation mode:

- *'any'* means the configFS object will be either looked up or created.
- *'lookup'* means the object MUST already exist configFS.
- *'create'* means the object must NOT already exist in configFS.

(type=string)

Return Value

A NetworkPortal object.

Overrides: object.__init__

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), delete(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

5.6.2 Properties

Name	Description
parent_tpg	Get the parent TPG object.
port	Get the NetworkPortal's TCP port as an int.

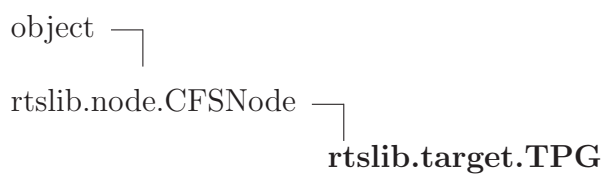
continued on next page

Name	Description
ip_address	Get the NetworkPortal's IP address as a string.
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

5.6.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

5.7 Class TPG



This is a an interface to Target Portal Groups in configFS. A TPG is identified by its parent Target object and its TPG Tag. To a TPG object is attached a list of NetworkPortals. Targets without the 'tpgts' feature cannot have more than a single TPG, so attempts to create more will raise an exception.

5.7.1 Methods

__init__(*self*, *parent_target*, *tag*, *mode*='any')

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Parameters

`parent_target`: The parent Target object of the TPG.

(type=Target)

`tag`: The TPG Tag (TPGT).

(type=int > 0)

`mode`: An optionnal string containing the object creation mode:

- *'any'* means the configFS object will be either looked up or created.
- *'lookup'* means the object MUST already exist configFS.
- *'create'* means the object must NOT already exist in configFS.

(type=string)

Return Value

A TPG object.

Overrides: object.__init__

has_feature(*self*, *feature*)

Whether or not this TPG has a certain feature.

delete(*self*)

Recursively deletes a TPG object. This will delete all attached LUN, NetworkPortal and Node ACL objects and then the TPG itself. Before starting the actual deletion process, all sessions will be disconnected.

Overrides: rtslib.node.CFSNode.delete

node_acl(*self*, *node_wwn*, *mode*='any')

Same as NodeACL() but without specifying the parent_tpg.

network_portal(*self*, *ip_address*, *port*, *mode*='any')

Same as NetworkPortal() but without specifying the parent_tpg.

lun(*self*, *lun*, *storage_object*=None, *alias*=None)

Same as LUN() but without specifying the parent_tpg.

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

5.7.2 Properties

Name	Description
alua_metadata_path	Get the ALUA metadata directory path for the TPG.
tag	Get the TPG Tag as an int.
parent_target	Get the parent Target object to which the TPG is attached.
enable	Get or set a boolean value representing the enable status of the TPG. True means the TPG is enabled, False means it is disabled.
network_portals	Get the list of NetworkPortal objects currently attached to the TPG.
node_acls	Get the list of NodeACL objects currently attached to the TPG.
luns	Get the list of LUN objects currently attached to the TPG.
nexus	Get or set (once) the TPG's Nexus is used.
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

5.7.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

5.8 Class Target

object
rtslib.node.CFSNode
rtslib.target.Target

This is an interface to Targets in configFS. A Target is identified by its wwn. To a Target is attached a list of TPG objects.

5.8.1 Methods

__init__(*self*, *fabric_module*, *wwn*=`None`, *mode*=`'any'`)

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Parameters

`fabric_module`: The target's fabric module.

(type=FabricModule)

`wwn`: The optionnal Target's wwn. If no wwn or an empty wwn is specified, one will be generated for you.

(type=string)

`mode`: An optionnal string containing the object creation mode:

- *'any'* means the configFS object will be either looked up or created.
- *'lookup'* means the object MUST already exist configFS.
- *'create'* means the object must NOT already exist in configFS.

(type=string)

Return Value

A Target object.

Overrides: object.__init__

has_feature(*self*, *feature*)

Whether or not this Target has a certain feature.

delete(*self*)

Recursively deletes a Target object. This will delete all attached TPG objects and then the Target itself.

Overrides: rtslib.node.CFSNode.delete

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

5.8.2 Properties

Name	Description
tpgs	Get the list of TPG for the Target.
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

5.8.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

6 Module rtslib.tcm

Implements the RTS Target backstore and storage object classes.

6.1 Variables

Name	Description
__package__	**Value:** `'rtslib'`

6.2 Class Backstore

object
└
rtslib.node.CFSNode
└
rtslib.tcm.Backstore

Known Subclasses: rtslib.tcm.FileIOBackstore, rtslib.tcm.IBlockBackstore, rtslib.tcm.PSCSIBackstore, rtslib.tcm.RDDRBackstore, rtslib.tcm.RDMCPBackstore

6.2.1 Methods

__init__(*self*, *plugin*, *storage_class*, *index*, *mode*)

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Overrides: object.__init__ extit(inherited documentation)

delete(*self*)

Recursively deletes a Backstore object. This will delete all attached StorageObject objects, and then the Backstore itself. The underlying file and block storages will not be touched, but all ramdisk data will be lost.

