Manage LUNs

REST API reference

NetApp
February 07, 2026

Table of Contents

Manage LUNs

Manage LUNs

Overview

A LUN is the logical representation of storage in a storage area network (SAN).

The LUN REST API allows you to create, update, delete, and discover LUNs.

A LUN must be mapped to an initiator group to grant access to the initiator group's initiators (client hosts). Initiators can then access the LUN and perform I/O over a Fibre Channel (FC) fabric using the FC Protocol or a TCP/IP network using iSCSI.

See the LUN object model to learn more about each of the properties supported by the LUN REST API.

A LUN is located within a volume. Optionally, it can be located within a qtree in a volume.

LUN names are paths of the form "/vol/<volume>[/<qtree>]/<lun>" where the qtree name is optional.

A LUN can be created to a specified size using thin or thick provisioning. A LUN can then be renamed, resized, cloned, moved to a different volume and copied. LUNs support the assignment of a QoS policy for performance management or a QoS policy can be assigned to a volume containing one or more LUNs.

Performance monitoring

Performance of a LUN can be monitored by observing the `metric.*` and `statistics.*` properties. These properties show the space utilization and performance of a LUN in terms of IOPS, latency and throughput. The `metric.*` properties denote an average whereas `statistics.*` properties denote a real-time monotonically increasing value aggregated across all nodes.

Examples

Creating a LUN

This example creates a 300 gigabyte, thin-provisioned LUN in SVM *svm1*, volume *vol1*, configured for use by *linux* initiators. The `return_records` query parameter is used to retrieve properties of the newly created LUN in the POST response.

```
# The API:
POST /api/storage/luns

# The call:
curl -X POST 'https://<mgmt-ip>/api/storage/luns?return_records=true' -H
'Accept: application/hal+json' -d '{ "svm": { "name": "svm1" }, "os_type":
"linux", "space": { "size": "300G" }, "name" : "/vol/vol1/lun1" }'

# The response:
{
"num_records": 1,
```

```
"records": [
  {
    "uuid": "5a24ae5b-28af-47fb-b129-5adf6cfba0a6",
    "svm": {
      "uuid": "6bf967fd-2a1c-11e9-b682-005056bbc17d",
      "name": "svm1",
      "_links": {
        "self": {
          "href": "/api/svm/svms/6bf967fd-2a1c-11e9-b682-005056bbc17d"
        }
      }
    },
    "name": "/vol/vol1/lun1",
    "location": {
      "logical_unit": "lun1",
      "volume": {
        "uuid": "71cd0dba-2a1c-11e9-b682-005056bbc17d",
        "name": "vol1",
        "_links": {
          "self": {
            "href": "/api/storage/volumes/71cd0dba-2a1c-11e9-b682-
005056bbc17d"
          }
        }
      }
    },
    "class": "regular",
    "enabled": true,
    "os_type": "linux",
    "serial_number": "wf0Iq+N4uck3",
    "space": {
      "guarantee": {
        "requested": false,
        "reserved": false
      },
      "scsi_thin_provisioning_support_enabled": true,
      "size": 322163441664,
      "used": 0
    },
    "status": {
      "container_state": "online",
      "read_only": false,
      "state": "online"
    },
    "_links": {
      "self": {
```

```
        "href": "/api/storage/luns/5a24ae5b-28af-47fb-b129-5adf6cfba0a6"
      }
    }
  }
]
}
```

Updating a LUN

This example sets the `comment` property of a LUN.

```
# The API:
PATCH /api/storage/luns/{uuid}

# The call:
curl -X PATCH 'https://<mgmt-ip>/api/storage/luns/5a24ae5b-28af-47fb-b129-
5adf6cfba0a6' -H 'Accept: application/hal+json' -d '{ "comment": "Data for
the finance department." }'
```

Retrieving LUNs

This example retrieves summary information for all online LUNs in SVM *svm1*. The `svm.name` and `status.state` query parameters are used to find the desired LUNs.

```
# The API:
GET /api/storage/luns

# The call:
curl -X GET 'https://<mgmt-
ip>/api/storage/luns?svm.name=svm1&status.state=online' -H 'Accept:
application/hal+json'

# The response:
{
"records": [
  {
    "uuid": "5a24ae5b-28af-47fb-b129-5adf6cfba0a6",
    "svm": {
      "name": "svm1"
    },
    "name": "/vol/vol1/lun1",
    "status": {
      "state": "online"
    },
    "_links": {
```

```
      "self": {
        "href": "/api/storage/luns/5a24ae5b-28af-47fb-b129-5adf6cfba0a6"
      }
    }
  },
  {
    "uuid": "c903a978-9bac-4ce9-8237-4a3ba8b13f08",
    "svm": {
      "name": "svm1"
    },
    "name": "/vol/vol1/lun2",
    "status": {
      "state": "online"
    },
    "_links": {
      "self": {
        "href": "/api/storage/luns/c903a978-9bac-4ce9-8237-4a3ba8b13f08"
      }
    }
  },
  {
    "uuid": "7faf0a9e-0a47-4876-8318-3638d5da16bf",
    "svm": {
      "name": "svm1"
    },
    "name": "/vol/vol2/lun3",
    "status": {
      "state": "online"
    },
    "_links": {
      "self": {
        "href": "/api/storage/luns/7faf0a9e-0a47-4876-8318-3638d5da16bf"
      }
    }
  }
],
"num_records": 3,
"_links": {
  "self": {
    "href": "/api/storage/luns?svm.name=svm1&status.state=online"
  }
}
}
```

Retrieving details for a specific LUN

In this example, the `fields` query parameter is used to request all fields, including advanced fields, that would not otherwise be returned by default for the LUN.

```
# The API:
GET /api/storage/luns/{uuid}

# The call:
curl -X GET 'https://<mgmt-ip>/api/storage/luns/5a24ae5b-28af-47fb-b129-
5adf6cfba0a6?fields=**' -H 'Accept: application/hal+json'

# The response:
{
"uuid": "5a24ae5b-28af-47fb-b129-5adf6cfba0a6",
"svm": {
  "uuid": "6bf967fd-2a1c-11e9-b682-005056bbc17d",
  "name": "svm1",
  "_links": {
    "self": {
      "href": "/api/svm/svms/6bf967fd-2a1c-11e9-b682-005056bbc17d"
    }
  }
},
"name": "/vol/vol1/lun1",
"location": {
  "logical_unit": "lun1",
  "volume": {
    "uuid": "71cd0dba-2a1c-11e9-b682-005056bbc17d",
    "name": "vol1",
    "_links": {
      "self": {
        "href": "/api/storage/volumes/71cd0dba-2a1c-11e9-b682-
005056bbc17d"
      }
    }
  }
},
"auto_delete": false,
"class": "vvol",
"comment": "Data for the finance department.",
"enabled": true,
"lun_maps": [
  {
    "logical_unit_number": 0,
    "igroup": {
```

```
      "uuid": "2b9d57e1-2a66-11e9-b682-005056bbc17d",
      "name": "ig1",
      "_links": {
        "self": {
          "href": "/api/protocols/san/igroups/2b9d57e1-2a66-11e9-b682-
005056bbc17d"
        }
      }
    },
    "_links": {
      "self": {
        "href": "/api/protocols/san/lun-maps/5a24ae5b-28af-47fb-b129-
5adf6cfba0a6/2b9d57e1-2a66-11e9-b682-005056bbc17d"
      }
    }
  }
],
"os_type": "linux",
"serial_number": "wf0Iq+N4uck3",
"space": {
  "guarantee": {
    "requested": false,
    "reserved": false
  },
  "scsi_thin_provisioning_support_enabled": true,
  "size": 322163441664,
  "used": 0
},
"vvol": {
  "is_bound": true,
  "bindings": [
    {
      "id": 4304512,
      "partner": {
        "uuid": "353c7262-be4b-4176-acf3-f1021faa8b64",
        "name": "/vol/vol1/pelun1",
        "_links": {
          "self": {
            "href": "/api/storage/luns/353c7262-be4b-4176-acf3-
f1021faa8b64"
          }
        }
      },
      "_links": {
        "self": {
          "href": "/api/protocols/san/vvol-bindings/353c7262-be4b-4176-
```

```
acf3-f1021faa8b64/5a24ae5b-28af-47fb-b129-5adf6cfba0a6"
        }
      }
    }
  ]
},
 "metric": {
  "timestamp": "2019-04-09T05:50:15Z",
  "duration": "PT15S",
  "status": "ok",
  "latency": {
    "other": 0,
    "total": 0,
    "read": 0,
    "write": 0
  },
  "iops": {
    "read": 0,
    "write": 0,
    "other": 0,
    "total": 0
  },
  "throughput": {
    "read": 0,
    "write": 0,
    "other": 0,
    "total": 0
  }
},
"statistics": {
  "timestamp": "2019-04-09T05:50:42Z",
  "status": "ok",
  "latency_raw": {
    "other": 38298,
    "total": 38298,
    "read": 0,
    "write": 0
  },
  "iops_raw": {
    "read": 0,
    "write": 0,
    "other": 3,
    "total": 3
  },
  "throughput_raw": {
    "read": 0,
```

```
    "write": 0,
    "other": 0,
    "total": 0
  }
},
"status": {
  "container_state": "online",
  "mapped": true,
  "read_only": false,
  "state": "online"
},
"consistency_group": {
  "name": "vol1",
  "uuid": "6d657aaf-b57a-5396-82ea-c01329e46c79",
  "_links": {
    "self": {
      "href": "/api/application/consistency-groups/6d657aaf-b57a-5396-
82ea-c01329e46c79"
    }
  }
},
"_links": {
  "self": {
    "href": "/api/storage/luns/5a24ae5b-28af-47fb-b129-
5adf6cfba0a6?fields=**"
  }
}
}
```

Deleting a LUN

```
# The API:
DELETE /api/storage/luns/{uuid}

# The call:
curl -X DELETE 'https://<mgmt-ip>/api/storage/luns/c903a978-9bac-4ce9-
8237-4a3ba8b13f08' -H 'Accept: application/hal+json'
```

LUN data

The LUN REST API also supports reading data from and writing data to a LUN via the REST API as multipart/form-data.

Reading data is performed using a GET request on the LUN endpoint. The request header must include

`Accept: multipart/form-data`. When this header entry is provided, query parameters `data.offset` and `data.size` are required and used to specify the portion of the LUN's data to read; no other query parameters are allowed. Reads are limited to one megabyte (1MB) per request. Data is returned as `multipart/form-data` content with exactly one form entry containing the data. The form entry has content type `application/octet-stream`.

Writing data is performed using a PATCH request on the LUN endpoint. The request header must include `Content-Type: multipart/form-data`. When this header entry is provided, query parameter `data.offset` is required and used to specify the location within the LUN at which to write the data; no other query parameters are allowed. The request body must be `multipart/form-data` content with exactly one form entry containing the data to write. The content type entry of the form data is ignored and always treated as `application/octet-stream`. Writes are limited to one megabyte (1MB) per request.

Reading data from a LUN

```
# The API:
GET /api/storage/luns/{uuid}

# The call:
curl -X GET 'https://<mgmt-ip>/api/storage/luns/c903a978-9bac-4ce9-8237-
4a3ba8b13f08?data.offset=0&data.size=9' -H 'Accept: multipart/form-data'

# In the response header:
Content-Type: multipart/form-data; boundary="c6e9cf51ab354af0"

# The response body:
--c6e9cf51ab354af0
Content-Disposition: form-data;
Content-Type: application/octet-stream
data here
--c6e9cf51ab354af0--
```

Writing data to a LUN

This example writes the contents of a file to a LUN starting at offset 1024.

```
# The API:
PATCH /api/storage/luns/{uuid}

# The call:
curl -X PATCH 'https://<mgmt-ip>/api/storage/luns/c903a978-9bac-4ce9-8237-
4a3ba8b13f08?data.offset=1024' -F "data=@file;type=application/octet-
stream"
```

Cloning LUNs

A clone of a LUN is an independent "copy" of the LUN that shares unchanged data blocks with the original. As blocks of the source and clone are modified, unique blocks are written for each. LUN clones can be created quickly and consume very little space initially. They can be created for the purpose of back-up, or to replicate data for multiple consumers.

Space reservations can be set for the LUN clone independent of the source LUN by setting the `space.guarantee.requested` property in a POST or PATCH request.

A LUN clone can be set to auto-delete by setting the `auto_delete` property. If the LUN's volume is configured for automatic deletion, LUNs that have auto-delete enabled are deleted when a volume is nearly full to reclaim a target amount of free space in the volume.

The value of property `space.scsi_thin_provisioning_support_enabled` is not propagated to the destination when a LUN is cloned as a new LUN; it is reset to false. The value of this property is maintained from the destination LUN when a LUN is overwritten as a clone.

Creating a new LUN clone

You create a new LUN clone as you create any LUN - a POST request to /storage/luns. Set `clone.source.uuid` or `clone.source.name` to identify the source LUN from which the clone is created. The LUN clone and its source must reside in the same volume.

The source LUN can reside in a snapshot, in which case the `clone.source.name` field must be used to identify it. Add `/.snapshot/<snapshot_name>` to the path after the volume name to identify the snapshot. For example `/vol/vol1/.snapshot/snap1/lun1`.

By default, new LUN clones do not inherit the QoS policy of the source LUN; a QoS policy should be set for the clone by setting the `qos_policy` property.

```
# The API:
POST /api/storage/luns

# The call:
curl -X POST 'https://<mgmt-ip>/api/storage/luns' -H 'Accept:
application/hal+json' -d '{ "svm": { "name": "svm1" }, "name":
"/vol/vol1/lun2clone1", "clone": { "source": { "name": "/vol/vol1/lun2" }
}, "qos_policy": { "name": "qos1" } }'
```

Over-writing an existing LUN's data as a clone of another

You can overwrite an existing LUN as a clone of another, using a PATCH request to /storage/luns/{uuid}. Set the `clone.source.uuid` or `clone.source.name` property to identify the source LUN from which the clone data is taken. The LUN clone and its source must reside in the same volume.

When used in a PATCH request, the patched LUN's data is overwritten as a clone of the source. The following properties are preserved from the patched LUN unless otherwise specified as part of the PATCH: `class`, `auto_delete`, `lun_maps`, `vvol`, `serial_number`, `status.state`, and `uuid`.

Persistent reservations for the updated LUN are also preserved.

```
# The API:
PATCH /api/storage/luns/{uuid}

# The call:
curl -X PATCH 'https://<mgmt-ip>/api/storage/luns/5a24ae5b-28af-47fb-b129-
5adf6cfba0a6' -H 'Accept: application/hal+json' -d '{ "clone": { "source":
{ "name": "/vol/vol1/lun2" } } }'
```

Converting an NVMe namespace into a LUN

An existing NVMe namespace can be converted in-place to a LUN with no modification to the data blocks. In other words, there is no additional copy created for the data blocks. There are certain requirements for converting an NVMe namespace to a LUN. For instance, the namespace should not be mapped to an NVMe subsystem. Additionally, the namespace should not have a block size other than 512 bytes.

The conversion process updates the metadata to the NVMe namespace, making it a LUN. The conversion is both time and space efficient. After conversion, the new LUN behaves as a regular LUN and may be mapped to an initiator group.

Convert an NVMe namespace into a LUN

You convert an NVMe namespace into a LUN by calling a POST to /storage/luns. Set `convert.namespace.uuid` or `convert.namespace.name` to identify the source NVMe namespace which is to be converted in-place into a LUN.

```
# The API:
POST /api/storage/luns

# The call:
curl -X POST 'https://<mgmt-ip>/api/storage/luns' -H 'Accept:
application/hal+json' -d '{ "svm": { "name": "svm1" }, "convert": {
"namespace": { "name": "/vol/vol1/namespace1" } } }'
```

Moving LUNs between volumes

You move a LUN between volumes by using a PATCH request to /storage/luns/{uuid}. Set the volume portion of the fully qualified LUN path `name` property, `path.volume.uuid`, or `path.volume.name` property to a different volume than the LUN's current volume. Moving a LUN between volumes is an asynchronous activity. A successful request returns a response of 200 synchronously, which indicates that the movement has been successfully queued. The LUN object can then be further polled with a GET request to `/storage/luns/{uuid}` to monitor the status of the movement.

The `movement` sub-object of the LUN object is populated while a LUN movement is in progress and for two minutes following completion of a movement.

Starting a LUN movement

```
# The API:
PATCH /api/storage/luns/{uuid}

# The call:
curl -X PATCH 'https://<mgmt-ip>/api/storage/luns/7faf0a9e-0a47-4876-8318-
3638d5da16bf' -H 'Accept: application/hal+json' -d '{ "name":
"/vol/vol1/lun3" }'
```

Checking on the status of the LUN movement

```
# The API:
GET /api/storage/luns/{uuid}

# The call:
curl -X GET 'https://<mgmt-ip>/api/storage/luns/7faf0a9e-0a47-4876-8318-
3638d5da16bf?fields=movement' -H 'Accept: application/hal+json'

# The response:
{
"uuid": "7faf0a9e-0a47-4876-8318-3638d5da16bf",
"name": "/vol/vol1/lun3",
"movement": {
  "paths": {
    "destination": "/vol/vol1/lun3",
    "source": "/vol/vol2/lun3"
  },
  "progress": {
    "elapsed": 1,
    "percent_complete": 0,
    "state": "preparing",
    "volume_snapshot_blocked": false
  }
},
"_links": {
  "self": {
    "href": "/api/storage/luns/7faf0a9e-0a47-4876-8318-3638d5da16bf"
  }
}
}
```

Retrieve LUNs

GET `/storage/luns`

Introduced In: 9.6

Retrieves LUNs.

Expensive properties

There is an added computational cost to retrieving values for these properties. They are not included by default in GET results and must be explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

- `attributes.*`
- `auto_delete`
- `copy.*`
- `lun_maps.*`
- `movement.*`
- `space.physical_used`
- `space.physical_used_by_snapshots`
- `space.efficiency_ratio`
- `statistics.*`
- `vvol.bindings.*`
- `metric.*`

Related ONTAP commands

- `lun bind show`
- `lun copy show`
- `lun mapping show`
- `lun move show`
- `lun show`
- `volume file clone show-autodelete`

Learn more

- DOC /storage/luns

Parameters

Name	Type	In	Required	Description
svm.name	string	query	False	Filter by svm.name
svm.uuid	string	query	False	Filter by svm.uuid
class	string	query	False	Filter by class
space.guarantee.reserved	boolean	query	False	Filter by space.guarantee.reserved
space.guarantee.requested	boolean	query	False	Filter by space.guarantee.requested
space.size	integer	query	False	Filter by space.size • Max value: 140737488355328 • Min value: 4096
space.used	integer	query	False	Filter by space.used
space.efficiency_ratio	number	query	False	Filter by space.efficiency_ratio • Introduced in: 9.16
space.physical_used_by_snapshots	integer	query	False	Filter by space.physical_used_by_snapshots • Introduced in: 9.16
space.physical_used	integer	query	False	Filter by space.physical_used • Introduced in: 9.16

Name	Type	In	Required	Description
space.scsi_thin_provisioning_support_enabled	boolean	query	False	Filter by space.scsi_thin_provisioning_support_enabled • Introduced in: 9.10
metric.status	string	query	False	Filter by metric.status • Introduced in: 9.7
metric.latency.other	integer	query	False	Filter by metric.latency.other • Introduced in: 9.7
metric.latency.total	integer	query	False	Filter by metric.latency.total • Introduced in: 9.7
metric.latency.read	integer	query	False	Filter by metric.latency.read • Introduced in: 9.7
metric.latency.write	integer	query	False	Filter by metric.latency.write • Introduced in: 9.7
metric.timestamp	string	query	False	Filter by metric.timestamp • Introduced in: 9.7

Name	Type	In	Required	Description
metric.throughput.other	integer	query	False	Filter by metric.throughput.other • Introduced in: 9.7
metric.throughput.total	integer	query	False	Filter by metric.throughput.total • Introduced in: 9.7
metric.throughput.read	integer	query	False	Filter by metric.throughput.read • Introduced in: 9.7
metric.throughput.write	integer	query	False	Filter by metric.throughput.write • Introduced in: 9.7
metric.duration	string	query	False	Filter by metric.duration • Introduced in: 9.7
metric.iops.other	integer	query	False	Filter by metric.iops.other • Introduced in: 9.7
metric.iops.total	integer	query	False	Filter by metric.iops.total • Introduced in: 9.7

Name	Type	In	Required	Description
metric.iops.read	integer	query	False	Filter by metric.iops.read • Introduced in: 9.7
metric.iops.write	integer	query	False	Filter by metric.iops.write • Introduced in: 9.7
comment	string	query	False	Filter by comment • maxLength: 254 • minLength: 0
attributes.value	string	query	False	Filter by attributes.value • Introduced in: 9.10 • maxLength: 4091 • minLength: 1
attributes.name	string	query	False	Filter by attributes.name • Introduced in: 9.10 • maxLength: 4091 • minLength: 1
movement.paths.source	string	query	False	Filter by movement.paths.source
movement.paths.destination	string	query	False	Filter by movement.paths.destination
movement.progress.elapsed	integer	query	False	Filter by movement.progress.elapsed

Name	Type	In	Required	Description
movement.progress.state	string	query	False	Filter by movement.progress.state
movement.progress.percent_complete	integer	query	False	Filter by movement.progress.percent_complete • Max value: 100 • Min value: 0
movement.progress.failure.arguments.code	string	query	False	Filter by movement.progress.failure.arguments.code
movement.progress.failure.arguments.message	string	query	False	Filter by movement.progress.failure.arguments.message
movement.progress.failure.message	string	query	False	Filter by movement.progress.failure.message
movement.progress.failure.code	string	query	False	Filter by movement.progress.failure.code
movement.progress.volume_snapshot_blocked	boolean	query	False	Filter by movement.progress.volume_snapshot_blocked
movement.max_throughput	integer	query	False	Filter by movement.max_throughput
lun_maps.logical_unit_number	integer	query	False	Filter by lun_maps.logical_unit_number

Name	Type	In	Required	Description
lun_maps.igroup.protocol	string	query	False	Filter by lun_maps.igroup.protocol • Introduced in: 9.16
lun_maps.igroup.comment	string	query	False	Filter by lun_maps.igroup.comment • Introduced in: 9.16 • maxLength: 254 • minLength: 0
lun_maps.igroup.igroups.uuid	string	query	False	Filter by lun_maps.igroup.igroups.uuid • Introduced in: 9.16
lun_maps.igroup.igroups.name	string	query	False	Filter by lun_maps.igroup.igroups.name • Introduced in: 9.16 • maxLength: 96 • minLength: 1
lun_maps.igroup.initiators.name	string	query	False	Filter by lun_maps.igroup.initiators.name • Introduced in: 9.16
lun_maps.igroup.initiators.comment	string	query	False	Filter by lun_maps.igroup.initiators.comment • Introduced in: 9.16 • maxLength: 254 • minLength: 0

Name	Type	In	Required	Description
lun_maps.igroup.os_type	string	query	False	Filter by lun_maps.igroup.os_type • Introduced in: 9.16
lun_maps.igroup.name	string	query	False	Filter by lun_maps.igroup.name
lun_maps.igroup.uuid	string	query	False	Filter by lun_maps.igroup.uuid
status.container_state	string	query	False	Filter by status.container_state
status.read_only	boolean	query	False	Filter by status.read_only
status.state	string	query	False	Filter by status.state
status.mapped	boolean	query	False	Filter by status.mapped
qos_policy.uuid	string	query	False	Filter by qos_policy.uuid
qos_policy.name	string	query	False	Filter by qos_policy.name
serial_number_hex	string	query	False	Filter by serial_number_hex • Introduced in: 9.17
vvol.is_bound	boolean	query	False	Filter by vvol.is_bound • Introduced in: 9.10

Name	Type	In	Required	Description
vvol.bindings.secondary_id	string	query	False	Filter by vvol.bindings.secondary_id • Introduced in: 9.13
vvol.bindings.partner.name	string	query	False	Filter by vvol.bindings.partner.name • Introduced in: 9.10
vvol.bindings.partner.uuid	string	query	False	Filter by vvol.bindings.partner.uuid • Introduced in: 9.10
vvol.bindings.id	integer	query	False	Filter by vvol.bindings.id • Introduced in: 9.10
serial_number	string	query	False	Filter by serial_number • maxLength: 12 • minLength: 12
uuid	string	query	False	Filter by uuid
consistency_group.uuid	string	query	False	Filter by consistency_group.uuid • Introduced in: 9.10
consistency_group.name	string	query	False	Filter by consistency_group.name • Introduced in: 9.10