Overrides: rtslib.node.CFSNode.delete

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

6.2.2 Properties

Name	Description
index	Get the backstore index as an int.
storage_objects	Get the list of StorageObjects attached to the backstore.
version	Get the Backstore plugin version string.
plugin	Get the Backstore plugin name.
name	Get the backstore name.
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

6.2.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

6.3 Class PSCSIBackstore

object
rtslib.node.CFSNode
rtslib.tcm.Backstore
rtslib.tcm.PSCSIBackstore

This is an interface to pscsi backstore plugin objects in configFS. A PSCSIBackstore object is identified by its backstore index.

6.3.1 Methods

__init__(*self*, *index*, *mode*='any', *legacy*=False)

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Parameters

`index`: The backstore index matching a physical SCSI HBA.

(type=int)

`mode`: An optionnal string containing the object creation mode:

- *'any'* the configFS object will be either lookuped or created.
- *'lookup'* the object MUST already exist configFS.
- *'create'* the object must NOT already exist in configFS.

(type=string)

`legacy`: Enable legacy physcal HBA mode. If True, you must specify it also in lookup mode for StorageObjects to be notified. You've been warned !

Return Value

A PSCSIBackstore object.

Overrides: object.__init__

storage_object(*self*, *name*, *dev*=None)

Same as PSCSIStorageObject() without specifying the backstore

Inherited from rtslib.tcm.Backstore(Section 6.2)

delete()

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

6.3.2 Properties

Name	Description
legacy_mode	Get the legacy mode flag. If True, the Vitualbackstore index must match the StorageObjects real HBAs.
Inherited from rtslib.tcm.Backstore (Section 6.2) index, name, plugin, storage_objects, version	
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

6.3.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

6.4 Class RDDRBackstore

object
rtslib.node.CFSNode
rtslib.tcm.Backstore
rtslib.tcm.RDDRBackstore

This is an interface to rd_dr backstore plugin objects in configFS. A RDDRBackstore object is identified by its backstore index.

6.4.1 Methods

__init__(*self*, *index*, *mode*='any')

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Parameters

`index`: The backstore index.

(type=int)

`mode`: An optionnal string containing the object creation mode:

- *'any'* the configFS object will be either lookupd or created.
- *'lookup'* the object MUST already exist configFS.
- *'create'* the object must NOT already exist in configFS.

(type=string)

Return Value

A RDDRBackstore object.

Overrides: object.__init__

storage_object(*self*, *name*, *size*=`None`, *gen_wwn*=`True`)

Same as RDDRStorageObject() without specifying the backstore

Inherited from rtslib.tcm.Backstore(Section 6.2)

delete()

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

6.4.2 Properties

Name	Description
Inherited from rtslib.tcm.Backstore (Section 6.2)	
index, name, plugin, storage_objects, version	
Inherited from rtslib.node.CFSNode (Section 3.2)	
exists, is_fresh, path	

continued on next page

Name	Description
Inherited from object	
__class__	

6.4.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2)	
alua_metadata_dir, configfs_dir, spec_dir	

6.5 Class RDMCPBackstore

This is an interface to rd_mcp backstore plugin objects in configFS. A RDMCPBackstore object is identified by its backstore index.

6.5.1 Methods

__init__(*self*, *index*, *mode*='any')

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Parameters

`index`: The backstore index.

(type=int)

`mode`: An optionnal string containing the object creation mode:

- *'any'* the configFS object will be either lookupd or created.
- *'lookup'* the object MUST already exist configFS.
- *'create'* the object must NOT already exist in configFS.

(type=string)

Return Value

A RDMCPBackstore object.

Overrides: object.__init__

storage_object(*self*, *name*, *size*=`None`, *gen_wwn*=`True`)

Same as RDMCPStorageObject() without specifying the backstore

Inherited from rtslib.tcm.Backstore(Section 6.2)

delete()

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

6.5.2 Properties

Name	Description
Inherited from rtslib.tcm.Backstore (Section 6.2) index, name, plugin, storage_objects, version	
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	

continued on next page

Name	Description
Inherited from object	
__class__	

6.5.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2)	
alua_metadata_dir, configfs_dir, spec_dir	

6.6 Class FileIOBackstore

This is an interface to fileio backstore plugin objects in configFS. A FileIOBackstore object is identified by its backstore index.

6.6.1 Methods

__init__(*self*, *index*, *mode*='any')

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Parameters

`index`: The backstore index.

(type=int)

`mode`: An optionnal string containing the object creation mode:

- *'any'* the configFS object will be either lookuped or created.
- *'lookup'* the object MUST already exist configFS.
- *'create'* the object must NOT already exist in configFS.

(type=string)

Return Value

A FileIOBackstore object.