Name	Type	In	Required	Description
location.logical_unit	string	query	False	Filter by location.logical_unit
location.qtree.name	string	query	False	Filter by location.qtree.name
location.qtree.id	integer	query	False	Filter by location.qtree.id • Max value: 4994 • Min value: 0
location.volume.uuid	string	query	False	Filter by location.volume.uuid
location.volume.name	string	query	False	Filter by location.volume.name
location.node.uuid	string	query	False	Filter by location.node.uuid • Introduced in: 9.10
location.node.name	string	query	False	Filter by location.node.name • Introduced in: 9.10
statistics.throughput_raw.other	integer	query	False	Filter by statistics.throughput_raw.other • Introduced in: 9.7
statistics.throughput_raw.total	integer	query	False	Filter by statistics.throughput_raw.total • Introduced in: 9.7

Name	Type	In	Required	Description
statistics.throughput_raw.read	integer	query	False	Filter by statistics.throughput_raw.read • Introduced in: 9.7
statistics.throughput_raw.write	integer	query	False	Filter by statistics.throughput_raw.write • Introduced in: 9.7
statistics.iops_raw.other	integer	query	False	Filter by statistics.iops_raw.other • Introduced in: 9.7
statistics.iops_raw.total	integer	query	False	Filter by statistics.iops_raw.total • Introduced in: 9.7
statistics.iops_raw.read	integer	query	False	Filter by statistics.iops_raw.read • Introduced in: 9.7
statistics.iops_raw.write	integer	query	False	Filter by statistics.iops_raw.write • Introduced in: 9.7
statistics.latency_raw.other	integer	query	False	Filter by statistics.latency_raw.other • Introduced in: 9.7

Name	Type	In	Required	Description
statistics.latency_raw.total	integer	query	False	Filter by statistics.latency_raw.total • Introduced in: 9.7
statistics.latency_raw.read	integer	query	False	Filter by statistics.latency_raw.read • Introduced in: 9.7
statistics.latency_raw.write	integer	query	False	Filter by statistics.latency_raw.write • Introduced in: 9.7
statistics.timestamp	string	query	False	Filter by statistics.timestamp • Introduced in: 9.7
statistics.status	string	query	False	Filter by statistics.status • Introduced in: 9.7
enabled	boolean	query	False	Filter by enabled
copy.destinations.uuid	string	query	False	Filter by copy.destinations.uuid • Introduced in: 9.10
copy.destinations.name	string	query	False	Filter by copy.destinations.name • Introduced in: 9.10

Name	Type	In	Required	Description
copy.destinations.progress.volume_snapshot_blocked	boolean	query	False	Filter by copy.destinations.progress.volume_snapshot_blocked • Introduced in: 9.10
copy.destinations.progress.failure.arguments.code	string	query	False	Filter by copy.destinations.progress.failure.arguments.code • Introduced in: 9.10
copy.destinations.progress.failure.arguments.message	string	query	False	Filter by copy.destinations.progress.failure.arguments.message • Introduced in: 9.10
copy.destinations.progress.failure.message	string	query	False	Filter by copy.destinations.progress.failure.message • Introduced in: 9.10
copy.destinations.progress.failure.code	string	query	False	Filter by copy.destinations.progress.failure.code • Introduced in: 9.10
copy.destinations.progress.state	string	query	False	Filter by copy.destinations.progress.state • Introduced in: 9.10

Name	Type	In	Required	Description
copy.destinations.progress.percent_complete	integer	query	False	Filter by copy.destinations.progress.percent_complete • Introduced in: 9.10 • Max value: 100 • Min value: 0
copy.destinations.progress.elapsed	integer	query	False	Filter by copy.destinations.progress.elapsed • Introduced in: 9.10
copy.destinations.peer.uuid	string	query	False	Filter by copy.destinations.peer.uuid • Introduced in: 9.14
copy.destinations.peer.name	string	query	False	Filter by copy.destinations.peer.name • Introduced in: 9.14
copy.destinations.max_throughput	integer	query	False	Filter by copy.destinations.max_throughput • Introduced in: 9.10
copy.source.max_throughput	integer	query	False	Filter by copy.source.max_throughput • Introduced in: 9.10

Name	Type	In	Required	Description
copy.source.peer.uuid	string	query	False	Filter by copy.source.peer.uuid • Introduced in: 9.14
copy.source.peer.name	string	query	False	Filter by copy.source.peer.name • Introduced in: 9.14
copy.source.progress.volume_snapshot_blocked	boolean	query	False	Filter by copy.source.progress.volume_snapshot_blocked • Introduced in: 9.10
copy.source.progress.state	string	query	False	Filter by copy.source.progress.state • Introduced in: 9.10
copy.source.progress.failure.arguments.code	string	query	False	Filter by copy.source.progress.failure.arguments.code • Introduced in: 9.10
copy.source.progress.failure.arguments.message	string	query	False	Filter by copy.source.progress.failure.arguments.message • Introduced in: 9.10

Name	Type	In	Required	Description
copy.source.progress.failure.message	string	query	False	Filter by copy.source.progress.failure.message • Introduced in: 9.10
copy.source.progress.failure.code	string	query	False	Filter by copy.source.progress.failure.code • Introduced in: 9.10
copy.source.progress.percent_complete	integer	query	False	Filter by copy.source.progress.percent_complete • Introduced in: 9.10 • Max value: 100 • Min value: 0
copy.source.progress.elapsed	integer	query	False	Filter by copy.source.progress.elapsed • Introduced in: 9.10
copy.source.name	string	query	False	Filter by copy.source.name • Introduced in: 9.10
copy.source.uuid	string	query	False	Filter by copy.source.uuid • Introduced in: 9.10
auto_delete	boolean	query	False	Filter by auto_delete
os_type	string	query	False	Filter by os_type
name	string	query	False	Filter by name

Name	Type	In	Required	Description
create_time	string	query	False	Filter by create_time • Introduced in: 9.7
fields	array[string]	query	False	Specify the fields to return.
max_records	integer	query	False	Limit the number of records returned.
return_records	boolean	query	False	The default is true for GET calls. When set to false, only the number of records is returned. • Default value: 1
return_timeout	integer	query	False	The number of seconds to allow the call to execute before returning. When iterating over a collection, the default is 15 seconds. ONTAP returns earlier if either max records or the end of the collection is reached. • Max value: 120 • Min value: 0 • Default value: 15
order_by	array[string]	query	False	Order results by specified fields and optional [asc

Response

```
Status: 200, Ok
```

Name	Type	Description
_links	_links	
num_records	integer	The number of records in the response.
records	array[lun]	

Example response


```
{
  "_links": {
    "next": {
      "href": "/api/resourcelink"
    },
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "num_records": 1,
  "records": [
    {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "attributes": [
        {
          "_links": {
            "self": {
              "href": "/api/resourcelink"
            }
          },
          "name": "name1",
          "value": "value1"
        }
      ],
      "class": "string",
      "comment": "string",
      "consistency_group": {
        "_links": {
          "self": {
            "href": "/api/resourcelink"
          }
        },
        "name": "cg1",
        "uuid": "4abc2317-4332-9d37-93a0-20bd29c22df0"
      },
      "copy": {
        "destinations": [
          {
            "_links": {
              "self": {
```

```
            "href": "/api/resourcelink"
          }
        },
        "max_throughput": 0,
        "name": "/vol/vol1/lun1",
        "peer": {
          "_links": {
            "self": {
              "href": "/api/resourcelink"
            }
          },
          "name": "peer1",
          "uuid": "4204cf77-4c82-9bdb-5644-b5a841c097a9"
        },
        "progress": {
          "elapsed": 0,
          "failure": {
            "arguments": [
              {
                "code": "string",
                "message": "string"
              }
            ],
            "code": "4",
            "message": "entry doesn't exist"
          },
          "percent_complete": 0,
          "state": "string"
        },
        "uuid": "1bc327d5-4654-5284-a116-f182282240b4"
      }
    ],
    "source": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "name": "/vol/vol2/lun1",
      "peer": {
        "_links": {
          "self": {
            "href": "/api/resourcelink"
          }
        },
        "name": "peer1",
```

```
        "uuid": "4204cf77-4c82-9bdb-5644-b5a841c097a9"
      },
      "progress": {
        "elapsed": 0,
        "failure": {
          "arguments": [
            {
              "code": "string",
              "message": "string"
            }
          ],
          "code": "4",
          "message": "entry doesn't exist"
        },
        "percent_complete": 0,
        "state": "string"
      },
      "uuid": "03c05019-40d9-3945-c767-dca4c3be5e90"
    }
  },
  "create_time": "2018-06-04 15:00:00 -0400",
  "location": {
    "logical_unit": "lun1",
    "node": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "name": "node1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    },
    "qtree": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "id": 1,
      "name": "qt1"
    },
    "volume": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
```

```
      },
      "name": "volume1",
      "uuid": "028baa66-41bd-11e9-81d5-00a0986138f7"
    }
  },
  "lun_maps": [
    {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "igroup": {
        "_links": {
          "self": {
            "href": "/api/resourcelink"
          }
        },
        "comment": "string",
        "igroups": [
          {
            "_links": {
              "self": {
                "href": "/api/resourcelink"
              }
            },
            "name": "igroup1",
            "uuid": "4ea7a442-86d1-11e0-ae1c-123478563412"
          }
        ],
        "initiators": [
          {
            "comment": "my comment",
            "name": "iqn.1998-01.com.corp.iscsi:name1"
          }
        ],
        "name": "igroup1",
        "os_type": "string",
        "protocol": "string",
        "uuid": "4ea7a442-86d1-11e0-ae1c-123478563412"
      }
    }
  ],
  "metric": {
    "_links": {
      "self": {
```

```
            "href": "/api/resourcelink"
          }
        },
        "duration": "PT15S",
        "iops": {
          "read": 200,
          "total": 1000,
          "write": 100
        },
        "latency": {
          "read": 200,
          "total": 1000,
          "write": 100
        },
        "status": "ok",
        "throughput": {
          "read": 200,
          "total": 1000,
          "write": 100
        },
        "timestamp": "2017-01-25 06:20:13 -0500"
      },
      "movement": {
        "paths": {
          "destination": "/vol/vol1/lun1",
          "source": "/vol/vol2/lun2"
        },
        "progress": {
          "elapsed": 0,
          "failure": {
            "arguments": [
              {
                "code": "string",
                "message": "string"
              }
            ],
            "code": "4",
            "message": "entry doesn't exist"
          },
          "percent_complete": 0,
          "state": "string"
        }
      },
      "name": "/vol/volume1/qtree1/lun1",
      "os_type": "string",
      "qos_policy": {
```

```
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "name": "qos1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    },
    "serial_number": "string",
    "serial_number_hex": "string",
    "space": {
      "efficiency_ratio": 2.5,
      "physical_used": 1073741824,
      "physical_used_by_snapshots": 1073741824,
      "size": 1073741824,
      "used": 0
    },
    "statistics": {
      "iops_raw": {
        "read": 200,
        "total": 1000,
        "write": 100
      },
      "latency_raw": {
        "read": 200,
        "total": 1000,
        "write": 100
      },
      "status": "ok",
      "throughput_raw": {
        "read": 200,
        "total": 1000,
        "write": 100
      },
      "timestamp": "2017-01-25 06:20:13 -0500"
    },
    "status": {
      "container_state": "string",
      "state": "online"
    },
    "svm": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
```

```
        "name": "svm1",
        "uuid": "02c9e252-41be-11e9-81d5-00a0986138f7"
      },
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412",
      "vvol": {
        "bindings": [
          {
            "_links": {
              "self": {
                "href": "/api/resourcelink"
              }
            },
            "id": 1,
            "partner": {
              "_links": {
                "self": {
                  "href": "/api/resourcelink"
                }
              },
              "name": "/vol/vol1/lun1",
              "uuid": "4ea7a442-86d1-11e0-ae1c-123478563412"
            },
            "secondary_id": "0000D20000010000h"
          }
        ]
      }
    }
  ]
}
```


Error

```
Status: Default, Error
```

Name	Type	Description
error	returned_error	

Example error


```
{
  "error": {
    "arguments": [
      {
        "code": "string",
        "message": "string"
      }
    ],
    "code": "4",
    "message": "entry doesn't exist",
    "target": "uuid"
  }
}
```


Definitions

See Definitions

href

Name	Type	Description
href	string	

_links

Name	Type	Description
next	href	
self	href	

_links

Name	Type	Description
self	href	

attributes

A name/value pair optionally stored with the LUN. Attributes are available to callers to persist small amounts of application-specific metadata. They are in no way interpreted by ONTAP.

Attribute names and values must be at least one byte and no more than 4091 bytes in length. The sum of the name and value lengths must be no more than 4092 bytes.

Optional in POST.

Name	Type	Description
_links	_links	
name	string	The attribute name.
value	string	The attribute value.

source

The source LUN for a LUN clone operation. This can be specified using property `clone.source.uuid` or `clone.source.name`. If both properties are supplied, they must refer to the same LUN.

Valid in POST to create a new LUN as a clone of the source.

Valid in PATCH to overwrite an existing LUN's data as a clone of another.

clone

This sub-object is used in POST to create a new LUN as a clone of an existing LUN, or PATCH to overwrite an existing LUN as a clone of another. Setting a property in this sub-object indicates that a LUN clone is desired. Consider the following other properties when cloning a LUN: `auto_delete`,

qos_policy, space.guarantee.requested and space.scsi_thin_provisioning_support_enabled.

When used in a PATCH, the patched LUN's data is over-written as a clone of the source and the following properties are preserved from the patched LUN unless otherwise specified as part of the PATCH: class, auto_delete, lun_maps, serial_number, status.state, and uuid.

Persistent reservations for the patched LUN are also preserved.

consistency_group

The LUN's consistency group. This property is populated for LUNs that are members of a consistency group. If the LUN is a member of a child consistency group, the parent consistency group is reported. A LUN's consistency group is the consistency group of its containing volume.

Name	Type	Description
_links	_links	
name	string	The name of the consistency group.
uuid	string	The unique identifier of the consistency group.

namespace

The source namespace for convert operation. This can be specified using property convert.namespace.uuid or convert.namespace.name. If both properties are supplied, they must refer to the same NVMe namespace.

Valid in POST. A convert request from NVMe namespace to LUN cannot be combined with setting any other LUN properties. All other properties of the converted LUN come from the source NVMe namespace.

convert

This sub-object is used in POST to convert a valid in-place NVMe namespace to a LUN. Setting a property in this sub-object indicates that a conversion from the specified NVMe namespace to LUN is desired.

peer

The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM.

This is only populated by GET when the LUN copy is inter-SVM.

Name	Type	Description
_links	_links	
name	string	The local name of the peer SVM. This name is unique among all local and peer SVMs.

Name	Type	Description
uuid	string	The unique identifier of the SVM peer relationship. This is the UUID of the relationship, not the UUID of the peer SVM itself.

error_arguments

Name	Type	Description
code	string	Argument code
message	string	Message argument

error

Error information provided if the asynchronous LUN copy operation fails.

Name	Type	Description
arguments	array[error_arguments]	Message arguments
code	string	Error code
message	string	Error message

progress

Properties related to the progress of an active or recently completed LUN copy.

Name	Type	Description
elapsed	integer	The amount of time that has elapsed since the start of the LUN copy, in seconds.
failure	error	Error information provided if the asynchronous LUN copy operation fails.
percent_complete	integer	The percentage completed of the LUN copy.
state	string	The state of the LUN copy.

Name	Type	Description
volume_snapshot_blocked	boolean	This property reports if volume snapshots are blocked by the LUN copy. This property can be polled to identify when volume snapshots can be resumed after beginning a LUN copy.

destinations

A LUN copy operation in which the containing LUN is the source of the copy.

Name	Type	Description
_links	_links	
max_throughput	integer	The maximum data throughput, in bytes per second, that should be utilized in support of the LUN copy. See property `copy.source.max_throughput` for further details.
name	string	The fully qualified path of the LUN copy destination composed of a "/vol" prefix, the volume name, the optional qtree name, and base name of the LUN.
peer	peer	The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM. This is only populated by GET when the LUN copy is inter-SVM.
progress	progress	Properties related to the progress of an active or recently completed LUN copy.
uuid	string	The unique identifier of the LUN copy destination.

peer

The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM.

Set this in POST to specify the source SVM for an inter-SVM LUN copy. Only populated by GET when the LUN copy is inter-SVM.

Name	Type	Description
_links	_links	
name	string	The local name of the peer SVM. This name is unique among all local and peer SVMs.
uuid	string	The unique identifier of the SVM peer relationship. This is the UUID of the relationship, not the UUID of the peer SVM itself.

progress

Properties related to the progress of an active or recently completed LUN copy.

Name	Type	Description
elapsed	integer	The amount of time that has elapsed since the start of the LUN copy, in seconds.
failure	error	Error information provided if the asynchronous LUN copy operation fails.
percent_complete	integer	The percentage completed of the LUN copy.
state	string	The state of the LUN copy. Valid in PATCH when an LUN copy is active. Set to *paused* to pause a LUN copy. Set to *replicating* to resume a paused LUN copy.
volume_snapshot_blocked	boolean	This property reports if volume snapshots are blocked by the LUN copy. This property can be polled to identify when volume snapshots can be resumed after beginning a LUN copy.

source

The source LUN of a LUN copy operation in which the containing LUN is the destination of the copy.

Valid in POST except when creating a LUN clone. A LUN copy request cannot be combined with setting any other LUN properties except the destination location. All other properties of the destination LUN come from the source LUN.

Name	Type	Description
_links	_links	
max_throughput	integer	The maximum data throughput, in bytes per second, that should be utilized in support of the LUN copy. This property can be used to throttle a transfer and limit its impact on the performance of the source and destination nodes. The specified value will be rounded up to the nearest megabyte. If this property is not specified in a POST that begins a LUN copy, throttling is not applied to the data transfer. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. Valid only in a POST that begins a LUN copy or a PATCH when a LUN copy is already in process. • format: int64 • Introduced in: 9.10 • x-nullable: true
name	string	The fully qualified path of the LUN copy source composed of a "/vol" prefix, the volume name, the optional qtree name, and base name of the LUN. Set this property in POST to specify the source for a LUN copy operation.

Name	Type	Description
peer	peer	The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM. Set this in POST to specify the source SVM for an inter-SVM LUN copy. Only populated by GET when the LUN copy is inter-SVM.
progress	progress	Properties related to the progress of an active or recently completed LUN copy.
uuid	string	The unique identifier of the LUN copy source. Set this property in POST to specify the source for a LUN copy operation.

copy

This sub-object applies to LUN copy operations. A LUN can be copied with a POST request that supplies `copy.source` properties.

Copying a LUN is an asynchronous activity begun by a POST request that specifies the source of the copy in the `copy.source` properties. The data for the LUN is then asynchronously copied from the source to the destination. The time required to complete the copy depends on the size of the LUN and the load on the cluster. The `copy` sub-object is populated while a LUN copy is in progress and for two (2) minutes following completion of a copy.

While LUNs are being copied, the status of the LUN copy operations can be obtained using a GET of the source or destination LUN that requests the `copy` properties. If the LUN is the source LUN for one or more copy operations, the `copy.destinations` array is populated in GET. If the containing LUN is the destination LUN for a copy operation, the `copy.source` sub-object is populated in GET. The LUN copy operation can be further modified using a PATCH on the properties on the `copy.source` sub-object of the copy destination LUN.

There is an added computational cost to retrieving property values for `copy`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
destinations	array[destinations]	An array of destination LUNs of LUN copy operations in which the containing LUN is the source of the copy.
source	source	The source LUN of a LUN copy operation in which the containing LUN is the destination of the copy. Valid in POST except when creating a LUN clone. A LUN copy request cannot be combined with setting any other LUN properties except the destination location. All other properties of the destination LUN come from the source LUN.

node

The cluster node that hosts the LUN.

Name	Type	Description
_links	_links	
name	string	
uuid	string	

qtree

The qtree in which the LUN is optionally located. Valid in POST and PATCH.

If properties `name` and `location.qtree.name` and/or `location.qtree.id` are specified in the same request, they must refer to the same qtree.

A PATCH that modifies the qtree of the LUN is considered a rename operation.

Name	Type	Description
_links	_links	
id	integer	The identifier for the qtree, unique within the qtree's volume.
name	string	The name of the qtree.

volume

The volume in which the LUN is located. Valid in POST and PATCH.

If properties `name` and `location.volume.name` and/or `location.volume.uuid` are specified in the same request, they must refer to the same volume.

A PATCH that modifies the volume of the LUN begins an asynchronous LUN movement operation.

Name	Type	Description
_links	_links	
name	string	The name of the volume. This field cannot be specified in a PATCH method.
uuid	string	Unique identifier for the volume. This corresponds to the instance-uuid that is exposed in the CLI and ONTAPI. It does not change due to a volume move. • example: 028baa66-41bd-11e9-81d5-00a0986138f7 • Introduced in: 9.6 • x-nullable: true

location

The location of the LUN within the ONTAP cluster. LUNs support rename and move between volumes. Valid in POST and PATCH.

Name	Type	Description
logical_unit	string	The base name component of the LUN. Valid in POST and PATCH. If properties `name` and `location.logical_unit` are specified in the same request, they must refer to the base name. A PATCH that modifies the base name of the LUN is considered a rename operation.
node	node	The cluster node that hosts the LUN.

Name	Type	Description
qtree	qtree	The qtree in which the LUN is optionally located. Valid in POST and PATCH. If properties `name` and `location.qtree.name` and/or `location.qtree.id` are specified in the same request, they must refer to the same qtree. A PATCH that modifies the qtree of the LUN is considered a rename operation.
volume	volume	The volume in which the LUN is located. Valid in POST and PATCH. If properties `name` and `location.volume.name` and/or `location.volume.uuid` are specified in the same request, they must refer to the same volume. A PATCH that modifies the volume of the LUN begins an asynchronous LUN movement operation.

self_link

Name	Type	Description
self	href	

igroups

Name	Type	Description
_links	self_link	
name	string	The name of the initiator group.
uuid	string	The unique identifier of the initiator group.

initiators

The initiators that are members of the initiator group.

Name	Type	Description
comment	string	A comment available for use by the administrator.
name	string	Name of initiator that is a member of the initiator group.

igroup

The initiator group to which the LUN is mapped.

Name	Type	Description
_links	_links	
comment	string	A comment available for use by the administrator. Valid in POST and PATCH.

Name	Type	Description
igroups	array[igroups]	The existing initiator groups that are members of the group. Optional in POST. This property is mutually exclusive with the *initiators* property during POST. This array contains only the direct children of the initiator group. If the member initiator groups have further nested initiator groups, those are reported in the `igroups` property of the child initiator group. Zero or more nested initiator groups can be supplied when the initiator group is created. The initiator group will act as if it contains the aggregation of all initiators in any nested initiator groups. After creation, nested initiator groups can be added or removed from the initiator group using the `/protocols/san/igroups/{igroup.uuid}/igroups` endpoint. See DELETE /protocols/san/igroups/{igroup.uuid}/igroups/{uuid} for more details.
initiators	array[initiators]	The initiators that are members of the group.
name	string	The name of the initiator group.
os_type	string	The host operating system of the initiator group. All initiators in the group should be hosts of the same operating system.

Name	Type	Description
protocol	string	The protocols supported by the initiator group. This restricts the type of initiators that can be added to the initiator group. Optional in POST; if not supplied, this defaults to *mixed*. The protocol of an initiator group cannot be changed after creation of the group.
uuid	string	The unique identifier of the initiator group.

lun_maps

A LUN map is an association between a LUN and an initiator group.

When a LUN is mapped to an initiator group, the initiator group's initiators are granted access to the LUN. The relationship between a LUN and an initiator group is many LUNs to many initiator groups.

Name	Type	Description
_links	_links	
igroup	igroup	The initiator group to which the LUN is mapped.
logical_unit_number	integer	The logical unit number assigned to the LUN when mapped to the specified initiator group. The number is used to identify the LUN to initiators in the initiator group when communicating through the Fibre Channel Protocol or iSCSI. Optional in POST; if no value is provided, ONTAP assigns the lowest available value. This property is not supported when the *provisioning_options.count* property is 2 or more. • Introduced in: 9.6 • readCreate: 1 • x-nullable: true

iops

The rate of I/O operations observed at the storage object.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

latency

The round trip latency in microseconds observed at the storage object.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

throughput

The rate of throughput bytes per second observed at the storage object.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.