Overrides: object.__init__

storage_object(*self*, *name*, *dev*=`None`, *size*=`None`, *gen_wwn*=`True`, *buffered_mode*=`False`)

Same as FileIOStorageObject() without specifying the backstore

Inherited from rtslib.tcm.Backstore(Section 6.2)

delete()

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

6.6.2 Properties

Name	Description
Inherited from rtslib.tcm.Backstore (Section 6.2)	
index, name, plugin, storage_objects, version	
Inherited from rtslib.node.CFSNode (Section 3.2)	

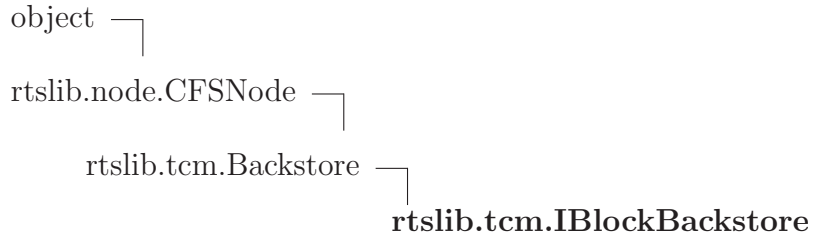
continued on next page

Name	Description
exists, is_fresh, path	
Inherited from object __class__	

6.6.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

6.7 Class IBlockBackstore



This is an interface to iblock backstore plugin objects in configFS. An IBlockBackstore object is identified by its backstore index.

6.7.1 Methods

__init__(*self*, *index*, *mode*='any')

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Parameters

`index`: The backstore index.

(type=int)

`mode`: An optionnal string containing the object creation mode:

- *'any'* the configFS object will be either lookupd or created.
- *'lookup'* the object MUST already exist configFS.
- *'create'* the object must NOT already exist in configFS.

(type=string)

Return Value

An IBlockBackstore object.

Overrides: object.__init__

storage_object(*self*, *name*, *dev*=`None`, *gen_wwn*=`True`)

Same as IBlockStorageObject() without specifying the backstore

Inherited from rtslib.tcm.Backstore(Section 6.2)

delete()

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

6.7.2 Properties

Name	Description
Inherited from rtslib.tcm.Backstore (Section 6.2)	
index, name, plugin, storage_objects, version	
Inherited from rtslib.node.CFSNode (Section 3.2)	
exists, is_fresh, path	

continued on next page

Name	Description
Inherited from object __class__	

6.7.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

6.8 Class StorageObject

object
└
rtslib.node.CFSNode
└
rtslib.tcm.StorageObject

Known Subclasses: rtslib.tcm.FileIOStorageObject, rtslib.tcm.IBlockStorageObject, rtslib.tcm.PSCSIStorageObject, rtslib.tcm.RDDRStorageObject, rtslib.tcm.RDMCPStorageObject

This is an interface to storage objects in configFS. A StorageObject is identified by its backstore and its name.

6.8.1 Methods

__init__(*self*, *backstore*, *backstore_class*, *name*, *mode*)

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Overrides: object.__init__ extit(inherited documentation)

delete(*self*)

Recursively deletes a StorageObject object. This will delete all attached LUNs currently using the StorageObject object, and then the StorageObject itself. The underlying file and block storages will not be touched, but all ramdisk data will be lost.

Overrides: rtslib.node.CFSNode.delete

is_configured(*self*)

Return Value

True if the StorageObject is configured, else returns False

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

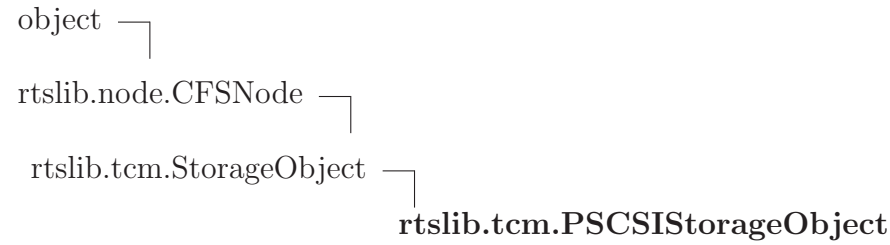
6.8.2 Properties

Name	Description
backstore	Get the backstore object.
name	Get the StorageObject name as a string.
udev_path	Get the StorageObject udev_path as a string.
wwn	Get or set the StorageObject T10 WWN Serial as a string.
status	Get the storage object status, depending on wether or not itis used by any LUN
attached_luns	Get the list of all LUN objects attached.
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

6.8.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

6.9 Class PSCSIStorageObject



An interface to configFS storage objects for pscsi backstore.

6.9.1 Methods

__init__(*self*, *backstore*, *name*, *dev*=`None`)

A PSCSIStorageObject can be instanciated in two ways:

- **Creation mode**: If *dev* is specified, the underlying configFS object will be created with that parameter. No PSCSIStorageObject with the same *name* can pre-exist in the parent PSCSIBackstore in that mode, or instanciation will fail.
- **Lookup mode**: If *dev* is not set, then the PSCSIStorageObject will be bound to the existing configFS object in the parent PSCSIBackstore having the specified *name*. The underlying configFS object must already exist in that mode, or instanciation will fail.

Parameters

`backstore`: The parent backstore of the PSCSIStorageObject.

(type=PSCSIBackstore)

`name`: The name of the PSCSIStorageObject.

(type=string)

`dev`: You have two choices:

- Use the SCSI id of the device: *dev="H:C:T:L"*. If the parent backstore is in legacy mode, you must use *dev="C:T:L"* instead, as the backstore index of the SCSI dev device would then be constrained by the parent backstore index.
- Use the path to the SCSI device: *dev="/path/to/dev"*. Note that if the parent Backstore is in legacy mode, the device must have the same backstore index as the parent backstore.

(type=string)

Return Value

A PSCSIStorageObject object.

Overrides: object.__init__

Inherited from rtslib.tcm.StorageObject(Section 6.8)

delete(), is_configured()

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

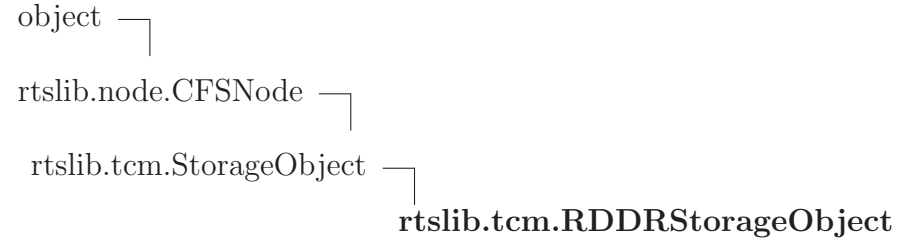
6.9.2 Properties

Name	Description
wwn	Get the StorageObject T10 WWN Unit Serial as a string. You cannot set it for pscsi-backed StorageObjects.
model	Get the SCSI device model string
vendor	Get the SCSI device vendor string
revision	Get the SCSI device revision string
host_id	Get the SCSI device host id
channel_id	Get the SCSI device channel id
target_id	Get the SCSI device target id
lun	Get the SCSI device LUN
Inherited from rtslib.tcm.StorageObject (Section 6.8) attached_luns, backstore, name, status, udev_path	
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

6.9.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

6.10 Class RDDRStorageObject



An interface to configFS storage objects for rd_dr backstore.

6.10.1 Methods

__init__(*self*, *backstore*, *name*, *size*=`None`, *gen_wwn*=`True`)

A RDDRStorageObject can be instanciated in two ways:

- **Creation mode**: If *size* is specified, the underlying configFS object will be created with that parameter. No RDDRStorageObject with the same *name* can pre-exist in the parent RDDRBackstore in that mode, or instanciation will fail.
- **Lookup mode**: If *size* is not set, then the RDDRStorageObject will be bound to the existing configFS object in the parent RDDRBackstore having the specified *name*. The underlying configFS object must already exist in that mode, or instanciation will fail.

Parameters

`backstore`: The parent backstore of the RDDRStorageObject.

(type=RDDRBackstore)

`name`: The name of the RDDRStorageObject.

(type=string)

`size`: The size of the ramdrive to create:

- If size is an int, it represents a number of bytes
- If size is a string, the following units can be used :
  - *B* or no unit present for bytes
  - *k*, *K*, *kB*, *KB* for kB (kilobytes)
  - *m*, *M*, *mB*, *MB* for MB (megabytes)
  - *g*, *G*, *gB*, *GB* for GB (gigabytes)
  - *t*, *T*, *tB*, *TB* for TB (terabytes) Example: size="1MB" for a one megabytes storage object.
  - Note that the size will be rounded to the closest 4096 Bytes RAM pages count. For instance, a size of 100000 Bytes will be rounded to 24 pages, really 98304 Bytes.
  - The base value for kilo is 1024, aka 1kB = 1024B. Strictly speaking, we use kiB, MiB, etc.

(type=string or int)

`gen_wwn`: Should we generate a T10 WWN Unit Serial ?

(type=bool)

Return Value

A RDDRStorageObject object.

Overrides: object.__init__

Inherited from rtslib.tcm.StorageObject(Section 6.8)

delete(), is_configured()

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

6.10.2 Properties

Name	Description
page_size	Get the ramdisk page size.
pages	Get the ramdisk number of pages.
size	Get the ramdisk size in bytes.
Inherited from rtslib.tcm.StorageObject (Section 6.8) attached_luns, backstore, name, status, udev_path, wwn	
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

6.10.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

6.11 Class RDMCPStorageObject

object
└ rtslib.node.CFSNode
└ rtslib.tcm.StorageObject
└ **rtslib.tcm.RDMCPStorageObject**

An interface to configFS storage objects for rd_mcp backstore.