Name	Type	Description
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

metric

Performance numbers, such as IOPS latency and throughput.

Name	Type	Description
_links	_links	
duration	string	The duration over which this sample is calculated. The time durations are represented in the ISO-8601 standard format. Samples can be calculated over the following durations:
iops	iops	The rate of I/O operations observed at the storage object.
latency	latency	The round trip latency in microseconds observed at the storage object.

Name	Type	Description
status	string	Errors associated with the sample. For example, if the aggregation of data over multiple nodes fails, then any partial errors might return "ok" on success or "error" on an internal uncategorized failure. Whenever a sample collection is missed but done at a later time, it is back filled to the previous 15 second timestamp and tagged with "backfilled_data". "Inconsistent_ delta_time" is encountered when the time between two collections is not the same for all nodes. Therefore, the aggregated value might be over or under inflated. "Negative_delta" is returned when an expected monotonically increasing value has decreased in value. "Inconsistent_old_data" is returned when one or more nodes do not have the latest data.
throughput	throughput	The rate of throughput bytes per second observed at the storage object.
timestamp	string	The timestamp of the performance data.

paths

The fully qualified LUN path names involved in the LUN movement.

Name	Type	Description
destination	string	The fully qualified path of the LUN movement destination composed of a "/vol" prefix, the volume name, the optional qtree name, and base name of the LUN.

Name	Type	Description
source	string	The fully qualified path of the LUN movement source composed of a "/vol" prefix, the volume name, the optional qtree name, and base name of the LUN.

error

Error information provided if the asynchronous LUN movement operation fails.

Name	Type	Description
arguments	array[error_arguments]	Message arguments
code	string	Error code
message	string	Error message

progress

Properties related to the progress of an active or recently completed LUN movement.

Name	Type	Description
elapsed	integer	The amount of time that has elapsed since the start of the LUN movement, in seconds.
failure	error	Error information provided if the asynchronous LUN movement operation fails.
percent_complete	integer	The percentage completed of the LUN movement.
state	string	The state of the LUN movement. Valid in PATCH when an LUN movement is active. Set to *paused* to pause a LUN movement. Set to *replicating* to resume a paused LUN movement.

Name	Type	Description
volume_snapshot_blocked	boolean	This property reports if volume snapshots are blocked by the LUN movement. This property can be polled to identify when volume snapshots can be resumed after beginning a LUN movement.

movement

This sub-object applies to LUN movement between volumes. A LUN can be moved to a new volume with a PATCH request that changes either the volume portion of property `name`, `location.volume.uuid`, or `location.volume.name`. If the volume is changed using more than one of these properties, the supplied properties used must refer to the same volume.

Moving a LUN between volumes is an asynchronous activity begun by a PATCH request. The data for the LUN is then asynchronously copied from the source volume to the destination volume. The time required to complete the move depends on the size of the LUN and the load on the cluster. The `movement` sub-object is populated while a LUN movement is in progress and for two (2) minutes following completion of a movement.

While the LUN is being moved, the status of the LUN movement operation can be obtained using a GET for the LUN that requests the `movement` properties. The LUN movement operation can be further modified using a PATCH on the properties on the `movement` sub-object.

There is an added computational cost to retrieving property values for `movement`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
max_throughput	integer	The maximum data throughput, in bytes per second, that should be utilized in support of the LUN movement. This property can be used to throttle a transfer and limit its impact on the performance of the source and destination nodes. The specified value will be rounded up to the nearest megabyte. If this property is not specified in a POST that begins a LUN movement, throttling is not applied to the data transfer. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. This property is valid only in a POST that begins a LUN movement or a PATCH when a LUN movement is already in process. • format: int64 • Introduced in: 9.6 • x-nullable: true
paths	paths	The fully qualified LUN path names involved in the LUN movement.
progress	progress	Properties related to the progress of an active or recently completed LUN movement.

qos_policy

The QoS policy for the volume provisioned to host the LUN. This property is only supported when the request provisions a new volume. This property is mutually exclusive with the LUN granular `qos_policy`. If no `qos_policy` is provided at LUN or volume granularity, a volume granular policy is set based on the `storage_service.name`, which defaults to the most performant service available.

Name	Type	Description
_links	self_link	

Name	Type	Description
name	string	The QoS policy group name. This is mutually exclusive with UUID and other QoS attributes during POST and PATCH.
uuid	string	The QoS policy group UUID. This is mutually exclusive with name and other QoS attributes during POST and PATCH.

snapshot_policy_reference

This is a reference to the snapshot policy.

Name	Type	Description
_links	_links	
name	string	
uuid	string	

storage_service

Determines the placement of the LUN based on the value specified. This property is only supported for regular and vvol LUNs. Valid in POST.

object_stores

tiering

The tiering placement and policy definitions for the volume provisioned to host the LUN. This property is only supported when the request provisions a new volume.

Name	Type	Description
object_stores	array[object_stores]	Object stores to use. Used for placement.

Name	Type	Description
policy	string	Policy that determines whether the user data blocks of a volume in a FabricPool will be tiered to the cloud store when they become cold. FabricPool combines flash (performance tier) with a cloud store into a single aggregate. Temperature of a volume block increases if it is accessed frequently and decreases when it is not. Valid in POST or PATCH. all ‐ Allows tiering of both snapshots and active file system user data to the cloud store as soon as possible by ignoring the temperature on the volume blocks. auto ‐ Allows tiering of both snapshot and active file system user data to the cloud store none ‐ Volume blocks are not be tiered to the cloud store. snapshot_only ‐ Allows tiering of only the volume snapshots not associated with the active file system. The default tiering policy is "snapshot-only" for a FlexVol volume and "none" for a FlexGroup volume. The default minimum cooling period for the "snapshot-only" tiering policy is 2 days and for the "auto" tiering policy it is 31 days.

provisioning_options

Options that are applied to the operation.

Name	Type	Description
storage_service	storage_service	Determines the placement of the LUN based on the value specified. This property is only supported for regular and vvol LUNs. Valid in POST.

qos_policy

The QoS policy for the LUN. Both traditional and adaptive QoS policies are supported. If both property `qos_policy.uuid` and `qos_policy.name` are specified in the same request, they must refer to the same QoS policy. To remove the QoS policy from a LUN, leaving it with no QoS policy, set the property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.

Name	Type	Description
_links	_links	
name	string	The name of the QoS policy. To remove the QoS policy from a LUN, leaving it with no QoS policy, set this property to an empty string ("") in a PATCH request. Valid in POST and PATCH.
uuid	string	The unique identifier of the QoS policy. Valid in POST and PATCH.

guarantee

Properties that request and report the space guarantee for the LUN.

Name	Type	Description
requested	boolean	The requested space reservation policy for the LUN. If *true*, a space reservation is requested for the LUN; if *false*, the LUN is thin provisioned. Guaranteeing a space reservation request for a LUN requires that the volume in which the LUN resides is also space reserved and that the fractional reserve for the volume is 100%. Valid in POST and PATCH. This property is caller settable as described above.

Name	Type	Description
reserved	boolean	Reports if the LUN is space guaranteed. If *true*, a space guarantee is requested and the containing volume and aggregate support the request. If *false*, a space guarantee is not requested or a space guarantee is requested and either the containing volume or aggregate do not support the request.

space

The storage space related properties of the LUN.

Name	Type	Description
efficiency_ratio	number	The storage efficiency ratio of the LUN without snapshots. (Logical Used / Used) This property is not available on the LUN object in the REST API and is not reported for GET requests. See the containing volume object for this information.
guarantee	guarantee	Properties that request and report the space guarantee for the LUN.
physical_used	integer	The number of bytes consumed on the disk by the LUN, excluding snapshots. This property is not available on the LUN object in the REST API and is not reported for GET requests. See the containing volume object for this information.
physical_used_by_snapshots	integer	The number of bytes consumed on the disk by the LUN's snapshots. This property is not available on the LUN object in the REST API and is not reported for GET requests. See the containing volume object for this information.

Name	Type	Description
scsi_thin_provisioning_support_enabled	boolean	To leverage the benefits of SCSI thin provisioning, it must be supported by your host. SCSI thin provisioning uses the Logical Block Provisioning feature as defined in the SCSI SBC-3 standard. Only hosts that support this standard can use SCSI thin provisioning in ONTAP. When you disable SCSI thin provisioning support in ONTAP, you turn off the following SCSI thin provisioning features: • Unmapping and reporting space usage for space reclamation • Reporting resource exhaustion errors The value of this property is not propagated to the destination when a LUN is cloned as a new LUN or copied; it is reset to false. The value of this property is maintained from the destination LUN when a LUN is overwritten as a clone. Valid in POST and PATCH. • Default value: 1 • Introduced in: 9.10 • x-nullable: true

Name	Type	Description
size	integer	The total provisioned size of the LUN. The LUN size can be increased but not decreased using the REST interface. The maximum and minimum sizes listed here are the absolute maximum and absolute minimum sizes, in bytes. The actual minimum and maximum sizes vary depending on the ONTAP version, ONTAP platform and the available space in the containing volume and aggregate. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. • example: 1073741824 • format: int64 • Max value: 140737488355328 • Min value: 4096 • Introduced in: 9.6 • x-nullable: true

Name	Type	Description
used	integer	The amount of space consumed by the main data stream of the LUN. This value is the total space consumed in the volume by the LUN, including filesystem overhead, but excluding prefix and suffix streams. Due to internal filesystem overhead and the many ways SAN filesystems and applications utilize blocks within a LUN, this value does not necessarily reflect actual consumption/availability from the perspective of the filesystem or application. Without specific knowledge of how the LUN blocks are utilized outside of ONTAP, this property should not be used as an indicator for an out-of-space condition. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. • format: int64 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true

iops_raw

The number of I/O operations observed at the storage object. This can be used along with delta time to calculate the rate of I/O operations per unit of time.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.

Name	Type	Description
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

latency_raw

The raw latency in microseconds observed at the storage object. This can be divided by the raw IOPS value to calculate the average latency per I/O operation.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

throughput_raw

Throughput bytes observed at the storage object. This can be used along with delta time to calculate the rate of throughput bytes per unit of time.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.

Name	Type	Description
write	integer	Performance metric for write I/O operations.

statistics

These are raw performance numbers, such as IOPS latency and throughput. These numbers are aggregated across all nodes in the cluster and increase with the uptime of the cluster.

Name	Type	Description
iops_raw	iops_raw	The number of I/O operations observed at the storage object. This can be used along with delta time to calculate the rate of I/O operations per unit of time.
latency_raw	latency_raw	The raw latency in microseconds observed at the storage object. This can be divided by the raw IOPS value to calculate the average latency per I/O operation.
status	string	Errors associated with the sample. For example, if the aggregation of data over multiple nodes fails, then any partial errors might return "ok" on success or "error" on an internal uncategorized failure. Whenever a sample collection is missed but done at a later time, it is back filled to the previous 15 second timestamp and tagged with "backfilled_data". "Inconsistent_ delta_time" is encountered when the time between two collections is not the same for all nodes. Therefore, the aggregated value might be over or under inflated. "Negative_delta" is returned when an expected monotonically increasing value has decreased in value. "Inconsistent_old_data" is returned when one or more nodes do not have the latest data.

Name	Type	Description
throughput_raw	throughput_raw	Throughput bytes observed at the storage object. This can be used along with delta time to calculate the rate of throughput bytes per unit of time.
timestamp	string	The timestamp of the performance data.

status

Status information about the LUN.

Name	Type	Description
container_state	string	The state of the volume and aggregate that contain the LUN. LUNs are only available when their containers are available.
mapped	boolean	Reports if the LUN is mapped to one or more initiator groups. There is an added computational cost to retrieving this property's value. It is not populated for a GET request unless it is explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
read_only	boolean	Reports if the LUN allows only read access.
state	string	The state of the LUN. Normal states for a LUN are *online* and *offline*. Other states indicate errors.

svm

The SVM in which the LUN is located.

Name	Type	Description
_links	_links	

Name	Type	Description
name	string	The name of the SVM. This field cannot be specified in a PATCH method.
uuid	string	The unique identifier of the SVM. This field cannot be specified in a PATCH method.

partner

The LUN partner that this LUN is bound to. If this LUN is a `vvol` class LUN, the partner is a `protocol_endpoint` class LUN.

Name	Type	Description
_links	_links	
name	string	The name of the partner LUN.
uuid	string	The unique identifier of the partner LUN.

bindings

A vVol binding with which the LUN is associated.

Name	Type	Description
_links	_links	

Name	Type	Description
id	integer	The ONTAP internal identifier assigned to the vVol binding. The bind identifier is unique amongst all class `vvol` LUNs bound to the same class `protocol_endpoint` LUN. This property was included in early releases of the REST API for vVols and is maintained for backward compatibility. See the `secondary_id` property, which replaces `id`. • example: 1 • readOnly: 1 • x-ntap-deprecated: 9.13.1 • Introduced in: 9.10 • x-nullable: true
partner	partner	The LUN partner that this LUN is bound to. If this LUN is a `vvol` class LUN, the partner is a `protocol_endpoint` class LUN.
secondary_id	string	The identifier assigned to the vVol binding, known as the secondary LUN ID. The identifier is unique amongst all class `vvol` LUNs bound to the same class `protocol_endpoint` LUN. The format for a secondary LUN ID is 16 hexadecimal digits (zero-filled) followed by a lower case "h".

vvol

A VMware virtual volume (vVol) binding is an association between a LUN of class `protocol_endpoint` and a LUN of class `vvol`. Class `protocol_endpoint` LUNs are mapped to igroups and granted access using the same configuration as class `regular` LUNs. When a class `vvol` LUN is bound to a mapped class `protocol_endpoint` LUN, VMware can access the class `vvol` LUN through the class `protocol_endpoint` LUN mapping.

See DELETE /protocols/san/vvol-bindings to learn more about deleting vVol bindings.

There is an added computational cost to retrieving property values for `vvol`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
bindings	array[bindings]	Bindings between the LUN, which must be of class `protocol_endpoint` or `vvol`, and LUNs of the opposite class. A class `vvol` LUN must be bound to a class `protocol_endpoint` LUN in order to be accessed. Class `protocol_endpoint` and `vvol` LUNs allow many-to-many bindings. A LUN of one class is allowed to be bound to zero or more LUNs of the opposite class. The binding between any two specific LUNs is reference counted. When a binding is created that already exists, the binding count is incremented. When a binding is deleted, the binding count is decremented, but the LUNs remain bound if the resultant reference count is greater than zero. When the binding count reaches zero, the binding is destroyed. The bindings array contains LUNs of the opposite class of the containing LUN object. There is an added computational cost to retrieving property values for `vvol.bindings`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
is_bound	boolean	Reports if the LUN is part of a VMware virtual volume (vVol) bind relationship. This is `true` if the LUN is of class `protocol_endpoint` or `vvol` and has one or more bindings to a LUN of the opposite class. This is false if the LUN is of class `regular` or unbound.

lun

A LUN is the logical representation of storage in a storage area network (SAN).

A LUN must be mapped to an initiator group to grant access to the initiator group's initiators (client hosts). Initiators can then access the LUN and perform I/O over a Fibre Channel (FC) fabric using the FC Protocol or a TCP/IP network using iSCSI.

See the LUN object model to learn more about each of the properties supported by the LUN REST API.

A LUN is located within a volume. Optionally, it can be located within a qtree in a volume.

LUN names are paths of the form "/vol/<volume>[/<qtree>]/<lun>" where the qtree name is optional.

A LUN can be created to a specified size using thin or thick provisioning. A LUN can then be renamed, resized, cloned, moved to a different volume and copied. LUNs support the assignment of a QoS policy for performance management or a QoS policy can be assigned to a volume containing one or more LUNs.

Name	Type	Description
_links	_links	

Name	Type	Description
attributes	array[attributes]	An array of name/value pairs optionally stored with the LUN. Attributes are available to callers to persist small amounts of application-specific metadata. They are in no way interpreted by ONTAP. Attribute names and values must be at least one byte and no more than 4091 bytes in length. The sum of the name and value lengths must be no more than 4092 bytes. Valid in POST except when creating a LUN clone. A cloned can already have attributes from its source. You can add, modify, and delete the attributes of a LUN clone in separate requests after creation of the LUN. Attributes can be added/modified/removed for an existing LUN using the /api/storage/luns/{lun.uuid}/attributes endpoint. For further information, see DOC /storage/luns/{lun.uuid}/attributes . There is an added computational cost to retrieving property values for `attributes`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more. • Introduced in: 9.10 • readCreate: 1

Name	Type	Description
auto_delete	boolean	This property marks the LUN for auto deletion when the volume containing the LUN runs out of space. This is most commonly set on LUN clones. When set to *true*, the LUN becomes eligible for automatic deletion when the volume runs out of space. Auto deletion only occurs when the volume containing the LUN is also configured for auto deletion and free space in the volume decreases below a particular threshold. This property is optional in POST and PATCH. The default value for a new LUN is *false*. There is an added computational cost to retrieving this property's value. It is not populated for a GET request unless it is explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
class	string	The class of LUN. Optional in POST.
comment	string	A configurable comment available for use by the administrator. Valid in POST and PATCH.
consistency_group	consistency_group	The LUN's consistency group. This property is populated for LUNs that are members of a consistency group. If the LUN is a member of a child consistency group, the parent consistency group is reported. A LUN's consistency group is the consistency group of its containing volume.

Name	Type	Description
copy	copy	This sub-object applies to LUN copy operations. A LUN can be copied with a POST request that supplies `copy.source` properties. Copying a LUN is an asynchronous activity begun by a POST request that specifies the source of the copy in the `copy.source` properties. The data for the LUN is then asynchronously copied from the source to the destination. The time required to complete the copy depends on the size of the LUN and the load on the cluster. The `copy` sub-object is populated while a LUN copy is in progress and for two (2) minutes following completion of a copy. While LUNs are being copied, the status of the LUN copy operations can be obtained using a GET of the source or destination LUN that requests the `copy` properties. If the LUN is the source LUN for one or more copy operations, the `copy.destinations` array is populated in GET. If the containing LUN is the destination LUN for a copy operation, the `copy.source` sub-object is populated in GET. The LUN copy operation can be further modified using a PATCH on the properties on the `copy.source` sub-object of the copy destination LUN. There is an added computational cost to retrieving property values for `copy`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
create_time	string	The time the LUN was created.

Name	Type	Description
enabled	boolean	The enabled state of the LUN. LUNs can be disabled to prevent access to the LUN. Certain error conditions also cause the LUN to become disabled. If the LUN is disabled, check the `status.state` property to determine if the LUN is administratively disabled (*offline*) or has become disabled as a result of an error. A LUN in an error condition can be brought online by setting the `enabled` property to *true* or brought administratively offline by setting the `enabled` property to *false*. Upon creation, a LUN is enabled by default. Valid in PATCH.
location	location	The location of the LUN within the ONTAP cluster. LUNs support rename and move between volumes. Valid in POST and PATCH. • Introduced in: 9.6
lun_maps	array[lun_maps]	The LUN maps with which the LUN is associated. There is an added computational cost to retrieving property values for `lun_maps`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more. These properties are supported for GET only.
metric	metric	Performance numbers, such as IOPS latency and throughput.

Name	Type	Description
movement	movement	This sub-object applies to LUN movement between volumes. A LUN can be moved to a new volume with a PATCH request that changes either the volume portion of property `name`, `location.volume.uuid`, or `location.volume.name`. If the volume is changed using more than one of these properties, the supplied properties used must refer to the same volume. Moving a LUN between volumes is an asynchronous activity begun by a PATCH request. The data for the LUN is then asynchronously copied from the source volume to the destination volume. The time required to complete the move depends on the size of the LUN and the load on the cluster. The `movement` sub-object is populated while a LUN movement is in progress and for two (2) minutes following completion of a movement. While the LUN is being moved, the status of the LUN movement operation can be obtained using a GET for the LUN that requests the `movement` properties. The LUN movement operation can be further modified using a PATCH on the properties on the `movement` sub-object. There is an added computational cost to retrieving property values for `movement`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
name	string	The name of the LUN. Valid in POST and PATCH. A LUN is located within a volume. Optionally, it can be located within a qtree in a volume. LUN names are paths of the form "/vol/<volume>[/<qtree>]/<lun>" where the qtree name is optional. A PATCH that modifies the qtree and/or base name portion of the LUN path is considered a rename operation. A PATCH that modifies the volume portion of the LUN path begins an asynchronous LUN movement operation.
os_type	string	The operating system type of the LUN. Required in POST when creating a LUN that is not a clone of another. Disallowed in POST when creating a LUN clone.
qos_policy	qos_policy	The QoS policy for the LUN. Both traditional and adaptive QoS policies are supported. If both property `qos_policy.uuid` and `qos_policy.name` are specified in the same request, they must refer to the same QoS policy. To remove the QoS policy from a LUN, leaving it with no QoS policy, set the property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.

Name	Type	Description
serial_number	string	The LUN serial number. The serial number is generated by ONTAP when the LUN is created. • maxLength: 12 • minLength: 12 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true
serial_number_hex	string	The LUN serial number encoded in hexadecimal format. The serial number is generated by ONTAP when the LUN is created. • readOnly: 1 • Introduced in: 9.17 • x-nullable: true
space	space	The storage space related properties of the LUN.
statistics	statistics	These are raw performance numbers, such as IOPS latency and throughput. These numbers are aggregated across all nodes in the cluster and increase with the uptime of the cluster.
status	status	Status information about the LUN.
svm	svm	The SVM in which the LUN is located.
uuid	string	The unique identifier of the LUN. The UUID is generated by ONTAP when the LUN is created. • example: 1cd8a442-86d1-11e0-ae1c-123478563412 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true

Name	Type	Description
vvol	vvol	A VMware virtual volume (vVol) binding is an association between a LUN of class `protocol_endpoint` and a LUN of class `vvol`. Class `protocol_endpoint` LUNs are mapped to igroups and granted access using the same configuration as class `regular` LUNs. When a class `vvol` LUN is bound to a mapped class `protocol_endpoint` LUN, VMware can access the class `vvol` LUN through the class `protocol_endpoint` LUN mapping. See DELETE /protocols/san/vvol-bindings to learn more about deleting vVol bindings. There is an added computational cost to retrieving property values for `vvol`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

returned_error

Name	Type	Description
arguments	array[error_arguments]	Message arguments
code	string	Error code
message	string	Error message
target	string	The target parameter that caused the error.

Create a LUN

POST `/storage/luns`

Introduced In: 9.6

Creates a LUN.

Required properties

- `svm.uuid` or `svm.name` - Existing SVM in which to create the LUN.
- `name`, `location.volume.name` or `location.volume.uuid` - Existing volume in which to create the LUN.
- `name` or `location.logical_unit` - Base name of the LUN.
- `os_type` - Operating system from which the LUN will be accessed. Required when creating a non-clone LUN and disallowed when creating a clone of an existing LUN. A clone's `os_type` is taken from the source LUN.
- `space.size` - Size of the LUN. Required when creating a non-clone LUN and disallowed when creating a clone of an existing LUN. A clone's size is taken from the source LUN.

Recommended optional properties

- `qos_policy.name` or `qos_policy.uuid` - Existing traditional or adaptive QoS policy to be applied to the LUN. All LUNs should be managed by a QoS policy at the volume or LUN level.

Default property values

If not specified in POST, the follow default property values are assigned.

- `auto_delete` - *false*

Related ONTAP commands

- `lun create`
- `lun convert-from-namespace`
- `lun copy start`
- `volume file clone autodelete`
- `volume file clone create`

POST is asynchronous when creating a new LUN. It is synchronous when converting a namespace to a LUN via the `convert` property.

Learn more

- DOC /storage/luns

Parameters

Name	Type	In	Required	Description
return_timeout	integer	query	False	The number of seconds to allow the call to execute before returning. When doing a POST, PATCH, or DELETE operation on a single record, the default is 0 seconds. This means that if an asynchronous operation is started, the server immediately returns HTTP code 202 (Accepted) along with a link to the job. If a non-zero value is specified for POST, PATCH, or DELETE operations, ONTAP waits that length of time to see if the job completes so it can return something other than 202. • Default value: 0 • Max value: 120 • Min value: 0
return_records	boolean	query	False	The default is false. If set to true, the records are returned. • Default value:

Request Body

Name	Type	Description
attributes	array[attributes]	An array of name/value pairs optionally stored with the LUN. Attributes are available to callers to persist small amounts of application-specific metadata. They are in no way interpreted by ONTAP. Attribute names and values must be at least one byte and no more than 4091 bytes in length. The sum of the name and value lengths must be no more than 4092 bytes. Valid in POST except when creating a LUN clone. A cloned can already have attributes from its source. You can add, modify, and delete the attributes of a LUN clone in separate requests after creation of the LUN. Attributes can be added/modified/removed for an existing LUN using the /api/storage/luns/{lun.uuid}/attributes endpoint. For further information, see DOC /storage/luns/{lun.uuid}/attributes . There is an added computational cost to retrieving property values for `attributes`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more. • Introduced in: 9.10 • readCreate: 1

Name	Type	Description
auto_delete	boolean	This property marks the LUN for auto deletion when the volume containing the LUN runs out of space. This is most commonly set on LUN clones. When set to *true*, the LUN becomes eligible for automatic deletion when the volume runs out of space. Auto deletion only occurs when the volume containing the LUN is also configured for auto deletion and free space in the volume decreases below a particular threshold. This property is optional in POST and PATCH. The default value for a new LUN is *false*. There is an added computational cost to retrieving this property's value. It is not populated for a GET request unless it is explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
class	string	The class of LUN. Optional in POST.

Name	Type	Description
clone	clone	This sub-object is used in POST to create a new LUN as a clone of an existing LUN, or PATCH to overwrite an existing LUN as a clone of another. Setting a property in this sub-object indicates that a LUN clone is desired. Consider the following other properties when cloning a LUN: `auto_delete`, `qos_policy`, `space.guarantee.requested` and `space.scsi_thin_provisioning_support_enabled`. When used in a PATCH, the patched LUN's data is over-written as a clone of the source and the following properties are preserved from the patched LUN unless otherwise specified as part of the PATCH: `class`, `auto_delete`, `lun_maps`, `serial_number`, `status.state`, and `uuid`. Persistent reservations for the patched LUN are also preserved.
comment	string	A configurable comment available for use by the administrator. Valid in POST and PATCH.
consistency_group	consistency_group	The LUN's consistency group. This property is populated for LUNs that are members of a consistency group. If the LUN is a member of a child consistency group, the parent consistency group is reported. A LUN's consistency group is the consistency group of its containing volume.
convert	convert	This sub-object is used in POST to convert a valid in-place NVMe namespace to a LUN. Setting a property in this sub-object indicates that a conversion from the specified NVMe namespace to LUN is desired.

Name	Type	Description
copy	copy	This sub-object applies to LUN copy operations. A LUN can be copied with a POST request that supplies `copy.source` properties. Copying a LUN is an asynchronous activity begun by a POST request that specifies the source of the copy in the `copy.source` properties. The data for the LUN is then asynchronously copied from the source to the destination. The time required to complete the copy depends on the size of the LUN and the load on the cluster. The `copy` sub-object is populated while a LUN copy is in progress and for two (2) minutes following completion of a copy. While LUNs are being copied, the status of the LUN copy operations can be obtained using a GET of the source or destination LUN that requests the `copy` properties. If the LUN is the source LUN for one or more copy operations, the `copy.destinations` array is populated in GET. If the containing LUN is the destination LUN for a copy operation, the `copy.source` sub-object is populated in GET. The LUN copy operation can be further modified using a PATCH on the properties on the `copy.source` sub-object of the copy destination LUN. There is an added computational cost to retrieving property values for `copy`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
create_time	string	The time the LUN was created.

Name	Type	Description
location	location	The location of the LUN within the ONTAP cluster. LUNs support rename and move between volumes. Valid in POST and PATCH. • Introduced in: 9.6
lun_maps	array[lun_maps]	The LUN maps with which the LUN is associated. There is an added computational cost to retrieving property values for `lun_maps`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more. These properties are supported for GET only.

Name	Type	Description
movement	movement	This sub-object applies to LUN movement between volumes. A LUN can be moved to a new volume with a PATCH request that changes either the volume portion of property `name`, `location.volume.uuid`, or `location.volume.name`. If the volume is changed using more than one of these properties, the supplied properties used must refer to the same volume. Moving a LUN between volumes is an asynchronous activity begun by a PATCH request. The data for the LUN is then asynchronously copied from the source volume to the destination volume. The time required to complete the move depends on the size of the LUN and the load on the cluster. The `movement` sub-object is populated while a LUN movement is in progress and for two (2) minutes following completion of a movement. While the LUN is being moved, the status of the LUN movement operation can be obtained using a GET for the LUN that requests the `movement` properties. The LUN movement operation can be further modified using a PATCH on the properties on the `movement` sub-object. There is an added computational cost to retrieving property values for `movement`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
name	string	The name of the LUN. Valid in POST and PATCH. A LUN is located within a volume. Optionally, it can be located within a qtree in a volume. LUN names are paths of the form "/vol/<volume>[/<qtree>]/<lun>" where the qtree name is optional. A PATCH that modifies the qtree and/or base name portion of the LUN path is considered a rename operation. A PATCH that modifies the volume portion of the LUN path begins an asynchronous LUN movement operation.
os_type	string	The operating system type of the LUN. Required in POST when creating a LUN that is not a clone of another. Disallowed in POST when creating a LUN clone.
provisioning_options	provisioning_options	Options that are applied to the operation.
qos_policy	qos_policy	The QoS policy for the LUN. Both traditional and adaptive QoS policies are supported. If both property `qos_policy.uuid` and `qos_policy.name` are specified in the same request, they must refer to the same QoS policy. To remove the QoS policy from a LUN, leaving it with no QoS policy, set the property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.

Name	Type	Description
serial_number	string	The LUN serial number. The serial number is generated by ONTAP when the LUN is created. • maxLength: 12 • minLength: 12 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true
serial_number_hex	string	The LUN serial number encoded in hexadecimal format. The serial number is generated by ONTAP when the LUN is created. • readOnly: 1 • Introduced in: 9.17 • x-nullable: true
space	space	The storage space related properties of the LUN.
status	status	Status information about the LUN.
svm	svm	The SVM in which the LUN is located.
uuid	string	The unique identifier of the LUN. The UUID is generated by ONTAP when the LUN is created. • example: 1cd8a442-86d1-11e0-ae1c-123478563412 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true

Name	Type	Description
vvol	vvol	A VMware virtual volume (vVol) binding is an association between a LUN of class `protocol_endpoint` and a LUN of class `vvol`. Class `protocol_endpoint` LUNs are mapped to igroups and granted access using the same configuration as class `regular` LUNs. When a class `vvol` LUN is bound to a mapped class `protocol_endpoint` LUN, VMware can access the class `vvol` LUN through the class `protocol_endpoint` LUN mapping. See DELETE /protocols/san/vvol-bindings to learn more about deleting vVol bindings. There is an added computational cost to retrieving property values for `vvol`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Example request


```
{
  "attributes": [
    {
      "name": "name1",
      "value": "value1"
    }
  ],
  "class": "string",
  "clone": {
    "source": {
      "name": "/vol/volume1/lun1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    }
  },
  "comment": "string",
  "consistency_group": {
    "name": "cg1",
    "uuid": "4abc2317-4332-9d37-93a0-20bd29c22df0"
  },
  "convert": {
    "namespace": {
      "name": "/vol/volume1/namespace1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    }
  },
  "copy": {
    "destinations": [
      {
        "max_throughput": 0,
        "name": "/vol/vol1/lun1",
        "peer": {
          "name": "peer1",
          "uuid": "4204cf77-4c82-9bdb-5644-b5a841c097a9"
        },
        "progress": {
          "elapsed": 0,
          "percent_complete": 0,
          "state": "string"
        },
        "uuid": "1bc327d5-4654-5284-a116-f182282240b4"
      }
    ],
    "source": {
      "name": "/vol/vol2/lun1",
```

```
      "peer": {
        "name": "peer1",
        "uuid": "4204cf77-4c82-9bdb-5644-b5a841c097a9"
      },
      "uuid": "03c05019-40d9-3945-c767-dca4c3be5e90"
    }
  },
  "create_time": "2018-06-04 15:00:00 -0400",
  "enabled": true,
  "location": {
    "logical_unit": "lun1",
    "node": {
      "name": "node1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    },
    "qtree": {
      "id": 1,
      "name": "qt1"
    },
    "volume": {
      "name": "volume1",
      "uuid": "028baa66-41bd-11e9-81d5-00a0986138f7"
    }
  },
  "lun_maps": [
    {
      "igroup": {
        "comment": "string",
        "igroups": [
          {
            "name": "igroup1",
            "uuid": "4ea7a442-86d1-11e0-ae1c-123478563412"
          }
        ],
        "initiators": [
          {
            "comment": "my comment",
            "name": "iqn.1998-01.com.corp.iscsi:name1"
          }
        ],
        "name": "igroup1",
        "os_type": "string",
        "protocol": "string",
        "uuid": "4ea7a442-86d1-11e0-ae1c-123478563412"
      }
    }
```

```
  ],
  "movement": {
    "paths": {
      "destination": "/vol/vol1/lun1",
      "source": "/vol/vol2/lun2"
    }
  },
  "name": "/vol/volume1/qtree1/lun1",
  "os_type": "string",
  "provisioning_options": {
    "qos_policy": {
      "name": "performance",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    },
    "snapshot_policy": {
      "name": "default",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    },
    "storage_service": {
      "name": "string"
    },
    "tiering": {
      "control": "string",
      "object_stores": [
        {
          "name": "string"
        }
      ],
      "policy": "string"
    }
  },
  "qos_policy": {
    "name": "qos1",
    "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
  },
  "serial_number": "string",
  "serial_number_hex": "string",
  "space": {
    "efficiency_ratio": 2.5,
    "physical_used": 1073741824,
    "physical_used_by_snapshots": 1073741824,
    "size": 1073741824,
    "used": 0
  },
  "status": {
    "container_state": "string",
```

```
    "state": "online"
  },
  "svm": {
    "name": "svm1",
    "uuid": "02c9e252-41be-11e9-81d5-00a0986138f7"
  },
  "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412",
  "vvol": {
    "bindings": [
      {
        "id": 1,
        "partner": {
          "name": "/vol/vol1/lun1",
          "uuid": "4ea7a442-86d1-11e0-ae1c-123478563412"
        },
        "secondary_id": "0000D20000010000h"
      }
    ]
  }
}
```

Response

```
Status: 202, Accepted
```

Name	Type	Description
job	job_link	

Example response

```
{
  "job": {
    "uuid": "string"
  }
}
```

Headers

Name	Description	Type
Location	Useful for tracking the resource location	string

Response

```
Status: 201, Created
```

Error

```
Status: Default
```

ONTAP Error Response Codes

Error Code	Description
917927	The specified volume was not found.
918236	The specified `location.volume.uuid` and `location.volume.name` do not refer to the same volume.
1260121	Cloning a LUN to a volume different than the source volume is not supported.
1260136	The specified destination for a clone operation already exists as a LUN, namespace, or file.
2621462	The specified SVM does not exist.
2621706	The specified `svm.uuid` and `svm.name` do not refer to the same SVM.
2621707	No SVM was specified. Either `svm.name` or `svm.uuid` must be supplied.
5242927	The specified qtree was not found.
5242950	The specified `location.qtree.id` and `location.qtree.name` do not refer to the same qtree.
5374121	A LUN name can only contain characters A-Z, a-z, 0-9, "-", ".", "_", "{" and "}".
5374123	A negative size was provided for the LUN.
5374124	The specified size is too small for the LUN.
5374125	The specified size is too large for the LUN.
5374127	The specified LUN name is invalid.
5374129	LUNs cannot be created on a load sharing mirror volume.
5374130	An invalid size value was provided.
5374237	LUNs cannot be created on an SVM root volume.
5374238	LUNs cannot be created in snapshots.

Error Code	Description
5374241	A size value with invalid units was provided.
5374242	A LUN or NVMe namespace already exists at the specified path.
5374352	An invalid name was provided for the LUN.
5374614	The specified storage availability zone does not exist.
5374707	Creating a LUN in the specific volume is not allowed because the volume is reserved for an application.
5374858	The volume specified by `name` is not the same as that specified by `location.volume`.
5374859	No volume was specified for the LUN.
5374860	The qtree specified by `name` is not the same as that specified by `location.qtree`.
5374861	The LUN base name specified by `name` is not the same as that specified by `location.logical_unit`.
5374862	No LUN path base name was provided for the LUN.
5374863	An error occurred after successfully creating the LUN. Some properties were not set.
5374874	The specified `clone.source.uuid` and `clone.source.name` do not refer to the same LUN.
5374875	The specified `clone.source` was not found.
5374876	The specified `clone.source` was not found.
5374883	The property cannot be specified when creating a LUN clone. The `target` property of the error object identifies the property.
5374884	A property that is required when creating a new LUN that is not a LUN clone or LUN copy was not supplied. The `target` property of the error object identifies the property.
5374886	An error occurred after successfully creating the LUN preventing the retrieval of its properties.
5374899	The `clone.source.uuid` property is not supported when specifying a source LUN from a snapshot.
5374928	An incomplete attribute name/value pair was supplied.
5374929	The combined sizes of an attribute name and value are too large.
5374932	A name for an attribute was duplicated.

Error Code	Description
5374942	The property cannot be specified at the same time when creating a LUN as a clone. The `target` property of the error object identifies the other property given with clone.
5374943	The property cannot be specified at the same time when creating a LUN as a copy. The `target` property of the error object identifies the other property given with copy.
5374944	The property cannot be specified when converting an NVMe namespace into a LUN. The `target` property of the error object identifies the property.
5375054	The source LUN is required when requesting an SVM copy operation.
5375059	An unsuitable QoS policy was specified.
5375061	The specified `location.storage_availability_zone.uuid` and `location.storage_availability_zone.name` do not refer to the same storage availability zone.
5376461	The specified LUN name is invalid.
5376462	The specified LUN name is too long.
5376463	The snapshot portion of the specified LUN name is too long.
5376469	The property cannot be set during the LUN create operation on this platform.
5376515	The specified storage availability zone is not configured for provisioning on the specified SVM.
5440509	No suitable storage can be found for the specified requirements.
5440688	A LUN cannot be created with the same name as an existing LUN.
5702832	A LUN or namespace with the same name is already being created. LUN and namespace names must be unique within an SVM.
7018877	Maximum combined total (50) of file and LUN copy and move operations reached. When one or more of the operations has completed, try the command again.
13565952	The LUN clone request failed.
26345672	An SVM peer relationship was not found with the provided 'copy.source.peer.uuid'.

Error Code	Description
26345673	An SVM peer relationship between the SVM provided in 'copy.source.peer.name' and the SVM provided in either 'svm.name' or 'svm.uuid' was not found.
26345674	The SVM peer relationship provided in 'copy.source.peer.uuid' does not match a peer relationship between the local SVM provided in either 'svm.uuid' or 'svm.name' and the peer SVM provided in 'copy.source.peer.name'.
26345675	The destination SVM provided via 'svm.name' or 'svm.uuid' does not match the local SVM for the provided SVM peer relationship UUID in 'copy.source.peer.uuid'.
72089755	NVMe namespace with a block size of 4096 bytes cannot be converted to a LUN.
72089756	Namespace is currently mapped to subsystem.
72089757	NVMe namespace in a snapshot cannot be converted to a LUN.

Also see the table of common errors in the Response body overview section of this documentation.

Definitions

See Definitions

href

Name	Type	Description
href	string	

_links

attributes

A name/value pair optionally stored with the LUN. Attributes are available to callers to persist small amounts of application-specific metadata. They are in no way interpreted by ONTAP.

Attribute names and values must be at least one byte and no more than 4091 bytes in length. The sum of the name and value lengths must be no more than 4092 bytes.

Optional in POST.

Name	Type	Description
name	string	The attribute name.
value	string	The attribute value.

source

The source LUN for a LUN clone operation. This can be specified using property `clone.source.uuid` or `clone.source.name`. If both properties are supplied, they must refer to the same LUN.

Valid in POST to create a new LUN as a clone of the source.

Valid in PATCH to overwrite an existing LUN's data as a clone of another.

Name	Type	Description
name	string	The name of the clone source LUN. A LUN is located within a volume. Optionally, it can be located within a qtree in a volume. LUN names are paths of the form "/vol/<volume>[/<qtree>]/<namespace>" where the qtree name is optional. Valid in POST and PATCH.

Name	Type	Description
uuid	string	The unique identifier of the clone source LUN. Valid in POST and PATCH.

clone

This sub-object is used in POST to create a new LUN as a clone of an existing LUN, or PATCH to overwrite an existing LUN as a clone of another. Setting a property in this sub-object indicates that a LUN clone is desired. Consider the following other properties when cloning a LUN: `auto_delete`, `qos_policy`, `space.guarantee.requested` and `space.scsi_thin_provisioning_support_enabled`.

When used in a PATCH, the patched LUN's data is over-written as a clone of the source and the following properties are preserved from the patched LUN unless otherwise specified as part of the PATCH: `class`, `auto_delete`, `lun_maps`, `serial_number`, `status.state`, and `uuid`.

Persistent reservations for the patched LUN are also preserved.

Name	Type	Description
source	source	The source LUN for a LUN clone operation. This can be specified using property `clone.source.uuid` or `clone.source.name`. If both properties are supplied, they must refer to the same LUN. Valid in POST to create a new LUN as a clone of the source. Valid in PATCH to overwrite an existing LUN's data as a clone of another.

consistency_group

The LUN's consistency group. This property is populated for LUNs that are members of a consistency group. If the LUN is a member of a child consistency group, the parent consistency group is reported. A LUN's consistency group is the consistency group of its containing volume.

Name	Type	Description
name	string	The name of the consistency group.
uuid	string	The unique identifier of the consistency group.

namespace

The source namespace for convert operation. This can be specified using property `convert.namespace.uuid` or `convert.namespace.name`. If both properties are supplied, they must refer to the same NVMe namespace.

Valid in POST. A convert request from NVMe namespace to LUN cannot be combined with setting any other LUN properties. All other properties of the converted LUN come from the source NVMe namespace.

Name	Type	Description
name	string	The name of the source NVMe namespace. Valid in POST. An NVMe namespace is located within a volume. Optionally, it can be located within a qtree in a volume. NVMe namespace names are paths of the form "/vol/<volume>[/<qtree>]/<NVMe namespace>" where the qtree name is optional.
uuid	string	The unique identifier of the source NVMe namespace. Valid in POST.

convert

This sub-object is used in POST to convert a valid in-place NVMe namespace to a LUN. Setting a property in this sub-object indicates that a conversion from the specified NVMe namespace to LUN is desired.

Name	Type	Description
namespace	namespace	The source namespace for convert operation. This can be specified using property `convert.namespace.uuid` or `convert.namespace.name`. If both properties are supplied, they must refer to the same NVMe namespace. Valid in POST. A convert request from NVMe namespace to LUN cannot be combined with setting any other LUN properties. All other properties of the converted LUN come from the source NVMe namespace.

peer

The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM.

This is only populated by GET when the LUN copy is inter-SVM.

Name	Type	Description
name	string	The local name of the peer SVM. This name is unique among all local and peer SVMs.
uuid	string	The unique identifier of the SVM peer relationship. This is the UUID of the relationship, not the UUID of the peer SVM itself.

error_arguments

Name	Type	Description
code	string	Argument code
message	string	Message argument

error

Error information provided if the asynchronous LUN copy operation fails.

Name	Type	Description
arguments	array[error_arguments]	Message arguments
code	string	Error code
message	string	Error message

progress

Properties related to the progress of an active or recently completed LUN copy.

Name	Type	Description
elapsed	integer	The amount of time that has elapsed since the start of the LUN copy, in seconds.

Name	Type	Description
percent_complete	integer	The percentage completed of the LUN copy.
state	string	The state of the LUN copy.
volume_snapshot_blocked	boolean	This property reports if volume snapshots are blocked by the LUN copy. This property can be polled to identify when volume snapshots can be resumed after beginning a LUN copy.

destinations

A LUN copy operation in which the containing LUN is the source of the copy.

Name	Type	Description
max_throughput	integer	The maximum data throughput, in bytes per second, that should be utilized in support of the LUN copy. See property `copy.source.max_throughput` for further details.
name	string	The fully qualified path of the LUN copy destination composed of a "/vol" prefix, the volume name, the optional qtree name, and base name of the LUN.
peer	peer	The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM. This is only populated by GET when the LUN copy is inter-SVM.
progress	progress	Properties related to the progress of an active or recently completed LUN copy.
uuid	string	The unique identifier of the LUN copy destination.

peer

The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM.

Set this in POST to specify the source SVM for an inter-SVM LUN copy. Only populated by GET when the LUN copy is inter-SVM.

Name	Type	Description
name	string	The local name of the peer SVM. This name is unique among all local and peer SVMs.
uuid	string	The unique identifier of the SVM peer relationship. This is the UUID of the relationship, not the UUID of the peer SVM itself.

progress

Properties related to the progress of an active or recently completed LUN copy.

Name	Type	Description
elapsed	integer	The amount of time that has elapsed since the start of the LUN copy, in seconds.
percent_complete	integer	The percentage completed of the LUN copy.
volume_snapshot_blocked	boolean	This property reports if volume snapshots are blocked by the LUN copy. This property can be polled to identify when volume snapshots can be resumed after beginning a LUN copy.

source

The source LUN of a LUN copy operation in which the containing LUN is the destination of the copy.

Valid in POST except when creating a LUN clone. A LUN copy request cannot be combined with setting any other LUN properties except the destination location. All other properties of the destination LUN come from the source LUN.

Name	Type	Description
max_throughput	integer	The maximum data throughput, in bytes per second, that should be utilized in support of the LUN copy. This property can be used to throttle a transfer and limit its impact on the performance of the source and destination nodes. The specified value will be rounded up to the nearest megabyte. If this property is not specified in a POST that begins a LUN copy, throttling is not applied to the data transfer. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. Valid only in a POST that begins a LUN copy or a PATCH when a LUN copy is already in process. • format: int64 • Introduced in: 9.10 • x-nullable: true
name	string	The fully qualified path of the LUN copy source composed of a "/vol" prefix, the volume name, the optional qtree name, and base name of the LUN. Set this property in POST to specify the source for a LUN copy operation.

Name	Type	Description
peer	peer	The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM. Set this in POST to specify the source SVM for an inter-SVM LUN copy. Only populated by GET when the LUN copy is inter-SVM.
uuid	string	The unique identifier of the LUN copy source. Set this property in POST to specify the source for a LUN copy operation.

copy

This sub-object applies to LUN copy operations. A LUN can be copied with a POST request that supplies `copy.source` properties.

Copying a LUN is an asynchronous activity begun by a POST request that specifies the source of the copy in the `copy.source` properties. The data for the LUN is then asynchronously copied from the source to the destination. The time required to complete the copy depends on the size of the LUN and the load on the cluster. The `copy` sub-object is populated while a LUN copy is in progress and for two (2) minutes following completion of a copy.

While LUNs are being copied, the status of the LUN copy operations can be obtained using a GET of the source or destination LUN that requests the `copy` properties. If the LUN is the source LUN for one or more copy operations, the `copy.destinations` array is populated in GET. If the containing LUN is the destination LUN for a copy operation, the `copy.source` sub-object is populated in GET. The LUN copy operation can be further modified using a PATCH on the properties on the `copy.source` sub-object of the copy destination LUN.

There is an added computational cost to retrieving property values for `copy`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
destinations	array[destinations]	An array of destination LUNs of LUN copy operations in which the containing LUN is the source of the copy.

Name	Type	Description
source	source	The source LUN of a LUN copy operation in which the containing LUN is the destination of the copy. Valid in POST except when creating a LUN clone. A LUN copy request cannot be combined with setting any other LUN properties except the destination location. All other properties of the destination LUN come from the source LUN.

node

The cluster node that hosts the LUN.

Name	Type	Description
name	string	
uuid	string	

qtree

The qtree in which the LUN is optionally located. Valid in POST and PATCH.

If properties `name` and `location.qtree.name` and/or `location.qtree.id` are specified in the same request, they must refer to the same qtree.

A PATCH that modifies the qtree of the LUN is considered a rename operation.

Name	Type	Description
id	integer	The identifier for the qtree, unique within the qtree's volume.
name	string	The name of the qtree.

volume

The volume in which the LUN is located. Valid in POST and PATCH.

If properties `name` and `location.volume.name` and/or `location.volume.uuid` are specified in the same request, they must refer to the same volume.