6.11.1 Methods

__init__(*self*, *backstore*, *name*, *size*=`None`, *gen_wwn*=`True`)

A RDMCPStorageObject can be instanciated in two ways:

- **Creation mode**: If *size* is specified, the underlying configFS object will be created with that parameter. No RDMCPStorageObject with the same *name* can pre-exist in the parent RDMCPBackstore in that mode, or instanciation will fail.
- **Lookup mode**: If *size* is not set, then the RDMCPStorageObject will be bound to the existing configFS object in the parent RDMCPBackstore having the specified *name*. The underlying configFS object must already exist in that mode, or instanciation will fail.

Parameters

`backstore`: The parent backstore of the RDMCPStorageObject.

(type=RDMCPBackstore)

`name`: The name of the RDMCPStorageObject.

(type=string)

`size`: The size of the ramdrive to create:

- If size is an int, it represents a number of bytes
- If size is a string, the following units can be used :
  - **B** or no unit present for bytes
  - **k**, **K**, **kB**, **KB** for kB (kilobytes)
  - **m**, **M**, **mB**, **MB** for MB (megabytes)
  - **g**, **G**, **gB**, **GB** for GB (gigabytes)
  - **t**, **T**, **tB**, **TB** for TB (terabytes) Example: size="1MB" for a one megabytes storage object.
  - Note that the size will be rounded to the closest 4096 Bytes RAM pages count. For instance, a size of 100000 Bytes will be rounded to 24 pages, really 98304 Bytes.
  - The base value for kilo is 1024, aka 1kB = 1024B. Strictly speaking, we use kiB, MiB, etc.

(type=string or int)

`gen_wwn`: Should we generate a T10 WWN Unit Serial ?

(type=bool)

Return Value

A RDMCPStorageObject object.

Overrides: object.__init__

Inherited from rtslib.tcm.StorageObject(Section 6.8)

delete(), is_configured()

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

6.11.2 Properties

Name	Description
page_size	Get the ramdisk page size.
pages	Get the ramdisk number of pages.
size	Get the ramdisk size in bytes.
Inherited from rtslib.tcm.StorageObject (Section 6.8) attached_luns, backstore, name, status, udev_path, wwn	
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

6.11.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

6.12 Class FileIOStorageObject

object
└ rtslib.node.CFSNode
 └ rtslib.tcm.StorageObject
  └ **rtslib.tcm.FileIOStorageObject**

An interface to configFS storage objects for fileio backstore.

6.12.1 Methods

__init__(*self*, *backstore*, *name*, *dev*=`None`, *size*=`None`, *gen_wwn*=`True`, *buffered_mode*=`False`)

A FileIOStorageObject can be instanciated in two ways:

- **Creation mode**: If *dev* and *size* are specified, the underlying configFS object will be created with those parameters. No FileIOStorageObject with the same *name* can pre-exist in the parent FileIOBackstore in that mode, or instanciation will fail.
- **Lookup mode**: If *dev* and *size* are not set, then the FileIOStorageObject will be bound to the existing configFS object in the parent FileIOBackstore having the specified *name*. The underlying configFS object must already exist in that mode, or instanciation will fail.

Parameters

`backstore`: The parent backstore of the FileIOStorageObject.

(type=FileIOBackstore)

`name`: The name of the FileIOStorageObject.

(type=string)

`dev`: The path to the backend file or block device to be used.

- Examples: *dev="/dev/sda"*, *dev="/tmp/myfile"*
- The only block device type that is accepted *TYPE_DISK*, or partitions of a *TYPE_DISK* device. For other device types, use pscsi.

(type=string)

`size`: The maximum size to allocate for the file. Not used for block devices.

- If size is an int, it represents a number of bytes
- If size is a string, the following units can be used :
  - **B** or no unit present for bytes
  - **k**, **K**, **kB**, **KB** for kB (kilobytes)
  - **m**, **M**, **mB**, **MB** for MB (megabytes)
  - **g**, **G**, **gB**, **GB** for GB (gigabytes)
  - **t**, **T**, **tB**, **TB** for TB (terabytes) Example: size="1MB" for a one megabytes storage object.
  - The base value for kilo is 1024, aka 1kB = 1024B. Strictly speaking, we use kiB, MiB, etc.

(type=string or int)

`gen_wwn`: Should we generate a T10 WWN Unit Serial ?

Inherited from rtslib.tcm.StorageObject(Section 6.8)

delete(), is_configured()

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __subclasshook__()

6.12.2 Properties

Name	Description
mode	Get the current FileIOStorage mode, buffered or synchronous
size	Get the current FileIOStorage size in bytes
Inherited from rtslib.tcm.StorageObject (Section 6.8) attached_luns, backstore, name, status, udev_path, wwn	
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

6.12.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

6.13 Class IBlockStorageObject

object
└
rtslib.node.CFSNode
└
rtslib.tcm.StorageObject
└
rtslib.tcm.IBlockStorageObject

An interface to configFS storage objects for iblock backstore.

6.13.1 Methods

__init__(*self*, *backstore*, *name*, *dev*=`None`, *gen_wwn*=`True`)

A BlockIOStorageObject can be instanciated in two ways:

- **Creation mode**: If *dev* is specified, the underlying configFS object will be created with that parameter. No BlockIOStorageObject with the same *name* can pre-exist in the parent BlockIOBackstore in that mode.
- **Lookup mode**: If *dev* is not set, then the BlockIOStorageObject will be bound to the existing configFS object in the parent BlockIOBackstore having the specified *name*. The underlying configFS object must already exist in that mode, or instanciation will fail.

Parameters

`backstore`: The parent backstore of the BlockIOStorageObject.

(type=BlockIOBackstore)

`name`: The name of the BlockIOStorageObject.

(type=string)

`dev`: The path to the backend block device to be used.

- Example: *dev="/dev/sda"*.
- The only device type that is accepted *TYPE_DISK*. For other device types, use pscsi.

(type=string)

`gen_wwn`: Should we generate a T10 WWN Unit Serial when creating the object ?

(type=bool)

Return Value

A BlockIOStorageObject object.

Overrides: object.__init__

Inherited from rtslib.tcm.StorageObject(Section 6.8)

delete(), is_configured()

Inherited from rtslib.node.CFSNode(Section 3.2)

__nonzero__(), __str__(), get_attribute(), get_parameter(), list_attributes(), list_parameters(), set_attribute(), set_parameter()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(),

__repr__(), __setattr__(), __sizeof__(), __subclasshook__()

6.13.2 Properties

Name	Description
major	Get the block device major number
minor	Get the block device minor number
Inherited from rtslib.tcm.StorageObject (Section 6.8) attached_luns, backstore, name, status, udev_path, wwn	
Inherited from rtslib.node.CFSNode (Section 3.2) exists, is_fresh, path	
Inherited from object __class__	

6.13.3 Class Variables

Name	Description
Inherited from rtslib.node.CFSNode (Section 3.2) alua_metadata_dir, configfs_dir, spec_dir	

7 Module rtslib.utils

Provides various utility functions.

7.1 Functions

flatten_nested_list(*nested_list*)

Function to flatten a nested list.

```
>>> import rtslib.utils as utils
>>> utils.flatten_nested_list([[1,2,3,[4,5,6]],[7,8],[[[9,10]],[11,]]])
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11]
```

Parameters

`nested_list`: A nested list (list of lists of lists etc.)

(type=list)

Return Value

A list with only non-list elements

gen_list_item(*nested_list*)

The generator for flatten_nested_list(). It returns one by one items that are not a list, and recurses when he finds an item that is a list.

fwrite(*path*, *string*)

This function writes a string to a file, and takes care of opening it and closing it. If the file does not exists, it will be created.

```
>>> from rtslib.utils import *
>>> fwrite("/tmp/test", "hello")
>>> fread("/tmp/test")
'hello'
```

Parameters

`path`: The file to write to.

(type=string)

`string`: The string to write to the file.

(type=string)

fread(*path*)

This function reads the contents of a file. It takes care of opening and closing it.

```
>>> from rtslib.utils import *
>>> fwrite("/tmp/test", "hello")
>>> fread("/tmp/test")
'hello'
>>> fread("/tmp/notexistingfile") # doctest: +ELLIPSIS
Traceback (most recent call last):
    ...
IOError: [Errno 2] No such file or directory: '/tmp/notexistingfile'
```

Parameters

`path`: The path to the file to read from.

(type=string)

Return Value

A string containing the file's contents.

is_dev_in_use(*path*)

This function will check if the device or file referenced by path is already mounted or used as a storage object backend. It works by trying to open the path with O_EXCL flag, which will fail if someone else already did. Note that the file is closed before the function returns, so this does not guaranteed the device will still be available after the check.

Parameters

`path:` path to the file of device to check

(type=string)

Return Value

A boolean, True is we cannot get exclusive descriptor on the path, False if we can.

is_disk_partition(*path*)

Try to find out if path is a partition of a TYPE_DISK device. Handles both /dev/sdaX and /dev/disk/by-*/*-part? schemes.

get_disk_size(*path*)

This function returns the size in bytes of a disk-type block device, or None if path does not point to a disk- type device.

get_block_numbers(*path*)

This function returns a (major,minor) tuple for the block device found at path, or (None, None) if path is not a block device.

get_block_type(*path*)

This function returns a block device's type. Example: 0 is TYPE_DISK If no match is found, None is returned.

```
>>> from rtslib.utils import *
>>> get_block_type("/dev/sda")
0
>>> get_block_type("/dev/sr0")
5
>>> get_block_type("/dev/scd0")
5
>>> get_block_type("/dev/nodevicehere") is None
True
```

Parameters

`path`: path to the block device

(type=string)

Return Value

An int for the block device type, or None if not a block device.

list_scsi_hbas()

This function returns the list of HBA indexes for existing SCSI HBAs.

convert_scsi_path_to_hctl(*path*)