A PATCH that modifies the volume of the LUN begins an asynchronous LUN movement operation.

Name	Type	Description
name	string	The name of the volume. This field cannot be specified in a PATCH method.
uuid	string	Unique identifier for the volume. This corresponds to the instance-uuid that is exposed in the CLI and ONTAPI. It does not change due to a volume move. • example: 028baa66-41bd-11e9-81d5-00a0986138f7 • Introduced in: 9.6 • x-nullable: true

location

The location of the LUN within the ONTAP cluster. LUNs support rename and move between volumes. Valid in POST and PATCH.

Name	Type	Description
logical_unit	string	The base name component of the LUN. Valid in POST and PATCH. If properties `name` and `location.logical_unit` are specified in the same request, they must refer to the base name. A PATCH that modifies the base name of the LUN is considered a rename operation.
node	node	The cluster node that hosts the LUN.

Name	Type	Description
qtree	qtree	The qtree in which the LUN is optionally located. Valid in POST and PATCH. If properties `name` and `location.qtree.name` and/or `location.qtree.id` are specified in the same request, they must refer to the same qtree. A PATCH that modifies the qtree of the LUN is considered a rename operation.
volume	volume	The volume in which the LUN is located. Valid in POST and PATCH. If properties `name` and `location.volume.name` and/or `location.volume.uuid` are specified in the same request, they must refer to the same volume. A PATCH that modifies the volume of the LUN begins an asynchronous LUN movement operation.

self_link

igroups

Name	Type	Description
name	string	The name of the initiator group.
uuid	string	The unique identifier of the initiator group.

initiators

The initiators that are members of the initiator group.

Name	Type	Description
comment	string	A comment available for use by the administrator.
name	string	Name of initiator that is a member of the initiator group.

igroup

The initiator group to which the LUN is mapped.

Name	Type	Description
comment	string	A comment available for use by the administrator. Valid in POST and PATCH.
igroups	array[igroups]	The existing initiator groups that are members of the group. Optional in POST. This property is mutually exclusive with the *initiators* property during POST. This array contains only the direct children of the initiator group. If the member initiator groups have further nested initiator groups, those are reported in the `igroups` property of the child initiator group. Zero or more nested initiator groups can be supplied when the initiator group is created. The initiator group will act as if it contains the aggregation of all initiators in any nested initiator groups. After creation, nested initiator groups can be added or removed from the initiator group using the `/protocols/san/igroups/{igroup.uuid}/igroups` endpoint. See DELETE /protocols/san/igroups/{igroup.uuid}/igroups/{uuid} for more details.

Name	Type	Description
initiators	array[initiators]	The initiators that are members of the group.
name	string	The name of the initiator group.
os_type	string	The host operating system of the initiator group. All initiators in the group should be hosts of the same operating system.
protocol	string	The protocols supported by the initiator group. This restricts the type of initiators that can be added to the initiator group. Optional in POST; if not supplied, this defaults to *mixed*. The protocol of an initiator group cannot be changed after creation of the group.
uuid	string	The unique identifier of the initiator group.

lun_maps

A LUN map is an association between a LUN and an initiator group.

When a LUN is mapped to an initiator group, the initiator group's initiators are granted access to the LUN. The relationship between a LUN and an initiator group is many LUNs to many initiator groups.

Name	Type	Description
igroup	igroup	The initiator group to which the LUN is mapped.

Name	Type	Description
logical_unit_number	integer	The logical unit number assigned to the LUN when mapped to the specified initiator group. The number is used to identify the LUN to initiators in the initiator group when communicating through the Fibre Channel Protocol or iSCSI. Optional in POST; if no value is provided, ONTAP assigns the lowest available value. This property is not supported when the *provisioning_options.count* property is 2 or more. • Introduced in: 9.6 • readCreate: 1 • x-nullable: true

iops

The rate of I/O operations observed at the storage object.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

latency

The round trip latency in microseconds observed at the storage object.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

throughput

The rate of throughput bytes per second observed at the storage object.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

metric

Performance numbers, such as IOPS latency and throughput.

Name	Type	Description
duration	string	The duration over which this sample is calculated. The time durations are represented in the ISO-8601 standard format. Samples can be calculated over the following durations:
iops	iops	The rate of I/O operations observed at the storage object.
latency	latency	The round trip latency in microseconds observed at the storage object.
status	string	Errors associated with the sample. For example, if the aggregation of data over multiple nodes fails, then any partial errors might return "ok" on success or "error" on an internal uncategorized failure. Whenever a sample collection is missed but done at a later time, it is back filled to the previous 15 second timestamp and tagged with "backfilled_data". "Inconsistent_ delta_time" is encountered when the time between two collections is not the same for all nodes. Therefore, the aggregated value might be over or under inflated. "Negative_delta" is returned when an expected monotonically increasing value has decreased in value. "Inconsistent_old_data" is returned when one or more nodes do not have the latest data.
throughput	throughput	The rate of throughput bytes per second observed at the storage object.
timestamp	string	The timestamp of the performance data.

paths

The fully qualified LUN path names involved in the LUN movement.

Name	Type	Description
destination	string	The fully qualified path of the LUN movement destination composed of a "/vol" prefix, the volume name, the optional qtree name, and base name of the LUN.
source	string	The fully qualified path of the LUN movement source composed of a "/vol" prefix, the volume name, the optional qtree name, and base name of the LUN.

error

Error information provided if the asynchronous LUN movement operation fails.

Name	Type	Description
arguments	array[error_arguments]	Message arguments
code	string	Error code
message	string	Error message

progress

Properties related to the progress of an active or recently completed LUN movement.

Name	Type	Description
elapsed	integer	The amount of time that has elapsed since the start of the LUN movement, in seconds.
percent_complete	integer	The percentage completed of the LUN movement.
volume_snapshot_blocked	boolean	This property reports if volume snapshots are blocked by the LUN movement. This property can be polled to identify when volume snapshots can be resumed after beginning a LUN movement.

movement

This sub-object applies to LUN movement between volumes. A LUN can be moved to a new volume with a PATCH request that changes either the volume portion of property `name`, `location.volume.uuid`, or `location.volume.name`. If the volume is changed using more than one of these properties, the supplied properties used must refer to the same volume.

Moving a LUN between volumes is an asynchronous activity begun by a PATCH request. The data for the LUN is then asynchronously copied from the source volume to the destination volume. The time required to complete the move depends on the size of the LUN and the load on the cluster. The `movement` sub-object is populated while a LUN movement is in progress and for two (2) minutes following completion of a movement.

While the LUN is being moved, the status of the LUN movement operation can be obtained using a GET for the LUN that requests the `movement` properties. The LUN movement operation can be further modified using a PATCH on the properties on the `movement` sub-object.

There is an added computational cost to retrieving property values for `movement`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
max_throughput	integer	The maximum data throughput, in bytes per second, that should be utilized in support of the LUN movement. This property can be used to throttle a transfer and limit its impact on the performance of the source and destination nodes. The specified value will be rounded up to the nearest megabyte. If this property is not specified in a POST that begins a LUN movement, throttling is not applied to the data transfer. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. This property is valid only in a POST that begins a LUN movement or a PATCH when a LUN movement is already in process. • format: int64 • Introduced in: 9.6 • x-nullable: true

Name	Type	Description
paths	paths	The fully qualified LUN path names involved in the LUN movement.

qos_policy

The QoS policy for the volume provisioned to host the LUN. This property is only supported when the request provisions a new volume. This property is mutually exclusive with the LUN granular `qos_policy`. If no `qos_policy` is provided at LUN or volume granularity, a volume granular policy is set based on the `storage_service.name`, which defaults to the most performant service available.

Name	Type	Description
name	string	The QoS policy group name. This is mutually exclusive with UUID and other QoS attributes during POST and PATCH.
uuid	string	The QoS policy group UUID. This is mutually exclusive with name and other QoS attributes during POST and PATCH.

snapshot_policy_reference

This is a reference to the snapshot policy.

Name	Type	Description
name	string	
uuid	string	

storage_service

Determines the placement of the LUN based on the value specified. This property is only supported for regular and vvol LUNs. Valid in POST.

Name	Type	Description
name	string	Storage service name. If not specified, the default value is the most performant for the platform.

object_stores

Name	Type	Description
name	string	The name of the object store to use. Used for placement.

tiering

The tiering placement and policy definitions for the volume provisioned to host the LUN. This property is only supported when the request provisions a new volume.

Name	Type	Description
control	string	Storage tiering placement rules for the object.
object_stores	array[object_stores]	Object stores to use. Used for placement.

Name	Type	Description
policy	string	Policy that determines whether the user data blocks of a volume in a FabricPool will be tiered to the cloud store when they become cold. FabricPool combines flash (performance tier) with a cloud store into a single aggregate. Temperature of a volume block increases if it is accessed frequently and decreases when it is not. Valid in POST or PATCH. all ‐ Allows tiering of both snapshots and active file system user data to the cloud store as soon as possible by ignoring the temperature on the volume blocks. auto ‐ Allows tiering of both snapshot and active file system user data to the cloud store none ‐ Volume blocks are not be tiered to the cloud store. snapshot_only ‐ Allows tiering of only the volume snapshots not associated with the active file system. The default tiering policy is "snapshot-only" for a FlexVol volume and "none" for a FlexGroup volume. The default minimum cooling period for the "snapshot-only" tiering policy is 2 days and for the "auto" tiering policy it is 31 days.

provisioning_options

Options that are applied to the operation.

Name	Type	Description
auto	boolean	If the volume specified in the request does not exist, automatically provision one of appropriate size. If the volume does exist, resize it to accommodate the new LUN. This property is only supported on Unified ONTAP. The following behavior is different from a traditional POST request: • The operation is asynchronous. • The `qos_policy` property is applied to the provisioned volume instead of the LUN. A default QoS policy is applied to the volume if one is not provided. • The `provisioning_options.count` property is supported, provisioning *count* LUNs on the volume using the specified properties. • The `lun_maps` property is supported. If the specified initiator group does not exist, it is created. The LUN is mapped to this initiator group. If an initiator group is provisioned in this way, it is deleted after it is no longer mapped to any LUNs. • The `clone`, `copy`, and `convert` properties are not supported. • When performing `records` based operations, specifying this property in the query applies to the entire operation. Specifying it for an individual record within the request applies to only that record. • Many other `provisioning_options` properties are supported to control the placement of the LUN and the properties of the volume containing the LUN.

Name	Type	Description
count	integer	The number of LUNs to provision with these properties. Only POST requests based on `space.size` are supported. When provided, the name is considered a prefix, and a suffix of the form _<*N*> is generated where N is the next available numeric index, starting with 1.
qos_policy	qos_policy	The QoS policy for the volume provisioned to host the LUN. This property is only supported when the request provisions a new volume. This property is mutually exclusive with the LUN granular `qos_policy`. If no `qos_policy` is provided at LUN or volume granularity, a volume granular policy is set based on the `storage_service.name`, which defaults to the most performant service available.
snapshot_policy	snapshot_policy_reference	This is a reference to the snapshot policy.
storage_service	storage_service	Determines the placement of the LUN based on the value specified. This property is only supported for regular and vvol LUNs. Valid in POST.
tiering	tiering	The tiering placement and policy definitions for the volume provisioned to host the LUN. This property is only supported when the request provisions a new volume.

Name	Type	Description
use_mirrored_aggregates	boolean	Specifies whether mirrored aggregates are selected when provisioning the volume to host the LUN. Only mirrored aggregates are used if this parameter is set to *true* and only unmirrored aggregates are used if this parameter is set to *false*. The default value is *true* for a MetroCluster configuration and is *false* for a non-MetroCluster configuration.

qos_policy

The QoS policy for the LUN. Both traditional and adaptive QoS policies are supported. If both property `qos_policy.uuid` and `qos_policy.name` are specified in the same request, they must refer to the same QoS policy. To remove the QoS policy from a LUN, leaving it with no QoS policy, set the property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.

Name	Type	Description
name	string	The name of the QoS policy. To remove the QoS policy from a LUN, leaving it with no QoS policy, set this property to an empty string ("") in a PATCH request. Valid in POST and PATCH.
uuid	string	The unique identifier of the QoS policy. Valid in POST and PATCH.

guarantee

Properties that request and report the space guarantee for the LUN.

Name	Type	Description
requested	boolean	The requested space reservation policy for the LUN. If *true*, a space reservation is requested for the LUN; if *false*, the LUN is thin provisioned. Guaranteeing a space reservation request for a LUN requires that the volume in which the LUN resides is also space reserved and that the fractional reserve for the volume is 100%. Valid in POST and PATCH. This property is caller settable as described above.
reserved	boolean	Reports if the LUN is space guaranteed. If *true*, a space guarantee is requested and the containing volume and aggregate support the request. If *false*, a space guarantee is not requested or a space guarantee is requested and either the containing volume or aggregate do not support the request.

space

The storage space related properties of the LUN.

Name	Type	Description
efficiency_ratio	number	The storage efficiency ratio of the LUN without snapshots. (Logical Used / Used) This property is not available on the LUN object in the REST API and is not reported for GET requests. See the containing volume object for this information.
guarantee	guarantee	Properties that request and report the space guarantee for the LUN.

Name	Type	Description
physical_used	integer	The number of bytes consumed on the disk by the LUN, excluding snapshots. This property is not available on the LUN object in the REST API and is not reported for GET requests. See the containing volume object for this information.
physical_used_by_snapshots	integer	The number of bytes consumed on the disk by the LUN's snapshots. This property is not available on the LUN object in the REST API and is not reported for GET requests. See the containing volume object for this information.

Name	Type	Description
scsi_thin_provisioning_support_enabled	boolean	To leverage the benefits of SCSI thin provisioning, it must be supported by your host. SCSI thin provisioning uses the Logical Block Provisioning feature as defined in the SCSI SBC-3 standard. Only hosts that support this standard can use SCSI thin provisioning in ONTAP. When you disable SCSI thin provisioning support in ONTAP, you turn off the following SCSI thin provisioning features: • Unmapping and reporting space usage for space reclamation • Reporting resource exhaustion errors The value of this property is not propagated to the destination when a LUN is cloned as a new LUN or copied; it is reset to false. The value of this property is maintained from the destination LUN when a LUN is overwritten as a clone. Valid in POST and PATCH. • Default value: 1 • Introduced in: 9.10 • x-nullable: true

Name	Type	Description
size	integer	The total provisioned size of the LUN. The LUN size can be increased but not decreased using the REST interface. The maximum and minimum sizes listed here are the absolute maximum and absolute minimum sizes, in bytes. The actual minimum and maximum sizes vary depending on the ONTAP version, ONTAP platform and the available space in the containing volume and aggregate. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. • example: 1073741824 • format: int64 • Max value: 140737488355328 • Min value: 4096 • Introduced in: 9.6 • x-nullable: true

Name	Type	Description
used	integer	The amount of space consumed by the main data stream of the LUN. This value is the total space consumed in the volume by the LUN, including filesystem overhead, but excluding prefix and suffix streams. Due to internal filesystem overhead and the many ways SAN filesystems and applications utilize blocks within a LUN, this value does not necessarily reflect actual consumption/availability from the perspective of the filesystem or application. Without specific knowledge of how the LUN blocks are utilized outside of ONTAP, this property should not be used as an indicator for an out-of-space condition. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. • format: int64 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true

iops_raw

The number of I/O operations observed at the storage object. This can be used along with delta time to calculate the rate of I/O operations per unit of time.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.

Name	Type	Description
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

latency_raw

The raw latency in microseconds observed at the storage object. This can be divided by the raw IOPS value to calculate the average latency per I/O operation.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

throughput_raw

Throughput bytes observed at the storage object. This can be used along with delta time to calculate the rate of throughput bytes per unit of time.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.

Name	Type	Description
write	integer	Performance metric for write I/O operations.

statistics

These are raw performance numbers, such as IOPS latency and throughput. These numbers are aggregated across all nodes in the cluster and increase with the uptime of the cluster.

Name	Type	Description
iops_raw	iops_raw	The number of I/O operations observed at the storage object. This can be used along with delta time to calculate the rate of I/O operations per unit of time.
latency_raw	latency_raw	The raw latency in microseconds observed at the storage object. This can be divided by the raw IOPS value to calculate the average latency per I/O operation.
status	string	Errors associated with the sample. For example, if the aggregation of data over multiple nodes fails, then any partial errors might return "ok" on success or "error" on an internal uncategorized failure. Whenever a sample collection is missed but done at a later time, it is back filled to the previous 15 second timestamp and tagged with "backfilled_data". "Inconsistent_ delta_time" is encountered when the time between two collections is not the same for all nodes. Therefore, the aggregated value might be over or under inflated. "Negative_delta" is returned when an expected monotonically increasing value has decreased in value. "Inconsistent_old_data" is returned when one or more nodes do not have the latest data.

Name	Type	Description
throughput_raw	throughput_raw	Throughput bytes observed at the storage object. This can be used along with delta time to calculate the rate of throughput bytes per unit of time.
timestamp	string	The timestamp of the performance data.

status

Status information about the LUN.

Name	Type	Description
container_state	string	The state of the volume and aggregate that contain the LUN. LUNs are only available when their containers are available.
mapped	boolean	Reports if the LUN is mapped to one or more initiator groups. There is an added computational cost to retrieving this property's value. It is not populated for a GET request unless it is explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
read_only	boolean	Reports if the LUN allows only read access.
state	string	The state of the LUN. Normal states for a LUN are *online* and *offline*. Other states indicate errors.

svm

The SVM in which the LUN is located.

Name	Type	Description
name	string	The name of the SVM. This field cannot be specified in a PATCH method.

Name	Type	Description
uuid	string	The unique identifier of the SVM. This field cannot be specified in a PATCH method.

partner

The LUN partner that this LUN is bound to. If this LUN is a `vvol` class LUN, the partner is a `protocol_endpoint` class LUN.

Name	Type	Description
name	string	The name of the partner LUN.
uuid	string	The unique identifier of the partner LUN.

bindings

A vVol binding with which the LUN is associated.

Name	Type	Description
id	integer	The ONTAP internal identifier assigned to the vVol binding. The bind identifier is unique amongst all class `vvol` LUNs bound to the same class `protocol_endpoint` LUN. This property was included in early releases of the REST API for vVols and is maintained for backward compatibility. See the `secondary_id` property, which replaces `id`. • example: 1 • readOnly: 1 • x-ntap-deprecated: 9.13.1 • Introduced in: 9.10 • x-nullable: true
partner	partner	The LUN partner that this LUN is bound to. If this LUN is a `vvol` class LUN, the partner is a `protocol_endpoint` class LUN.

Name	Type	Description
secondary_id	string	The identifier assigned to the vVol binding, known as the secondary LUN ID. The identifier is unique amongst all class `vvol` LUNs bound to the same class `protocol_endpoint` LUN. The format for a secondary LUN ID is 16 hexadecimal digits (zero-filled) followed by a lower case "h".

vvol

A VMware virtual volume (vVol) binding is an association between a LUN of class `protocol_endpoint` and a LUN of class `vvol`. Class `protocol_endpoint` LUNs are mapped to igroups and granted access using the same configuration as class `regular` LUNs. When a class `vvol` LUN is bound to a mapped class `protocol_endpoint` LUN, VMware can access the class `vvol` LUN through the class `protocol_endpoint` LUN mapping.

See DELETE /protocols/san/vvol-bindings to learn more about deleting vVol bindings.

There is an added computational cost to retrieving property values for `vvol`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
bindings	array[bindings]	Bindings between the LUN, which must be of class `protocol_endpoint` or `vvol`, and LUNs of the opposite class. A class `vvol` LUN must be bound to a class `protocol_endpoint` LUN in order to be accessed. Class `protocol_endpoint` and `vvol` LUNs allow many-to-many bindings. A LUN of one class is allowed to be bound to zero or more LUNs of the opposite class. The binding between any two specific LUNs is reference counted. When a binding is created that already exists, the binding count is incremented. When a binding is deleted, the binding count is decremented, but the LUNs remain bound if the resultant reference count is greater than zero. When the binding count reaches zero, the binding is destroyed. The bindings array contains LUNs of the opposite class of the containing LUN object. There is an added computational cost to retrieving property values for `vvol.bindings`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
is_bound	boolean	Reports if the LUN is part of a VMware virtual volume (vVol) bind relationship. This is `true` if the LUN is of class `protocol_endpoint` or `vvol` and has one or more bindings to a LUN of the opposite class. This is false if the LUN is of class `regular` or unbound.

lun

A LUN is the logical representation of storage in a storage area network (SAN).

A LUN must be mapped to an initiator group to grant access to the initiator group’s initiators (client hosts). Initiators can then access the LUN and perform I/O over a Fibre Channel (FC) fabric using the FC Protocol or a TCP/IP network using iSCSI.

See the LUN object model to learn more about each of the properties supported by the LUN REST API.

A LUN is located within a volume. Optionally, it can be located within a qtree in a volume.

LUN names are paths of the form "/vol/<volume>[/<qtree>]/<lun>" where the qtree name is optional.

A LUN can be created to a specified size using thin or thick provisioning. A LUN can then be renamed, resized, cloned, moved to a different volume and copied. LUNs support the assignment of a QoS policy for performance management or a QoS policy can be assigned to a volume containing one or more LUNs.

Name	Type	Description
attributes	array[attributes]	An array of name/value pairs optionally stored with the LUN. Attributes are available to callers to persist small amounts of application-specific metadata. They are in no way interpreted by ONTAP. Attribute names and values must be at least one byte and no more than 4091 bytes in length. The sum of the name and value lengths must be no more than 4092 bytes. Valid in POST except when creating a LUN clone. A cloned can already have attributes from its source. You can add, modify, and delete the attributes of a LUN clone in separate requests after creation of the LUN. Attributes can be added/modified/removed for an existing LUN using the /api/storage/luns/{lun.uuid}/attributes endpoint. For further information, see DOC /storage/luns/{lun.uuid}/attributes . There is an added computational cost to retrieving property values for `attributes`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more. • Introduced in: 9.10 • readCreate: 1

Name	Type	Description
auto_delete	boolean	This property marks the LUN for auto deletion when the volume containing the LUN runs out of space. This is most commonly set on LUN clones. When set to *true*, the LUN becomes eligible for automatic deletion when the volume runs out of space. Auto deletion only occurs when the volume containing the LUN is also configured for auto deletion and free space in the volume decreases below a particular threshold. This property is optional in POST and PATCH. The default value for a new LUN is *false*. There is an added computational cost to retrieving this property's value. It is not populated for a GET request unless it is explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
class	string	The class of LUN. Optional in POST.

Name	Type	Description
clone	clone	This sub-object is used in POST to create a new LUN as a clone of an existing LUN, or PATCH to overwrite an existing LUN as a clone of another. Setting a property in this sub-object indicates that a LUN clone is desired. Consider the following other properties when cloning a LUN: `auto_delete`, `qos_policy`, `space.guarantee.requested` and `space.scsi_thin_provisioning_support_enabled`. When used in a PATCH, the patched LUN's data is over-written as a clone of the source and the following properties are preserved from the patched LUN unless otherwise specified as part of the PATCH: `class`, `auto_delete`, `lun_maps`, `serial_number`, `status.state`, and `uuid`. Persistent reservations for the patched LUN are also preserved.
comment	string	A configurable comment available for use by the administrator. Valid in POST and PATCH.
consistency_group	consistency_group	The LUN's consistency group. This property is populated for LUNs that are members of a consistency group. If the LUN is a member of a child consistency group, the parent consistency group is reported. A LUN's consistency group is the consistency group of its containing volume.

Name	Type	Description
convert	convert	This sub-object is used in POST to convert a valid in-place NVMe namespace to a LUN. Setting a property in this sub-object indicates that a conversion from the specified NVMe namespace to LUN is desired.

Name	Type	Description
copy	copy	This sub-object applies to LUN copy operations. A LUN can be copied with a POST request that supplies `copy.source` properties. Copying a LUN is an asynchronous activity begun by a POST request that specifies the source of the copy in the `copy.source` properties. The data for the LUN is then asynchronously copied from the source to the destination. The time required to complete the copy depends on the size of the LUN and the load on the cluster. The `copy` sub-object is populated while a LUN copy is in progress and for two (2) minutes following completion of a copy. While LUNs are being copied, the status of the LUN copy operations can be obtained using a GET of the source or destination LUN that requests the `copy` properties. If the LUN is the source LUN for one or more copy operations, the `copy.destinations` array is populated in GET. If the containing LUN is the destination LUN for a copy operation, the `copy.source` sub-object is populated in GET. The LUN copy operation can be further modified using a PATCH on the properties on the `copy.source` sub-object of the copy destination LUN. There is an added computational cost to retrieving property values for `copy`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
create_time	string	The time the LUN was created.