This function returns the SCSI ID in H:C:T:L form for the block device being mapped to the udev path specified. If no match is found, None is returned.

```
>>> import rtslib.utils as utils
>>> utils.convert_scsi_path_to_hctl('/dev/scd0')
(2, 0, 0, 0)
>>> utils.convert_scsi_path_to_hctl('/dev/sr0')
(2, 0, 0, 0)
>>> utils.convert_scsi_path_to_hctl('/dev/sda')
(3, 0, 0, 0)
>>> utils.convert_scsi_path_to_hctl('/dev/sda1')
>>> utils.convert_scsi_path_to_hctl('/dev/sdb')
(3, 0, 1, 0)
>>> utils.convert_scsi_path_to_hctl('/dev/sdc')
(3, 0, 2, 0)
```

Parameters

`path`: The udev path to the SCSI block device.

(type=string)

Return Value

An (host, controller, target, lun) tuple of integer values representing the SCSI ID of the device, or None if no match is found.

convert_scsi_hctl_to_path(*host*, *controller*, *target*, *lun*)

This function returns a udev path pointing to the block device being mapped to the SCSI device that has the provided H:C:T:L.

```
>>> import rtslib.utils as utils
>>> utils.convert_scsi_hctl_to_path(0,0,0,0)
''
>>> utils.convert_scsi_hctl_to_path(2,0,0,0) # doctest: +ELLIPSIS
'/dev/s...0'
>>> utils.convert_scsi_hctl_to_path(3,0,2,0)
'/dev/sdc'
```

Parameters

`host`: The SCSI host id.
(type=int)

`controller`: The SCSI controller id.
(type=int)

`target`: The SCSI target id.
(type=int)

`lun`: The SCSI Logical Unit Number.
(type=int)

Return Value

A string for the canonical path to the device, or empty string.

convert_human_to_bytes(*hsize*, *kilo*=1024)

This function converts human-readable amounts of bytes to bytes. It understands the following units :

- *B* or no unit present for Bytes
- *k*, *K*, *kB*, *KB* for kB (kilobytes)
- *m*, *M*, *mB*, *MB* for MB (megabytes)
- *g*, *G*, *gB*, *GB* for GB (gigabytes)
- *t*, *T*, *tB*, *TB* for TB (terabytes)

Note: The definition of *kilo* defaults to 1kB = 1024Bytes. Strictly speaking, those should not be called *kB* but *kiB*. You can override that with the optional kilo parameter.

Example:

```
>>> import rtslib.utils as utils
>>> utils.convert_human_to_bytes("1k")
1024
>>> utils.convert_human_to_bytes("1k", 1000)
1000
>>> utils.convert_human_to_bytes("1MB")
1048576
>>> utils.convert_human_to_bytes("12kB")
12288
```

Parameters

`hsize`: The human-readable version of the Bytes amount to convert

(type=string or int)

`kilo`: Optionnal base for the kilo prefix

(type=int)

Return Value

An int representing the human-readable string converted to bytes

generate_wwn(*wwn_type*)

Generates a random WWN of the specified type:

- unit_serial: T10 WWN Unit Serial.
- iqn: iSCSI IQN
- naa: SAS NAA address

Parameters

`wwn_type`: The WWN address type.

(type=str)

Return Value

A string containing the WWN.

is_valid_wwn(*wwn_type*, *wwn*, *wwn_list*=`None`)

Returns True if the wwn is a valid wwn of type wwn_type.

Parameters

`wwn_type`: The WWN address type.

(type=str)

`wwn`: The WWN address to check.

(type=str)

`wwn_list`: An optional list of wwns to check the wwn parameter from.

(type=list of str)

Return Value

bool.

list_available_kernel_modules()

List all loadable kernel modules as registered by depmod

list_loaded_kernel_modules()

List all currently loaded kernel modules

modprobe(*module*)

Load the specified kernel module if needed.

Parameters

`module`: The name of the kernel module to be loaded.

(type=str)

Return Value

Whether of not we had to load the module.

exec_argv(*argv*, *strip*=`True`, *shell*=`False`)

Executes a command line given as an argv table and either:

- raise an exception if return != 0
- return the output

If strip is True, then output lines will be stripped. If shell is True, the argv must be a string that will be evaluated by the shell, instead of the argv list.

list_eth_names(*max_eth*=`1024`)

List the max_eth first local ethernet interfaces names from SIOCGIFCONF struct.

list_eth_ips(*ifnames*=`None`)

List the IPv4 and IPv6 non-loopback, non link-local addresses of a list of ethernet interfaces from the SIOCGIFADDR struct. If ifname is omitted, list all IPs of all ifaces excepted for lo.

is_ipv4_address(*addr*)

is_ipv6_address(*addr*)

get_main_ip()

Try to guess the local machine non-loopback IP. If available, local hostname resolution is used (if non-loopback), else try to find an other non-loopback IP on configured NICs. If no usable IP address is found, returns None.