Name	Type	Description
location	location	The location of the LUN within the ONTAP cluster. LUNs support rename and move between volumes. Valid in POST and PATCH. • Introduced in: 9.6
lun_maps	array[lun_maps]	The LUN maps with which the LUN is associated. There is an added computational cost to retrieving property values for `lun_maps`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more. These properties are supported for GET only.

Name	Type	Description
movement	movement	This sub-object applies to LUN movement between volumes. A LUN can be moved to a new volume with a PATCH request that changes either the volume portion of property `name`, `location.volume.uuid`, or `location.volume.name`. If the volume is changed using more than one of these properties, the supplied properties used must refer to the same volume. Moving a LUN between volumes is an asynchronous activity begun by a PATCH request. The data for the LUN is then asynchronously copied from the source volume to the destination volume. The time required to complete the move depends on the size of the LUN and the load on the cluster. The `movement` sub-object is populated while a LUN movement is in progress and for two (2) minutes following completion of a movement. While the LUN is being moved, the status of the LUN movement operation can be obtained using a GET for the LUN that requests the `movement` properties. The LUN movement operation can be further modified using a PATCH on the properties on the `movement` sub-object. There is an added computational cost to retrieving property values for `movement`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
name	string	The name of the LUN. Valid in POST and PATCH. A LUN is located within a volume. Optionally, it can be located within a qtree in a volume. LUN names are paths of the form "/vol/<volume>[/<qtree>]/<lun>" where the qtree name is optional. A PATCH that modifies the qtree and/or base name portion of the LUN path is considered a rename operation. A PATCH that modifies the volume portion of the LUN path begins an asynchronous LUN movement operation.
os_type	string	The operating system type of the LUN. Required in POST when creating a LUN that is not a clone of another. Disallowed in POST when creating a LUN clone.
provisioning_options	provisioning_options	Options that are applied to the operation.
qos_policy	qos_policy	The QoS policy for the LUN. Both traditional and adaptive QoS policies are supported. If both property `qos_policy.uuid` and `qos_policy.name` are specified in the same request, they must refer to the same QoS policy. To remove the QoS policy from a LUN, leaving it with no QoS policy, set the property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.

Name	Type	Description
serial_number	string	The LUN serial number. The serial number is generated by ONTAP when the LUN is created. • maxLength: 12 • minLength: 12 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true
serial_number_hex	string	The LUN serial number encoded in hexadecimal format. The serial number is generated by ONTAP when the LUN is created. • readOnly: 1 • Introduced in: 9.17 • x-nullable: true
space	space	The storage space related properties of the LUN.
status	status	Status information about the LUN.
svm	svm	The SVM in which the LUN is located.
uuid	string	The unique identifier of the LUN. The UUID is generated by ONTAP when the LUN is created. • example: 1cd8a442-86d1-11e0-ae1c-123478563412 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true

Name	Type	Description
vvol	vvol	A VMware virtual volume (vVol) binding is an association between a LUN of class `protocol_endpoint` and a LUN of class `vvol`. Class `protocol_endpoint` LUNs are mapped to igroups and granted access using the same configuration as class `regular` LUNs. When a class `vvol` LUN is bound to a mapped class `protocol_endpoint` LUN, VMware can access the class `vvol` LUN through the class `protocol_endpoint` LUN mapping. See DELETE /protocols/san/vvol-bindings to learn more about deleting vVol bindings. There is an added computational cost to retrieving property values for `vvol`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

job_link

Name	Type	Description
uuid	string	The UUID of the asynchronous job that is triggered by a POST, PATCH, or DELETE operation.

returned_error

Name	Type	Description
arguments	array[error_arguments]	Message arguments
code	string	Error code
message	string	Error message

Name	Type	Description
target	string	The target parameter that caused the error.

Delete a LUN

DELETE `/storage/luns/{uuid}`

Introduced In: 9.6

Deletes a LUN.

Related ONTAP commands

- `lun copy cancel`
- `lun delete`

Learn more

- DOC /storage/luns

Parameters

Name	Type	In	Required	Description
uuid	string	path	True	The unique identifier of the LUN to retrieve.
allow_delete_while_mapped	boolean	query	False	Allows deletion of a mapped LUN. A mapped LUN might be in use. Deleting a mapped LUN also deletes the LUN map and makes the data no longer available. This might cause a disruption in the availability of data. **This parameter should be used with caution.** • Default value:

Name	Type	In	Required	Description
return_timeout	integer	query	False	The number of seconds to allow the call to execute before returning. When doing a POST, PATCH, or DELETE operation on a single record, the default is 0 seconds. This means that if an asynchronous operation is started, the server immediately returns HTTP code 202 (Accepted) along with a link to the job. If a non-zero value is specified for POST, PATCH, or DELETE operations, ONTAP waits that length of time to see if the job completes so it can return something other than 202. • Default value: 0 • Max value: 120 • Min value: 0

Response

```
Status: 200, Ok
```

Response

```
Status: 202, Accepted
```

Error

```
Status: Default
```

ONTAP Error Response Codes

Error Code	Description
1254197	The LUN is mapped and cannot be deleted without specifying the `allow_delete_while_mapped` query parameter.
5374705	Deleting the LUN is not allowed because it is part of an application.
5374865	The LUN's aggregate is offline. The aggregate must be online to modify or remove the LUN.
5374866	The LUN's volume is offline. The volume must be online to modify or remove the LUN.
5374875	The specified LUN was not found.
5374876	The specified LUN was not found.

Also see the table of common errors in the Response body overview section of this documentation.

Name	Type	Description
error	returned_error	

Example error

```
{
  "error": {
    "arguments": [
      {
        "code": "string",
        "message": "string"
      }
    ],
    "code": "4",
    "message": "entry doesn't exist",
    "target": "uuid"
  }
}
```

Definitions

See Definitions

error_arguments

Name	Type	Description
code	string	Argument code
message	string	Message argument

returned_error

Name	Type	Description
arguments	array[error_arguments]	Message arguments
code	string	Error code
message	string	Error message
target	string	The target parameter that caused the error.

Retrieve LUN properties or data

GET `/storage/luns/{uuid}`

Introduced In: 9.6

Retrieves a LUN's properties or a LUN's data.

LUN data read requests are distinguished by the header entry `Accept: multipart/form-data`. When this header entry is provided, query parameters `data.offset` and `data.size` are required and used to specify the portion of the LUN's data to read; no other query parameters are allowed. Reads are limited to one megabyte (1MB) per request. Data is returned as `multipart/form-data` content with exactly one form entry containing the data. The form entry has content type `application/octet-stream`.

Expensive properties

There is an added computational cost to retrieving values for these properties. They are not included by default in GET results and must be explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

- `attributes.*`
- `auto_delete`
- `copy.*`
- `lun_maps.*`

- `movement.*`
- `space.physical_used`
- `space.physical_used_by_snapshots`
- `space.efficiency_ratio`
- `statistics.*`
- `vvol.bindings.*`
- `metric.*`

Related ONTAP commands

- `lun bind show`
- `lun copy show`
- `lun mapping show`
- `lun move show`
- `lun show`
- `volume file clone show-autodelete`

Learn more

- DOC /storage/luns

Parameters

Name	Type	In	Required	Description
uuid	string	path	True	The unique identifier of the LUN to retrieve.

Name	Type	In	Required	Description
data.offset	integer	query	False	The offset, in bytes, at which to begin reading LUN data. LUN data read requests are distinguished by the header entry `Accept: multipart/form-data`. When this header entry is provided, query parameters `data.offset` and `data.size` are required and used to specify the portion of the LUN's data to read; no other query parameters are allowed. Reads are limited to one megabyte (1MB) per request. Data is returned as `multipart/form-data` content with exactly one form entry containing the data. The form entry has content type `application/octet-stream`. • format: int64 • Min value: 0 • Introduced in: 9.11

Name	Type	In	Required	Description
data.size	integer	query	False	The size, in bytes, of LUN data to read. LUN data read requests are distinguished by the header entry `Accept: multipart/form-data`. When this header entry is provided, query parameters `data.offset` and `data.size` are required and used to specify the portion of the LUN's data to read; no other query parameters are allowed. Reads are limited to one megabyte (1MB) per request. Data is returned as `multipart/form-data` content with exactly one form entry containing the data. The form entry has content type `application/octet-stream`. • format: int64 • Min value: 1 • Introduced in: 9.11 • Max value: 1048576
fields	array[string]	query	False	Specify the fields to return.

Response

```
Status: 200, Ok
```

Name	Type	Description
_links	_links	
attributes	array[attributes]	An array of name/value pairs optionally stored with the LUN. Attributes are available to callers to persist small amounts of application-specific metadata. They are in no way interpreted by ONTAP. Attribute names and values must be at least one byte and no more than 4091 bytes in length. The sum of the name and value lengths must be no more than 4092 bytes. Valid in POST except when creating a LUN clone. A cloned can already have attributes from its source. You can add, modify, and delete the attributes of a LUN clone in separate requests after creation of the LUN. Attributes can be added/modified/removed for an existing LUN using the /api/storage/luns/{lun.uuid}/attributes endpoint. For further information, see DOC /storage/luns/{lun.uuid}/attributes . There is an added computational cost to retrieving property values for `attributes`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more. • Introduced in: 9.10 • readCreate: 1

Name	Type	Description
auto_delete	boolean	This property marks the LUN for auto deletion when the volume containing the LUN runs out of space. This is most commonly set on LUN clones. When set to *true*, the LUN becomes eligible for automatic deletion when the volume runs out of space. Auto deletion only occurs when the volume containing the LUN is also configured for auto deletion and free space in the volume decreases below a particular threshold. This property is optional in POST and PATCH. The default value for a new LUN is *false*. There is an added computational cost to retrieving this property's value. It is not populated for a GET request unless it is explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
class	string	The class of LUN. Optional in POST.
comment	string	A configurable comment available for use by the administrator. Valid in POST and PATCH.
consistency_group	consistency_group	The LUN's consistency group. This property is populated for LUNs that are members of a consistency group. If the LUN is a member of a child consistency group, the parent consistency group is reported. A LUN's consistency group is the consistency group of its containing volume.

Name	Type	Description
copy	copy	This sub-object applies to LUN copy operations. A LUN can be copied with a POST request that supplies `copy.source` properties. Copying a LUN is an asynchronous activity begun by a POST request that specifies the source of the copy in the `copy.source` properties. The data for the LUN is then asynchronously copied from the source to the destination. The time required to complete the copy depends on the size of the LUN and the load on the cluster. The `copy` sub-object is populated while a LUN copy is in progress and for two (2) minutes following completion of a copy. While LUNs are being copied, the status of the LUN copy operations can be obtained using a GET of the source or destination LUN that requests the `copy` properties. If the LUN is the source LUN for one or more copy operations, the `copy.destinations` array is populated in GET. If the containing LUN is the destination LUN for a copy operation, the `copy.source` sub-object is populated in GET. The LUN copy operation can be further modified using a PATCH on the properties on the `copy.source` sub-object of the copy destination LUN. There is an added computational cost to retrieving property values for `copy`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
create_time	string	The time the LUN was created.

Name	Type	Description
enabled	boolean	The enabled state of the LUN. LUNs can be disabled to prevent access to the LUN. Certain error conditions also cause the LUN to become disabled. If the LUN is disabled, check the `status.state` property to determine if the LUN is administratively disabled (*offline*) or has become disabled as a result of an error. A LUN in an error condition can be brought online by setting the `enabled` property to *true* or brought administratively offline by setting the `enabled` property to *false*. Upon creation, a LUN is enabled by default. Valid in PATCH.
location	location	The location of the LUN within the ONTAP cluster. LUNs support rename and move between volumes. Valid in POST and PATCH. • Introduced in: 9.6
lun_maps	array[lun_maps]	The LUN maps with which the LUN is associated. There is an added computational cost to retrieving property values for `lun_maps`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more. These properties are supported for GET only.
metric	metric	Performance numbers, such as IOPS latency and throughput.

Name	Type	Description
movement	movement	This sub-object applies to LUN movement between volumes. A LUN can be moved to a new volume with a PATCH request that changes either the volume portion of property `name`, `location.volume.uuid`, or `location.volume.name`. If the volume is changed using more than one of these properties, the supplied properties used must refer to the same volume. Moving a LUN between volumes is an asynchronous activity begun by a PATCH request. The data for the LUN is then asynchronously copied from the source volume to the destination volume. The time required to complete the move depends on the size of the LUN and the load on the cluster. The `movement` sub-object is populated while a LUN movement is in progress and for two (2) minutes following completion of a movement. While the LUN is being moved, the status of the LUN movement operation can be obtained using a GET for the LUN that requests the `movement` properties. The LUN movement operation can be further modified using a PATCH on the properties on the `movement` sub-object. There is an added computational cost to retrieving property values for `movement`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
name	string	The name of the LUN. Valid in POST and PATCH. A LUN is located within a volume. Optionally, it can be located within a qtree in a volume. LUN names are paths of the form "/vol/<volume>[/<qtree>]/<lun>" where the qtree name is optional. A PATCH that modifies the qtree and/or base name portion of the LUN path is considered a rename operation. A PATCH that modifies the volume portion of the LUN path begins an asynchronous LUN movement operation.
os_type	string	The operating system type of the LUN. Required in POST when creating a LUN that is not a clone of another. Disallowed in POST when creating a LUN clone.
qos_policy	qos_policy	The QoS policy for the LUN. Both traditional and adaptive QoS policies are supported. If both property `qos_policy.uuid` and `qos_policy.name` are specified in the same request, they must refer to the same QoS policy. To remove the QoS policy from a LUN, leaving it with no QoS policy, set the property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.

Name	Type	Description
serial_number	string	The LUN serial number. The serial number is generated by ONTAP when the LUN is created. • maxLength: 12 • minLength: 12 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true
serial_number_hex	string	The LUN serial number encoded in hexadecimal format. The serial number is generated by ONTAP when the LUN is created. • readOnly: 1 • Introduced in: 9.17 • x-nullable: true
space	space	The storage space related properties of the LUN.
statistics	statistics	These are raw performance numbers, such as IOPS latency and throughput. These numbers are aggregated across all nodes in the cluster and increase with the uptime of the cluster.
status	status	Status information about the LUN.
svm	svm	The SVM in which the LUN is located.
uuid	string	The unique identifier of the LUN. The UUID is generated by ONTAP when the LUN is created. • example: 1cd8a442-86d1-11e0-ae1c-123478563412 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true

Name	Type	Description
vvol	vvol	A VMware virtual volume (vVol) binding is an association between a LUN of class `protocol_endpoint` and a LUN of class `vvol`. Class `protocol_endpoint` LUNs are mapped to igroups and granted access using the same configuration as class `regular` LUNs. When a class `vvol` LUN is bound to a mapped class `protocol_endpoint` LUN, VMware can access the class `vvol` LUN through the class `protocol_endpoint` LUN mapping. See DELETE /protocols/san/vvol-bindings to learn more about deleting vVol bindings. There is an added computational cost to retrieving property values for `vvol`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Example response


```
{
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "attributes": [
    {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "name": "name1",
      "value": "value1"
    }
  ],
  "class": "string",
  "comment": "string",
  "consistency_group": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "name": "cg1",
    "uuid": "4abc2317-4332-9d37-93a0-20bd29c22df0"
  },
  "copy": {
    "destinations": [
      {
        "_links": {
          "self": {
            "href": "/api/resourcelink"
          }
        },
        "max_throughput": 0,
        "name": "/vol/vol1/lun1",
        "peer": {
          "_links": {
            "self": {
              "href": "/api/resourcelink"
            }
          },
```

```
        "name": "peer1",
        "uuid": "4204cf77-4c82-9bdb-5644-b5a841c097a9"
      },
      "progress": {
        "elapsed": 0,
        "failure": {
          "arguments": [
            {
              "code": "string",
              "message": "string"
            }
          ],
          "code": "4",
          "message": "entry doesn't exist"
        },
        "percent_complete": 0,
        "state": "string"
      },
      "uuid": "1bc327d5-4654-5284-a116-f182282240b4"
    }
  ],
  "source": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "name": "/vol/vol2/lun1",
    "peer": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "name": "peer1",
      "uuid": "4204cf77-4c82-9bdb-5644-b5a841c097a9"
    },
    "progress": {
      "elapsed": 0,
      "failure": {
        "arguments": [
          {
            "code": "string",
            "message": "string"
          }
        ],
```

```
          "code": "4",
          "message": "entry doesn't exist"
        },
        "percent_complete": 0,
        "state": "string"
      },
      "uuid": "03c05019-40d9-3945-c767-dca4c3be5e90"
    }
  },
  "create_time": "2018-06-04 15:00:00 -0400",
  "location": {
    "logical_unit": "lun1",
    "node": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "name": "node1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    },
    "qtree": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "id": 1,
      "name": "qt1"
    },
    "volume": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "name": "volume1",
      "uuid": "028baa66-41bd-11e9-81d5-00a0986138f7"
    }
  },
  "lun_maps": [
    {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
```

```
      },
      "igroup": {
        "_links": {
          "self": {
            "href": "/api/resourcelink"
          }
        },
        "comment": "string",
        "igroups": [
          {
            "_links": {
              "self": {
                "href": "/api/resourcelink"
              }
            },
            "name": "igroup1",
            "uuid": "4ea7a442-86d1-11e0-ae1c-123478563412"
          }
        ],
        "initiators": [
          {
            "comment": "my comment",
            "name": "iqn.1998-01.com.corp.iscsi:name1"
          }
        ],
        "name": "igroup1",
        "os_type": "string",
        "protocol": "string",
        "uuid": "4ea7a442-86d1-11e0-ae1c-123478563412"
      }
    }
  ],
  "metric": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "duration": "PT15S",
    "iops": {
      "read": 200,
      "total": 1000,
      "write": 100
    },
    "latency": {
      "read": 200,
```

```
      "total": 1000,
      "write": 100
    },
    "status": "ok",
    "throughput": {
      "read": 200,
      "total": 1000,
      "write": 100
    },
    "timestamp": "2017-01-25 06:20:13 -0500"
  },
  "movement": {
    "paths": {
      "destination": "/vol/vol1/lun1",
      "source": "/vol/vol2/lun2"
    },
    "progress": {
      "elapsed": 0,
      "failure": {
        "arguments": [
          {
            "code": "string",
            "message": "string"
          }
        ],
        "code": "4",
        "message": "entry doesn't exist"
      },
      "percent_complete": 0,
      "state": "string"
    }
  },
  "name": "/vol/volume1/qtree1/lun1",
  "os_type": "string",
  "qos_policy": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "name": "qos1",
    "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
  },
  "serial_number": "string",
  "serial_number_hex": "string",
  "space": {
```

```
    "efficiency_ratio": 2.5,
    "physical_used": 1073741824,
    "physical_used_by_snapshots": 1073741824,
    "size": 1073741824,
    "used": 0
  },
  "statistics": {
    "iops_raw": {
      "read": 200,
      "total": 1000,
      "write": 100
    },
    "latency_raw": {
      "read": 200,
      "total": 1000,
      "write": 100
    },
    "status": "ok",
    "throughput_raw": {
      "read": 200,
      "total": 1000,
      "write": 100
    },
    "timestamp": "2017-01-25 06:20:13 -0500"
  },
  "status": {
    "container_state": "string",
    "state": "online"
  },
  "svm": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "name": "svm1",
    "uuid": "02c9e252-41be-11e9-81d5-00a0986138f7"
  },
  "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412",
  "vvol": {
    "bindings": [
      {
        "_links": {
          "self": {
            "href": "/api/resourcelink"
          }
```

```
        },
        "id": 1,
        "partner": {
          "_links": {
            "self": {
              "href": "/api/resourcelink"
            }
          },
          "name": "/vol/vol1/lun1",
          "uuid": "4ea7a442-86d1-11e0-ae1c-123478563412"
        },
        "secondary_id": "0000D20000010000h"
      }
    ]
  }
}
```


Error

```
Status: Default
```

ONTAP Error Response Codes

Error Code	Description
5374875	The specified LUN was not found.
5374876	The specified LUN was not found.

Also see the table of common errors in the Response body overview section of this documentation.

Name	Type	Description
error	returned_error	

Example error


```
{
  "error": {
    "arguments": [
      {
        "code": "string",
        "message": "string"
      }
    ],
    "code": "4",
    "message": "entry doesn't exist",
    "target": "uuid"
  }
}
```


Definitions

See Definitions

href

Name	Type	Description
href	string	

_links

Name	Type	Description
self	href	

attributes

A name/value pair optionally stored with the LUN. Attributes are available to callers to persist small amounts of application-specific metadata. They are in no way interpreted by ONTAP.

Attribute names and values must be at least one byte and no more than 4091 bytes in length. The sum of the name and value lengths must be no more than 4092 bytes.

Optional in POST.

Name	Type	Description
_links	_links	
name	string	The attribute name.
value	string	The attribute value.

source

The source LUN for a LUN clone operation. This can be specified using property `clone.source.uuid` or `clone.source.name`. If both properties are supplied, they must refer to the same LUN.

Valid in POST to create a new LUN as a clone of the source.

Valid in PATCH to overwrite an existing LUN's data as a clone of another.

clone

This sub-object is used in POST to create a new LUN as a clone of an existing LUN, or PATCH to overwrite an existing LUN as a clone of another. Setting a property in this sub-object indicates that a LUN clone is desired. Consider the following other properties when cloning a LUN: `auto_delete`, `qos_policy`, `space.guarantee.requested` and `space.scsi_thin_provisioning_support_enabled`.

When used in a PATCH, the patched LUN's data is over-written as a clone of the source and the following properties are preserved from the patched LUN unless otherwise specified as part of the PATCH: `class`, `auto_delete`, `lun_maps`, `serial_number`, `status.state`, and `uuid`.

Persistent reservations for the patched LUN are also preserved.

consistency_group

The LUN's consistency group. This property is populated for LUNs that are members of a consistency group. If the LUN is a member of a child consistency group, the parent consistency group is reported. A LUN's consistency group is the consistency group of its containing volume.

Name	Type	Description
_links	_links	
name	string	The name of the consistency group.
uuid	string	The unique identifier of the consistency group.

namespace

The source namespace for convert operation. This can be specified using property `convert.namespace.uuid` or `convert.namespace.name`. If both properties are supplied, they must refer to the same NVMe namespace.

Valid in POST. A convert request from NVMe namespace to LUN cannot be combined with setting any other LUN properties. All other properties of the converted LUN come from the source NVMe namespace.

convert

This sub-object is used in POST to convert a valid in-place NVMe namespace to a LUN. Setting a property in this sub-object indicates that a conversion from the specified NVMe namespace to LUN is desired.

peer

The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM.

This is only populated by GET when the LUN copy is inter-SVM.

Name	Type	Description
_links	_links	
name	string	The local name of the peer SVM. This name is unique among all local and peer SVMs.
uuid	string	The unique identifier of the SVM peer relationship. This is the UUID of the relationship, not the UUID of the peer SVM itself.

error_arguments

Name	Type	Description
code	string	Argument code
message	string	Message argument

error

Error information provided if the asynchronous LUN copy operation fails.

Name	Type	Description
arguments	array[error_arguments]	Message arguments
code	string	Error code
message	string	Error message

progress

Properties related to the progress of an active or recently completed LUN copy.

Name	Type	Description
elapsed	integer	The amount of time that has elapsed since the start of the LUN copy, in seconds.
failure	error	Error information provided if the asynchronous LUN copy operation fails.
percent_complete	integer	The percentage completed of the LUN copy.
state	string	The state of the LUN copy.
volume_snapshot_blocked	boolean	This property reports if volume snapshots are blocked by the LUN copy. This property can be polled to identify when volume snapshots can be resumed after beginning a LUN copy.

destinations

A LUN copy operation in which the containing LUN is the source of the copy.

Name	Type	Description
_links	_links	
max_throughput	integer	The maximum data throughput, in bytes per second, that should be utilized in support of the LUN copy. See property `copy.source.max_throughput` for further details.
name	string	The fully qualified path of the LUN copy destination composed of a "/vol" prefix, the volume name, the optional qtree name, and base name of the LUN.
peer	peer	The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM. This is only populated by GET when the LUN copy is inter-SVM.
progress	progress	Properties related to the progress of an active or recently completed LUN copy.
uuid	string	The unique identifier of the LUN copy destination.

peer

The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM.