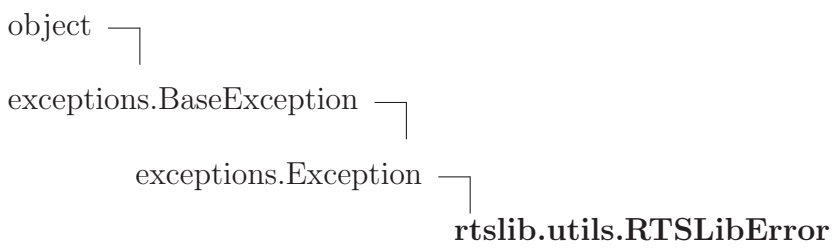
7.2 Variables

Name	Description
__package__	**Value:** `'rtslib'`

continued on next page

Name	Description

7.3 Class RTSLibError



Known Subclasses: rtslib.utils.RTSLibBrokenLink, rtslib.utils.RTSLibNotInCFS

Generic rtslib error.

7.3.1 Methods

Inherited from exceptions.Exception

__init__(), __new__()

Inherited from exceptions.BaseException

__delattr__(), __getattribute__(), __getitem__(), __getslice__(), __reduce__(), __repr__(), __setattr__(), __setstate__(), __str__(), __unicode__()

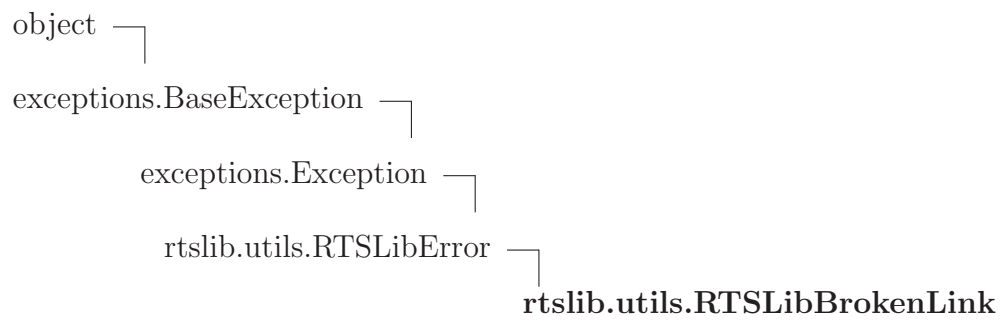
Inherited from object

__format__(), __hash__(), __reduce_ex__(), __sizeof__(), __subclasshook__()

7.3.2 Properties

Name	Description
Inherited from exceptions.BaseException args, message	
Inherited from object __class__	

7.4 Class RTSLibBrokenLink



Broken link in configfs, i.e. missing LUN storage object.

7.4.1 Methods

Inherited from exceptions.Exception

__init__(), __new__()

Inherited from exceptions.BaseException

__delattr__(), __getattribute__(), __getitem__(), __getslice__(), __reduce__(), __repr__(), __setattr__(), __setstate__(), __str__(), __unicode__()

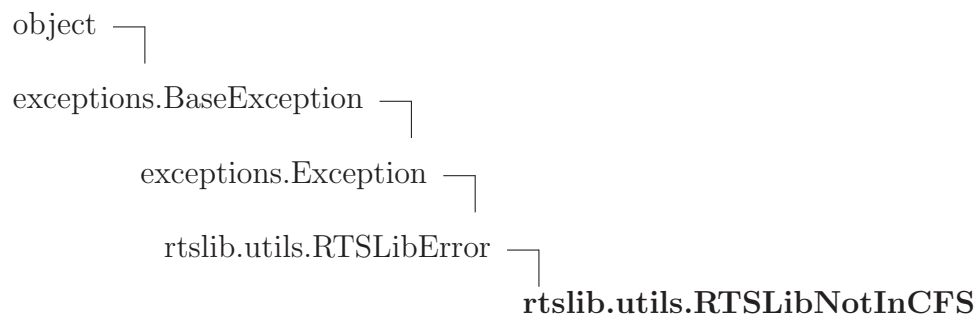
Inherited from object

__format__(), __hash__(), __reduce_ex__(), __sizeof__(), __subclasshook__()

7.4.2 Properties

Name	Description
Inherited from exceptions.BaseException	
args, message	
Inherited from object	
__class__	

7.5 Class RTSLibNotInCFS



The underlying configfs object does not exist. Happens when calling methods of an object that is instanciated but have been deleted from congifs, or when trying to lookup an object that does not exist.

7.5.1 Methods

Inherited from exceptions.Exception

__init__(), __new__()

Inherited from exceptions.BaseException

__delattr__(), __getattribute__(), __getitem__(), __getslice__(), __reduce__(), __repr__(), __setattr__(), __setstate__(), __str__(), __unicode__()

Inherited from object

__format__(), __hash__(), __reduce_ex__(), __sizeof__(), __subclasshook__()

7.5.2 Properties

Name	Description
Inherited from exceptions.BaseException args, message	
Inherited from object __class__	

Index