Set this in POST to specify the source SVM for an inter-SVM LUN copy. Only populated by GET when the LUN copy is inter-SVM.

Name	Type	Description
_links	_links	
name	string	The local name of the peer SVM. This name is unique among all local and peer SVMs.

Name	Type	Description
uuid	string	The unique identifier of the SVM peer relationship. This is the UUID of the relationship, not the UUID of the peer SVM itself.

progress

Properties related to the progress of an active or recently completed LUN copy.

Name	Type	Description
elapsed	integer	The amount of time that has elapsed since the start of the LUN copy, in seconds.
failure	error	Error information provided if the asynchronous LUN copy operation fails.
percent_complete	integer	The percentage completed of the LUN copy.
state	string	The state of the LUN copy. Valid in PATCH when an LUN copy is active. Set to *paused* to pause a LUN copy. Set to *replicating* to resume a paused LUN copy.
volume_snapshot_blocked	boolean	This property reports if volume snapshots are blocked by the LUN copy. This property can be polled to identify when volume snapshots can be resumed after beginning a LUN copy.

source

The source LUN of a LUN copy operation in which the containing LUN is the destination of the copy.

Valid in POST except when creating a LUN clone. A LUN copy request cannot be combined with setting any other LUN properties except the destination location. All other properties of the destination LUN come from the source LUN.

Name	Type	Description
_links	_links	

Name	Type	Description
max_throughput	integer	The maximum data throughput, in bytes per second, that should be utilized in support of the LUN copy. This property can be used to throttle a transfer and limit its impact on the performance of the source and destination nodes. The specified value will be rounded up to the nearest megabyte. If this property is not specified in a POST that begins a LUN copy, throttling is not applied to the data transfer. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. Valid only in a POST that begins a LUN copy or a PATCH when a LUN copy is already in process. • format: int64 • Introduced in: 9.10 • x-nullable: true
name	string	The fully qualified path of the LUN copy source composed of a "/vol" prefix, the volume name, the optional qtree name, and base name of the LUN. Set this property in POST to specify the source for a LUN copy operation.

Name	Type	Description
peer	peer	The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM. Set this in POST to specify the source SVM for an inter-SVM LUN copy. Only populated by GET when the LUN copy is inter-SVM.
progress	progress	Properties related to the progress of an active or recently completed LUN copy.
uuid	string	The unique identifier of the LUN copy source. Set this property in POST to specify the source for a LUN copy operation.

copy

This sub-object applies to LUN copy operations. A LUN can be copied with a POST request that supplies `copy.source` properties.

Copying a LUN is an asynchronous activity begun by a POST request that specifies the source of the copy in the `copy.source` properties. The data for the LUN is then asynchronously copied from the source to the destination. The time required to complete the copy depends on the size of the LUN and the load on the cluster. The `copy` sub-object is populated while a LUN copy is in progress and for two (2) minutes following completion of a copy.

While LUNs are being copied, the status of the LUN copy operations can be obtained using a GET of the source or destination LUN that requests the `copy` properties. If the LUN is the source LUN for one or more copy operations, the `copy.destinations` array is populated in GET. If the containing LUN is the destination LUN for a copy operation, the `copy.source` sub-object is populated in GET. The LUN copy operation can be further modified using a PATCH on the properties on the `copy.source` sub-object of the copy destination LUN.

There is an added computational cost to retrieving property values for `copy`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
destinations	array[destinations]	An array of destination LUNs of LUN copy operations in which the containing LUN is the source of the copy.
source	source	The source LUN of a LUN copy operation in which the containing LUN is the destination of the copy. Valid in POST except when creating a LUN clone. A LUN copy request cannot be combined with setting any other LUN properties except the destination location. All other properties of the destination LUN come from the source LUN.

node

The cluster node that hosts the LUN.

Name	Type	Description
_links	_links	
name	string	
uuid	string	

qtree

The qtree in which the LUN is optionally located. Valid in POST and PATCH.

If properties `name` and `location.qtree.name` and/or `location.qtree.id` are specified in the same request, they must refer to the same qtree.

A PATCH that modifies the qtree of the LUN is considered a rename operation.

Name	Type	Description
_links	_links	
id	integer	The identifier for the qtree, unique within the qtree's volume.
name	string	The name of the qtree.

volume

The volume in which the LUN is located. Valid in POST and PATCH.

If properties `name` and `location.volume.name` and/or `location.volume.uuid` are specified in the same request, they must refer to the same volume.

A PATCH that modifies the volume of the LUN begins an asynchronous LUN movement operation.

Name	Type	Description
_links	_links	
name	string	The name of the volume. This field cannot be specified in a PATCH method.
uuid	string	Unique identifier for the volume. This corresponds to the instance-uuid that is exposed in the CLI and ONTAPI. It does not change due to a volume move. • example: 028baa66-41bd-11e9-81d5-00a0986138f7 • Introduced in: 9.6 • x-nullable: true

location

The location of the LUN within the ONTAP cluster. LUNs support rename and move between volumes. Valid in POST and PATCH.

Name	Type	Description
logical_unit	string	The base name component of the LUN. Valid in POST and PATCH. If properties `name` and `location.logical_unit` are specified in the same request, they must refer to the base name. A PATCH that modifies the base name of the LUN is considered a rename operation.
node	node	The cluster node that hosts the LUN.

Name	Type	Description
qtree	qtree	The qtree in which the LUN is optionally located. Valid in POST and PATCH. If properties `name` and `location.qtree.name` and/or `location.qtree.id` are specified in the same request, they must refer to the same qtree. A PATCH that modifies the qtree of the LUN is considered a rename operation.
volume	volume	The volume in which the LUN is located. Valid in POST and PATCH. If properties `name` and `location.volume.name` and/or `location.volume.uuid` are specified in the same request, they must refer to the same volume. A PATCH that modifies the volume of the LUN begins an asynchronous LUN movement operation.

self_link

Name	Type	Description
self	href	

igroups

Name	Type	Description
_links	self_link	
name	string	The name of the initiator group.
uuid	string	The unique identifier of the initiator group.

initiators

The initiators that are members of the initiator group.

Name	Type	Description
comment	string	A comment available for use by the administrator.
name	string	Name of initiator that is a member of the initiator group.

igroup

The initiator group to which the LUN is mapped.

Name	Type	Description
_links	_links	
comment	string	A comment available for use by the administrator. Valid in POST and PATCH.

Name	Type	Description
igroups	array[igroups]	The existing initiator groups that are members of the group. Optional in POST. This property is mutually exclusive with the *initiators* property during POST. This array contains only the direct children of the initiator group. If the member initiator groups have further nested initiator groups, those are reported in the `igroups` property of the child initiator group. Zero or more nested initiator groups can be supplied when the initiator group is created. The initiator group will act as if it contains the aggregation of all initiators in any nested initiator groups. After creation, nested initiator groups can be added or removed from the initiator group using the `/protocols/san/igroups/{igroup.uuid}/igroups` endpoint. See DELETE /protocols/san/igroups/{igroup.uuid}/igroups/{uuid} for more details.
initiators	array[initiators]	The initiators that are members of the group.
name	string	The name of the initiator group.
os_type	string	The host operating system of the initiator group. All initiators in the group should be hosts of the same operating system.

Name	Type	Description
protocol	string	The protocols supported by the initiator group. This restricts the type of initiators that can be added to the initiator group. Optional in POST; if not supplied, this defaults to *mixed*. The protocol of an initiator group cannot be changed after creation of the group.
uuid	string	The unique identifier of the initiator group.

lun_maps

A LUN map is an association between a LUN and an initiator group.

When a LUN is mapped to an initiator group, the initiator group's initiators are granted access to the LUN. The relationship between a LUN and an initiator group is many LUNs to many initiator groups.

Name	Type	Description
_links	_links	
igroup	igroup	The initiator group to which the LUN is mapped.
logical_unit_number	integer	The logical unit number assigned to the LUN when mapped to the specified initiator group. The number is used to identify the LUN to initiators in the initiator group when communicating through the Fibre Channel Protocol or iSCSI. Optional in POST; if no value is provided, ONTAP assigns the lowest available value. This property is not supported when the *provisioning_options.count* property is 2 or more. • Introduced in: 9.6 • readCreate: 1 • x-nullable: true

iops

The rate of I/O operations observed at the storage object.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

latency

The round trip latency in microseconds observed at the storage object.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

throughput

The rate of throughput bytes per second observed at the storage object.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.

Name	Type	Description
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

metric

Performance numbers, such as IOPS latency and throughput.

Name	Type	Description
_links	_links	
duration	string	The duration over which this sample is calculated. The time durations are represented in the ISO-8601 standard format. Samples can be calculated over the following durations:
iops	iops	The rate of I/O operations observed at the storage object.
latency	latency	The round trip latency in microseconds observed at the storage object.

Name	Type	Description
status	string	Errors associated with the sample. For example, if the aggregation of data over multiple nodes fails, then any partial errors might return "ok" on success or "error" on an internal uncategorized failure. Whenever a sample collection is missed but done at a later time, it is back filled to the previous 15 second timestamp and tagged with "backfilled_data". "Inconsistent_ delta_time" is encountered when the time between two collections is not the same for all nodes. Therefore, the aggregated value might be over or under inflated. "Negative_delta" is returned when an expected monotonically increasing value has decreased in value. "Inconsistent_old_data" is returned when one or more nodes do not have the latest data.
throughput	throughput	The rate of throughput bytes per second observed at the storage object.
timestamp	string	The timestamp of the performance data.

paths

The fully qualified LUN path names involved in the LUN movement.

Name	Type	Description
destination	string	The fully qualified path of the LUN movement destination composed of a "/vol" prefix, the volume name, the optional qtree name, and base name of the LUN.

Name	Type	Description
source	string	The fully qualified path of the LUN movement source composed of a "/vol" prefix, the volume name, the optional qtree name, and base name of the LUN.

error

Error information provided if the asynchronous LUN movement operation fails.

Name	Type	Description
arguments	array[error_arguments]	Message arguments
code	string	Error code
message	string	Error message

progress

Properties related to the progress of an active or recently completed LUN movement.

Name	Type	Description
elapsed	integer	The amount of time that has elapsed since the start of the LUN movement, in seconds.
failure	error	Error information provided if the asynchronous LUN movement operation fails.
percent_complete	integer	The percentage completed of the LUN movement.
state	string	The state of the LUN movement. Valid in PATCH when an LUN movement is active. Set to *paused* to pause a LUN movement. Set to *replicating* to resume a paused LUN movement.

Name	Type	Description
volume_snapshot_blocked	boolean	This property reports if volume snapshots are blocked by the LUN movement. This property can be polled to identify when volume snapshots can be resumed after beginning a LUN movement.

movement

This sub-object applies to LUN movement between volumes. A LUN can be moved to a new volume with a PATCH request that changes either the volume portion of property `name`, `location.volume.uuid`, or `location.volume.name`. If the volume is changed using more than one of these properties, the supplied properties used must refer to the same volume.

Moving a LUN between volumes is an asynchronous activity begun by a PATCH request. The data for the LUN is then asynchronously copied from the source volume to the destination volume. The time required to complete the move depends on the size of the LUN and the load on the cluster. The `movement` sub-object is populated while a LUN movement is in progress and for two (2) minutes following completion of a movement.

While the LUN is being moved, the status of the LUN movement operation can be obtained using a GET for the LUN that requests the `movement` properties. The LUN movement operation can be further modified using a PATCH on the properties on the `movement` sub-object.

There is an added computational cost to retrieving property values for `movement`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
max_throughput	integer	The maximum data throughput, in bytes per second, that should be utilized in support of the LUN movement. This property can be used to throttle a transfer and limit its impact on the performance of the source and destination nodes. The specified value will be rounded up to the nearest megabyte. If this property is not specified in a POST that begins a LUN movement, throttling is not applied to the data transfer. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. This property is valid only in a POST that begins a LUN movement or a PATCH when a LUN movement is already in process. • format: int64 • Introduced in: 9.6 • x-nullable: true
paths	paths	The fully qualified LUN path names involved in the LUN movement.
progress	progress	Properties related to the progress of an active or recently completed LUN movement.

qos_policy

The QoS policy for the volume provisioned to host the LUN. This property is only supported when the request provisions a new volume. This property is mutually exclusive with the LUN granular `qos_policy`. If no `qos_policy` is provided at LUN or volume granularity, a volume granular policy is set based on the `storage_service.name`, which defaults to the most performant service available.

Name	Type	Description
_links	self_link	

Name	Type	Description
name	string	The QoS policy group name. This is mutually exclusive with UUID and other QoS attributes during POST and PATCH.
uuid	string	The QoS policy group UUID. This is mutually exclusive with name and other QoS attributes during POST and PATCH.

snapshot_policy_reference

This is a reference to the snapshot policy.

Name	Type	Description
_links	_links	
name	string	
uuid	string	

storage_service

Determines the placement of the LUN based on the value specified. This property is only supported for regular and vvol LUNs. Valid in POST.

object_stores

tiering

The tiering placement and policy definitions for the volume provisioned to host the LUN. This property is only supported when the request provisions a new volume.

Name	Type	Description
object_stores	array[object_stores]	Object stores to use. Used for placement.

Name	Type	Description
policy	string	Policy that determines whether the user data blocks of a volume in a FabricPool will be tiered to the cloud store when they become cold. FabricPool combines flash (performance tier) with a cloud store into a single aggregate. Temperature of a volume block increases if it is accessed frequently and decreases when it is not. Valid in POST or PATCH. all ‐ Allows tiering of both snapshots and active file system user data to the cloud store as soon as possible by ignoring the temperature on the volume blocks. auto ‐ Allows tiering of both snapshot and active file system user data to the cloud store none ‐ Volume blocks are not be tiered to the cloud store. snapshot_only ‐ Allows tiering of only the volume snapshots not associated with the active file system. The default tiering policy is "snapshot-only" for a FlexVol volume and "none" for a FlexGroup volume. The default minimum cooling period for the "snapshot-only" tiering policy is 2 days and for the "auto" tiering policy it is 31 days.

provisioning_options

Options that are applied to the operation.

Name	Type	Description
storage_service	storage_service	Determines the placement of the LUN based on the value specified. This property is only supported for regular and vvol LUNs. Valid in POST.

qos_policy

The QoS policy for the LUN. Both traditional and adaptive QoS policies are supported. If both property `qos_policy.uuid` and `qos_policy.name` are specified in the same request, they must refer to the same QoS policy. To remove the QoS policy from a LUN, leaving it with no QoS policy, set the property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.

Name	Type	Description
_links	_links	
name	string	The name of the QoS policy. To remove the QoS policy from a LUN, leaving it with no QoS policy, set this property to an empty string ("") in a PATCH request. Valid in POST and PATCH.
uuid	string	The unique identifier of the QoS policy. Valid in POST and PATCH.

guarantee

Properties that request and report the space guarantee for the LUN.

Name	Type	Description
requested	boolean	The requested space reservation policy for the LUN. If *true*, a space reservation is requested for the LUN; if *false*, the LUN is thin provisioned. Guaranteeing a space reservation request for a LUN requires that the volume in which the LUN resides is also space reserved and that the fractional reserve for the volume is 100%. Valid in POST and PATCH. This property is caller settable as described above.

Name	Type	Description
reserved	boolean	Reports if the LUN is space guaranteed. If *true*, a space guarantee is requested and the containing volume and aggregate support the request. If *false*, a space guarantee is not requested or a space guarantee is requested and either the containing volume or aggregate do not support the request.

space

The storage space related properties of the LUN.

Name	Type	Description
efficiency_ratio	number	The storage efficiency ratio of the LUN without snapshots. (Logical Used / Used) This property is not available on the LUN object in the REST API and is not reported for GET requests. See the containing volume object for this information.
guarantee	guarantee	Properties that request and report the space guarantee for the LUN.
physical_used	integer	The number of bytes consumed on the disk by the LUN, excluding snapshots. This property is not available on the LUN object in the REST API and is not reported for GET requests. See the containing volume object for this information.
physical_used_by_snapshots	integer	The number of bytes consumed on the disk by the LUN's snapshots. This property is not available on the LUN object in the REST API and is not reported for GET requests. See the containing volume object for this information.

Name	Type	Description
scsi_thin_provisioning_support_enabled	boolean	To leverage the benefits of SCSI thin provisioning, it must be supported by your host. SCSI thin provisioning uses the Logical Block Provisioning feature as defined in the SCSI SBC-3 standard. Only hosts that support this standard can use SCSI thin provisioning in ONTAP. When you disable SCSI thin provisioning support in ONTAP, you turn off the following SCSI thin provisioning features: • Unmapping and reporting space usage for space reclamation • Reporting resource exhaustion errors The value of this property is not propagated to the destination when a LUN is cloned as a new LUN or copied; it is reset to false. The value of this property is maintained from the destination LUN when a LUN is overwritten as a clone. Valid in POST and PATCH. • Default value: 1 • Introduced in: 9.10 • x-nullable: true

Name	Type	Description
size	integer	The total provisioned size of the LUN. The LUN size can be increased but not decreased using the REST interface. The maximum and minimum sizes listed here are the absolute maximum and absolute minimum sizes, in bytes. The actual minimum and maximum sizes vary depending on the ONTAP version, ONTAP platform and the available space in the containing volume and aggregate. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. • example: 1073741824 • format: int64 • Max value: 140737488355328 • Min value: 4096 • Introduced in: 9.6 • x-nullable: true

Name	Type	Description
used	integer	The amount of space consumed by the main data stream of the LUN. This value is the total space consumed in the volume by the LUN, including filesystem overhead, but excluding prefix and suffix streams. Due to internal filesystem overhead and the many ways SAN filesystems and applications utilize blocks within a LUN, this value does not necessarily reflect actual consumption/availability from the perspective of the filesystem or application. Without specific knowledge of how the LUN blocks are utilized outside of ONTAP, this property should not be used as an indicator for an out-of-space condition. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. • format: int64 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true

iops_raw

The number of I/O operations observed at the storage object. This can be used along with delta time to calculate the rate of I/O operations per unit of time.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.

Name	Type	Description
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

latency_raw

The raw latency in microseconds observed at the storage object. This can be divided by the raw IOPS value to calculate the average latency per I/O operation.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

throughput_raw

Throughput bytes observed at the storage object. This can be used along with delta time to calculate the rate of throughput bytes per unit of time.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.

Name	Type	Description
write	integer	Performance metric for write I/O operations.

statistics

These are raw performance numbers, such as IOPS latency and throughput. These numbers are aggregated across all nodes in the cluster and increase with the uptime of the cluster.

Name	Type	Description
iops_raw	iops_raw	The number of I/O operations observed at the storage object. This can be used along with delta time to calculate the rate of I/O operations per unit of time.
latency_raw	latency_raw	The raw latency in microseconds observed at the storage object. This can be divided by the raw IOPS value to calculate the average latency per I/O operation.
status	string	Errors associated with the sample. For example, if the aggregation of data over multiple nodes fails, then any partial errors might return "ok" on success or "error" on an internal uncategorized failure. Whenever a sample collection is missed but done at a later time, it is back filled to the previous 15 second timestamp and tagged with "backfilled_data". "Inconsistent_ delta_time" is encountered when the time between two collections is not the same for all nodes. Therefore, the aggregated value might be over or under inflated. "Negative_delta" is returned when an expected monotonically increasing value has decreased in value. "Inconsistent_old_data" is returned when one or more nodes do not have the latest data.

Name	Type	Description
throughput_raw	throughput_raw	Throughput bytes observed at the storage object. This can be used along with delta time to calculate the rate of throughput bytes per unit of time.
timestamp	string	The timestamp of the performance data.

status

Status information about the LUN.

Name	Type	Description
container_state	string	The state of the volume and aggregate that contain the LUN. LUNs are only available when their containers are available.
mapped	boolean	Reports if the LUN is mapped to one or more initiator groups. There is an added computational cost to retrieving this property's value. It is not populated for a GET request unless it is explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
read_only	boolean	Reports if the LUN allows only read access.
state	string	The state of the LUN. Normal states for a LUN are *online* and *offline*. Other states indicate errors.

svm

The SVM in which the LUN is located.

Name	Type	Description
_links	_links	

Name	Type	Description
name	string	The name of the SVM. This field cannot be specified in a PATCH method.
uuid	string	The unique identifier of the SVM. This field cannot be specified in a PATCH method.

partner

The LUN partner that this LUN is bound to. If this LUN is a `vvol` class LUN, the partner is a `protocol_endpoint` class LUN.

Name	Type	Description
_links	_links	
name	string	The name of the partner LUN.
uuid	string	The unique identifier of the partner LUN.

bindings

A vVol binding with which the LUN is associated.

Name	Type	Description
_links	_links	

Name	Type	Description
id	integer	The ONTAP internal identifier assigned to the vVol binding. The bind identifier is unique amongst all class `vvol` LUNs bound to the same class `protocol_endpoint` LUN. This property was included in early releases of the REST API for vVols and is maintained for backward compatibility. See the `secondary_id` property, which replaces `id`. • example: 1 • readOnly: 1 • x-ntap-deprecated: 9.13.1 • Introduced in: 9.10 • x-nullable: true
partner	partner	The LUN partner that this LUN is bound to. If this LUN is a `vvol` class LUN, the partner is a `protocol_endpoint` class LUN.
secondary_id	string	The identifier assigned to the vVol binding, known as the secondary LUN ID. The identifier is unique amongst all class `vvol` LUNs bound to the same class `protocol_endpoint` LUN. The format for a secondary LUN ID is 16 hexadecimal digits (zero-filled) followed by a lower case "h".

vvol

A VMware virtual volume (vVol) binding is an association between a LUN of class `protocol_endpoint` and a LUN of class `vvol`. Class `protocol_endpoint` LUNs are mapped to igroups and granted access using the same configuration as class `regular` LUNs. When a class `vvol` LUN is bound to a mapped class `protocol_endpoint` LUN, VMware can access the class `vvol` LUN through the class `protocol_endpoint` LUN mapping.

See DELETE /protocols/san/vvol-bindings to learn more about deleting vVol bindings.

There is an added computational cost to retrieving property values for `vvol`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
bindings	array[bindings]	Bindings between the LUN, which must be of class `protocol_endpoint` or `vvol`, and LUNs of the opposite class. A class `vvol` LUN must be bound to a class `protocol_endpoint` LUN in order to be accessed. Class `protocol_endpoint` and `vvol` LUNs allow many-to-many bindings. A LUN of one class is allowed to be bound to zero or more LUNs of the opposite class. The binding between any two specific LUNs is reference counted. When a binding is created that already exists, the binding count is incremented. When a binding is deleted, the binding count is decremented, but the LUNs remain bound if the resultant reference count is greater than zero. When the binding count reaches zero, the binding is destroyed. The bindings array contains LUNs of the opposite class of the containing LUN object. There is an added computational cost to retrieving property values for `vvol.bindings`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
is_bound	boolean	Reports if the LUN is part of a VMware virtual volume (vVol) bind relationship. This is `true` if the LUN is of class `protocol_endpoint` or `vvol` and has one or more bindings to a LUN of the opposite class. This is false if the LUN is of class `regular` or unbound.

returned_error

Name	Type	Description
arguments	array[error_arguments]	Message arguments
code	string	Error code
message	string	Error message
target	string	The target parameter that caused the error.

Update an existing LUN

PATCH `/storage/luns/{uuid}`

Introduced In: 9.6

Updates an existing LUN in one of several ways:

- Updates the properties of a LUN.
- Writes data to a LUN. LUN data write requests are distinguished by the header entry `Content-Type: multipart/form-data`. When this header entry is provided, query parameter `data.offset` is required and used to specify the location within the LUN at which to write the data; no other query parameters are allowed. The request body must be `multipart/form-data` content with exactly one form entry containing the data to write. The content type entry of the form data is ignored and always treated as `application/octet-stream`. Writes are limited to one megabyte (1MB) per request.
- Overwrites the contents of a LUN as a clone of another.
- Begins the movement of a LUN between volumes. PATCH can also pause and resume the movement of a LUN between volumes that is already in active.

Related ONTAP commands

- `lun copy modify`

- `lun copy pause`
- `lun copy resume`
- `lun modify`
- `lun move-in-volume`
- `lun move modify`
- `lun move pause`
- `lun move resume`
- `lun move start`
- `lun resize`
- `volume file clone autodelete`

Learn more

- DOC /storage/luns

Parameters

Name	Type	In	Required	Description
uuid	string	path	True	The unique identifier of the LUN to retrieve.

Name	Type	In	Required	Description
data.offset	integer	query	False	The offset, in bytes, at which to begin writing LUN data. LUN data write requests are distinguished by the header entry `Content-Type: multipart/form-data`. When this header entry is provided, query parameter `data.offset` is required and used to specify the location within the LUN at which to write the data; no other query parameters are allowed. The request body must be `multipart/form-data` content with exactly one form entry containing the data to write. The content type entry of the form data is ignored and always treated as `application/octet-stream`. Writes are limited to one megabyte (1MB) per request. • format: int64 • Min value: 0 • Introduced in: 9.11

Name	Type	In	Required	Description
return_timeout	integer	query	False	The number of seconds to allow the call to execute before returning. When doing a POST, PATCH, or DELETE operation on a single record, the default is 0 seconds. This means that if an asynchronous operation is started, the server immediately returns HTTP code 202 (Accepted) along with a link to the job. If a non-zero value is specified for POST, PATCH, or DELETE operations, ONTAP waits that length of time to see if the job completes so it can return something other than 202. • Min value: 0 • Default value: 0 • Max value: 120

Request Body

Name	Type	Description
auto_delete	boolean	This property marks the LUN for auto deletion when the volume containing the LUN runs out of space. This is most commonly set on LUN clones. When set to *true*, the LUN becomes eligible for automatic deletion when the volume runs out of space. Auto deletion only occurs when the volume containing the LUN is also configured for auto deletion and free space in the volume decreases below a particular threshold. This property is optional in POST and PATCH. The default value for a new LUN is *false*. There is an added computational cost to retrieving this property's value. It is not populated for a GET request unless it is explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
clone	clone	This sub-object is used in POST to create a new LUN as a clone of an existing LUN, or PATCH to overwrite an existing LUN as a clone of another. Setting a property in this sub-object indicates that a LUN clone is desired. Consider the following other properties when cloning a LUN: `auto_delete`, `qos_policy`, `space.guarantee.requested` and `space.scsi_thin_provisioning_support_enabled`. When used in a PATCH, the patched LUN's data is over-written as a clone of the source and the following properties are preserved from the patched LUN unless otherwise specified as part of the PATCH: `class`, `auto_delete`, `lun_maps`, `serial_number`, `status.state`, and `uuid`. Persistent reservations for the patched LUN are also preserved.
comment	string	A configurable comment available for use by the administrator. Valid in POST and PATCH.
consistency_group	consistency_group	The LUN's consistency group. This property is populated for LUNs that are members of a consistency group. If the LUN is a member of a child consistency group, the parent consistency group is reported. A LUN's consistency group is the consistency group of its containing volume.

Name	Type	Description
copy	copy	This sub-object applies to LUN copy operations. A LUN can be copied with a POST request that supplies `copy.source` properties. Copying a LUN is an asynchronous activity begun by a POST request that specifies the source of the copy in the `copy.source` properties. The data for the LUN is then asynchronously copied from the source to the destination. The time required to complete the copy depends on the size of the LUN and the load on the cluster. The `copy` sub-object is populated while a LUN copy is in progress and for two (2) minutes following completion of a copy. While LUNs are being copied, the status of the LUN copy operations can be obtained using a GET of the source or destination LUN that requests the `copy` properties. If the LUN is the source LUN for one or more copy operations, the `copy.destinations` array is populated in GET. If the containing LUN is the destination LUN for a copy operation, the `copy.source` sub-object is populated in GET. The LUN copy operation can be further modified using a PATCH on the properties on the `copy.source` sub-object of the copy destination LUN. There is an added computational cost to retrieving property values for `copy`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
create_time	string	The time the LUN was created.

Name	Type	Description
enabled	boolean	The enabled state of the LUN. LUNs can be disabled to prevent access to the LUN. Certain error conditions also cause the LUN to become disabled. If the LUN is disabled, check the `status.state` property to determine if the LUN is administratively disabled (*offline*) or has become disabled as a result of an error. A LUN in an error condition can be brought online by setting the `enabled` property to *true* or brought administratively offline by setting the `enabled` property to *false*. Upon creation, a LUN is enabled by default. Valid in PATCH.
location	location	The location of the LUN within the ONTAP cluster. LUNs support rename and move between volumes. Valid in POST and PATCH. • Introduced in: 9.6

Name	Type	Description
movement	movement	This sub-object applies to LUN movement between volumes. A LUN can be moved to a new volume with a PATCH request that changes either the volume portion of property `name`, `location.volume.uuid`, or `location.volume.name`. If the volume is changed using more than one of these properties, the supplied properties used must refer to the same volume. Moving a LUN between volumes is an asynchronous activity begun by a PATCH request. The data for the LUN is then asynchronously copied from the source volume to the destination volume. The time required to complete the move depends on the size of the LUN and the load on the cluster. The `movement` sub-object is populated while a LUN movement is in progress and for two (2) minutes following completion of a movement. While the LUN is being moved, the status of the LUN movement operation can be obtained using a GET for the LUN that requests the `movement` properties. The LUN movement operation can be further modified using a PATCH on the properties on the `movement` sub-object. There is an added computational cost to retrieving property values for `movement`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
name	string	The name of the LUN. Valid in POST and PATCH. A LUN is located within a volume. Optionally, it can be located within a qtree in a volume. LUN names are paths of the form "/vol/<volume>[/<qtree>]/<lun>" where the qtree name is optional. A PATCH that modifies the qtree and/or base name portion of the LUN path is considered a rename operation. A PATCH that modifies the volume portion of the LUN path begins an asynchronous LUN movement operation.
qos_policy	qos_policy	The QoS policy for the LUN. Both traditional and adaptive QoS policies are supported. If both property `qos_policy.uuid` and `qos_policy.name` are specified in the same request, they must refer to the same QoS policy. To remove the QoS policy from a LUN, leaving it with no QoS policy, set the property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.
serial_number	string	The LUN serial number. The serial number is generated by ONTAP when the LUN is created. • maxLength: 12 • minLength: 12 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true

Name	Type	Description
serial_number_hex	string	The LUN serial number encoded in hexadecimal format. The serial number is generated by ONTAP when the LUN is created. • readOnly: 1 • Introduced in: 9.17 • x-nullable: true
space	space	The storage space related properties of the LUN.
status	status	Status information about the LUN.
uuid	string	The unique identifier of the LUN. The UUID is generated by ONTAP when the LUN is created. • example: 1cd8a442-86d1-11e0-ae1c-123478563412 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true

Name	Type	Description
vvol	vvol	A VMware virtual volume (vVol) binding is an association between a LUN of class `protocol_endpoint` and a LUN of class `vvol`. Class `protocol_endpoint` LUNs are mapped to igroups and granted access using the same configuration as class `regular` LUNs. When a class `vvol` LUN is bound to a mapped class `protocol_endpoint` LUN, VMware can access the class `vvol` LUN through the class `protocol_endpoint` LUN mapping. See DELETE /protocols/san/vvol-bindings to learn more about deleting vVol bindings. There is an added computational cost to retrieving property values for `vvol`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Example request


```
{
  "clone": {
    "source": {
      "name": "/vol/volume1/lun1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    }
  },
  "comment": "string",
  "consistency_group": {
    "name": "cg1",
    "uuid": "4abc2317-4332-9d37-93a0-20bd29c22df0"
  },
  "copy": {
    "destinations": [
      {
        "max_throughput": 0,
        "name": "/vol/vol1/lun1",
        "peer": {
          "name": "peer1",
          "uuid": "4204cf77-4c82-9bdb-5644-b5a841c097a9"
        },
        "progress": {
          "elapsed": 0,
          "percent_complete": 0,
          "state": "string"
        },
        "uuid": "1bc327d5-4654-5284-a116-f182282240b4"
      }
    ],
    "source": {
      "peer": {
        "name": "peer1",
        "uuid": "4204cf77-4c82-9bdb-5644-b5a841c097a9"
      },
      "progress": {
        "elapsed": 0,
        "percent_complete": 0,
        "state": "string"
      }
    }
  },
  "create_time": "2018-06-04 15:00:00 -0400",
  "location": {
    "logical_unit": "lun1",
```

```
    "node": {
      "name": "node1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    },
    "qtree": {
      "id": 1,
      "name": "qt1"
    },
    "volume": {
      "name": "volume1",
      "uuid": "028baa66-41bd-11e9-81d5-00a0986138f7"
    }
  },
  "movement": {
    "paths": {
      "destination": "/vol/vol1/lun1",
      "source": "/vol/vol2/lun2"
    },
    "progress": {
      "elapsed": 0,
      "percent_complete": 0,
      "state": "string"
    }
  },
  "name": "/vol/volume1/qtree1/lun1",
  "qos_policy": {
    "name": "qos1",
    "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
  },
  "serial_number": "string",
  "serial_number_hex": "string",
  "space": {
    "efficiency_ratio": 2.5,
    "physical_used": 1073741824,
    "physical_used_by_snapshots": 1073741824,
    "size": 1073741824,
    "used": 0
  },
  "status": {
    "container_state": "string",
    "state": "online"
  },
  "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412",
  "vvol": {
    "bindings": [
      {
```

```
        "id": 1,
        "partner": {
          "name": "/vol/vol1/lun1",
          "uuid": "4ea7a442-86d1-11e0-ae1c-123478563412"
        },
        "secondary_id": "0000D20000010000h"
      }
    ]
  }
}
```


Response

```
Status: 200, Ok
```

Response

```
Status: 202, Accepted
```

Error

```
Status: Default
```

ONTAP Error Response Codes

Error Code	Description
917927	The specified volume was not found.
918236	The specified `location.volume.uuid` and `location.volume.name` do not refer to the same volume.
1254196	A LUN already exists at the specified path.
1254226	Cannot patch a LUN's size to its current size.
5242927	The specified qtree was not found.
5242950	The specified `location.qtree.id` and `location.qtree.name` do not refer to the same qtree.
5374124	The specified LUN size is too small.
5374125	The specified LUN size is too large.
5374127	The specified LUN name is invalid.

Error Code	Description
5374130	An invalid size value was provided.
5374241	A size value with invalid units was provided.
5374480	Modifying the LUN is not allowed because it is in a foreign LUN import relationship.
5374858	The volume specified by `name` is not the same as that specified by `location.volume`.
5374860	The qtree specified by `name` is not the same as that specified by `location.qtree`.
5374861	The LUN base name specified by `name` is not the same as that specified by `location.logical_unit`.
5374864	An error occurred after successfully overwriting data for the LUN as a clone. Some properties were not modified.
5374865	The LUN's aggregate is offline. The aggregate must be online to modify or remove the LUN.
5374866	The LUN's volume is offline. The volume must be online to modify or remove the LUN.
5374874	The specified `clone.source.uuid` and `clone.source.name` do not refer to the same LUN.
5374875	The specified LUN was not found. This can apply to `clone.source` or the target LUN. The `target` property of the error object identifies the property.
5374876	The specified LUN was not found. This can apply to `clone.source` or the target LUN. The `target` property of the error object identifies the property.
5374885	An error occurred after successfully modifying some of the properties of the LUN. Some properties were not modified.
5374889	An invalid value was specified for `movement.progress.state`. Active LUN movement operations can be PATCHed to only *paused* or *replicating*.
5374892	An attempt was made to reduce the size of a LUN.
5374904	The destination volume is not online.
5375059	An unsuitable QoS policy was specified.
5376461	The specified LUN name is invalid.
5376462	The specified LUN name is too long.
5376463	The snapshot portion of the specified LUN name is too long.

Error Code	Description
5376466	An attempt was made to rename a LUN to a snapshot name.
5376467	An attempt was made to rename a primary LUN to a secondary name.
5376468	An attempt was made to rename a LUN to a reserved name.
5376470	The property cannot be set during the LUN modify operation on this platform.
7018877	Maximum combined total (50) of file and LUN copy and move operations reached. When one or more of the operations has completed, try the command again.
7018919	A copy or move job exists with the same destination LUN.
8454243	The specified QoS policy does not exist while modifying a LUN.
13565952	The LUN clone request failed.

Also see the table of common errors in the Response body overview section of this documentation.

Definitions

See Definitions

href

Name	Type	Description
href	string	

_links

attributes

A name/value pair optionally stored with the LUN. Attributes are available to callers to persist small amounts of application-specific metadata. They are in no way interpreted by ONTAP.

Attribute names and values must be at least one byte and no more than 4091 bytes in length. The sum of the name and value lengths must be no more than 4092 bytes.

Optional in POST.

source

The source LUN for a LUN clone operation. This can be specified using property `clone.source.uuid` or `clone.source.name`. If both properties are supplied, they must refer to the same LUN.

Valid in POST to create a new LUN as a clone of the source.

Valid in PATCH to overwrite an existing LUN's data as a clone of another.

Name	Type	Description
name	string	The name of the clone source LUN. A LUN is located within a volume. Optionally, it can be located within a qtree in a volume. LUN names are paths of the form "/vol/<volume>[/<qtree>]/<namespace>" where the qtree name is optional. Valid in POST and PATCH.
uuid	string	The unique identifier of the clone source LUN. Valid in POST and PATCH.

clone

This sub-object is used in POST to create a new LUN as a clone of an existing LUN, or PATCH to overwrite an existing LUN as a clone of another. Setting a property in this sub-object indicates that a LUN clone is desired. Consider the following other properties when cloning a LUN: `auto_delete`,

`qos_policy`, `space.guarantee.requested` and `space.scsi_thin_provisioning_support_enabled`.

When used in a PATCH, the patched LUN's data is over-written as a clone of the source and the following properties are preserved from the patched LUN unless otherwise specified as part of the PATCH: `class`, `auto_delete`, `lun_maps`, `serial_number`, `status.state`, and `uuid`.

Persistent reservations for the patched LUN are also preserved.

Name	Type	Description
source	source	The source LUN for a LUN clone operation. This can be specified using property `clone.source.uuid` or `clone.source.name`. If both properties are supplied, they must refer to the same LUN. Valid in POST to create a new LUN as a clone of the source. Valid in PATCH to overwrite an existing LUN's data as a clone of another.

consistency_group

The LUN's consistency group. This property is populated for LUNs that are members of a consistency group. If the LUN is a member of a child consistency group, the parent consistency group is reported. A LUN's consistency group is the consistency group of its containing volume.

Name	Type	Description
name	string	The name of the consistency group.
uuid	string	The unique identifier of the consistency group.

namespace

The source namespace for convert operation. This can be specified using property `convert.namespace.uuid` or `convert.namespace.name`. If both properties are supplied, they must refer to the same NVMe namespace.

Valid in POST. A convert request from NVMe namespace to LUN cannot be combined with setting any other LUN properties. All other properties of the converted LUN come from the source NVMe namespace.

convert

This sub-object is used in POST to convert a valid in-place NVMe namespace to a LUN. Setting a property in this sub-object indicates that a conversion from the specified NVMe namespace to LUN is

desired.

peer

The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM.

This is only populated by GET when the LUN copy is inter-SVM.

Name	Type	Description
name	string	The local name of the peer SVM. This name is unique among all local and peer SVMs.
uuid	string	The unique identifier of the SVM peer relationship. This is the UUID of the relationship, not the UUID of the peer SVM itself.

error_arguments

Name	Type	Description
code	string	Argument code
message	string	Message argument

error

Error information provided if the asynchronous LUN copy operation fails.

Name	Type	Description
arguments	array[error_arguments]	Message arguments
code	string	Error code
message	string	Error message

progress

Properties related to the progress of an active or recently completed LUN copy.

Name	Type	Description
elapsed	integer	The amount of time that has elapsed since the start of the LUN copy, in seconds.

Name	Type	Description
percent_complete	integer	The percentage completed of the LUN copy.
state	string	The state of the LUN copy.
volume_snapshot_blocked	boolean	This property reports if volume snapshots are blocked by the LUN copy. This property can be polled to identify when volume snapshots can be resumed after beginning a LUN copy.

destinations

A LUN copy operation in which the containing LUN is the source of the copy.

Name	Type	Description
max_throughput	integer	The maximum data throughput, in bytes per second, that should be utilized in support of the LUN copy. See property `copy.source.max_throughput` for further details.
name	string	The fully qualified path of the LUN copy destination composed of a "/vol" prefix, the volume name, the optional qtree name, and base name of the LUN.
peer	peer	The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM. This is only populated by GET when the LUN copy is inter-SVM.
progress	progress	Properties related to the progress of an active or recently completed LUN copy.
uuid	string	The unique identifier of the LUN copy destination.

peer

The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM.

Set this in POST to specify the source SVM for an inter-SVM LUN copy. Only populated by GET when the LUN copy is inter-SVM.

Name	Type	Description
name	string	The local name of the peer SVM. This name is unique among all local and peer SVMs.
uuid	string	The unique identifier of the SVM peer relationship. This is the UUID of the relationship, not the UUID of the peer SVM itself.

progress

Properties related to the progress of an active or recently completed LUN copy.

Name	Type	Description
elapsed	integer	The amount of time that has elapsed since the start of the LUN copy, in seconds.
percent_complete	integer	The percentage completed of the LUN copy.
state	string	The state of the LUN copy. Valid in PATCH when an LUN copy is active. Set to *paused* to pause a LUN copy. Set to *replicating* to resume a paused LUN copy.
volume_snapshot_blocked	boolean	This property reports if volume snapshots are blocked by the LUN copy. This property can be polled to identify when volume snapshots can be resumed after beginning a LUN copy.

source

The source LUN of a LUN copy operation in which the containing LUN is the destination of the copy.

Valid in POST except when creating a LUN clone. A LUN copy request cannot be combined with setting

any other LUN properties except the destination location. All other properties of the destination LUN come from the source LUN.

Name	Type	Description
max_throughput	integer	The maximum data throughput, in bytes per second, that should be utilized in support of the LUN copy. This property can be used to throttle a transfer and limit its impact on the performance of the source and destination nodes. The specified value will be rounded up to the nearest megabyte. If this property is not specified in a POST that begins a LUN copy, throttling is not applied to the data transfer. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. Valid only in a POST that begins a LUN copy or a PATCH when a LUN copy is already in process. • format: int64 • Introduced in: 9.10 • x-nullable: true
peer	peer	The SVM peer relationship object for an inter-SVM LUN copy operation. The peer SVM in the relationship is the source SVM and the local SVM is the destination SVM. Set this in POST to specify the source SVM for an inter-SVM LUN copy. Only populated by GET when the LUN copy is inter-SVM.
progress	progress	Properties related to the progress of an active or recently completed LUN copy.

copy

This sub-object applies to LUN copy operations. A LUN can be copied with a POST request that supplies `copy.source` properties.

Copying a LUN is an asynchronous activity begun by a POST request that specifies the source of the copy in the `copy.source` properties. The data for the LUN is then asynchronously copied from the source to the destination. The time required to complete the copy depends on the size of the LUN and the load on the cluster. The `copy` sub-object is populated while a LUN copy is in progress and for two (2) minutes following completion of a copy.

While LUNs are being copied, the status of the LUN copy operations can be obtained using a GET of the source or destination LUN that requests the `copy` properties. If the LUN is the source LUN for one or more copy operations, the `copy.destinations` array is populated in GET. If the containing LUN is the destination LUN for a copy operation, the `copy.source` sub-object is populated in GET. The LUN copy operation can be further modified using a PATCH on the properties on the `copy.source` sub-object of the copy destination LUN.

There is an added computational cost to retrieving property values for `copy`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
destinations	array[destinations]	An array of destination LUNs of LUN copy operations in which the containing LUN is the source of the copy.
source	source	The source LUN of a LUN copy operation in which the containing LUN is the destination of the copy. Valid in POST except when creating a LUN clone. A LUN copy request cannot be combined with setting any other LUN properties except the destination location. All other properties of the destination LUN come from the source LUN.

node

The cluster node that hosts the LUN.

Name	Type	Description
name	string	
uuid	string	

qtree

The qtree in which the LUN is optionally located. Valid in POST and PATCH.

If properties `name` and `location.qtree.name` and/or `location.qtree.id` are specified in the same request, they must refer to the same qtree.

A PATCH that modifies the qtree of the LUN is considered a rename operation.

Name	Type	Description
id	integer	The identifier for the qtree, unique within the qtree's volume.
name	string	The name of the qtree.

volume

The volume in which the LUN is located. Valid in POST and PATCH.

If properties `name` and `location.volume.name` and/or `location.volume.uuid` are specified in the same request, they must refer to the same volume.

A PATCH that modifies the volume of the LUN begins an asynchronous LUN movement operation.

Name	Type	Description
name	string	The name of the volume. This field cannot be specified in a PATCH method.
uuid	string	Unique identifier for the volume. This corresponds to the instance-uuid that is exposed in the CLI and ONTAPI. It does not change due to a volume move. • example: 028baa66-41bd-11e9-81d5-00a0986138f7 • Introduced in: 9.6 • x-nullable: true

location

The location of the LUN within the ONTAP cluster. LUNs support rename and move between volumes. Valid in POST and PATCH.

Name	Type	Description
logical_unit	string	The base name component of the LUN. Valid in POST and PATCH. If properties `name` and `location.logical_unit` are specified in the same request, they must refer to the base name. A PATCH that modifies the base name of the LUN is considered a rename operation.
node	node	The cluster node that hosts the LUN.
qtree	qtree	The qtree in which the LUN is optionally located. Valid in POST and PATCH. If properties `name` and `location.qtree.name` and/or `location.qtree.id` are specified in the same request, they must refer to the same qtree. A PATCH that modifies the qtree of the LUN is considered a rename operation.
volume	volume	The volume in which the LUN is located. Valid in POST and PATCH. If properties `name` and `location.volume.name` and/or `location.volume.uuid` are specified in the same request, they must refer to the same volume. A PATCH that modifies the volume of the LUN begins an asynchronous LUN movement operation.

self_link

igroups

Name	Type	Description
name	string	The name of the initiator group.
uuid	string	The unique identifier of the initiator group.

initiators

The initiators that are members of the initiator group.

Name	Type	Description
comment	string	A comment available for use by the administrator.
name	string	Name of initiator that is a member of the initiator group.

igroup

The initiator group to which the LUN is mapped.

lun_maps

A LUN map is an association between a LUN and an initiator group.

When a LUN is mapped to an initiator group, the initiator group's initiators are granted access to the LUN. The relationship between a LUN and an initiator group is many LUNs to many initiator groups.

iops

The rate of I/O operations observed at the storage object.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

latency

The round trip latency in microseconds observed at the storage object.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

throughput

The rate of throughput bytes per second observed at the storage object.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

metric

Performance numbers, such as IOPS latency and throughput.

Name	Type	Description
duration	string	The duration over which this sample is calculated. The time durations are represented in the ISO-8601 standard format. Samples can be calculated over the following durations:
iops	iops	The rate of I/O operations observed at the storage object.
latency	latency	The round trip latency in microseconds observed at the storage object.
status	string	Errors associated with the sample. For example, if the aggregation of data over multiple nodes fails, then any partial errors might return "ok" on success or "error" on an internal uncategorized failure. Whenever a sample collection is missed but done at a later time, it is back filled to the previous 15 second timestamp and tagged with "backfilled_data". "Inconsistent_ delta_time" is encountered when the time between two collections is not the same for all nodes. Therefore, the aggregated value might be over or under inflated. "Negative_delta" is returned when an expected monotonically increasing value has decreased in value. "Inconsistent_old_data" is returned when one or more nodes do not have the latest data.
throughput	throughput	The rate of throughput bytes per second observed at the storage object.
timestamp	string	The timestamp of the performance data.

paths

The fully qualified LUN path names involved in the LUN movement.

Name	Type	Description
destination	string	The fully qualified path of the LUN movement destination composed of a "/vol" prefix, the volume name, the optional qtree name, and base name of the LUN.
source	string	The fully qualified path of the LUN movement source composed of a "/vol" prefix, the volume name, the optional qtree name, and base name of the LUN.

error

Error information provided if the asynchronous LUN movement operation fails.

Name	Type	Description
arguments	array[error_arguments]	Message arguments
code	string	Error code
message	string	Error message

progress

Properties related to the progress of an active or recently completed LUN movement.

Name	Type	Description
elapsed	integer	The amount of time that has elapsed since the start of the LUN movement, in seconds.
percent_complete	integer	The percentage completed of the LUN movement.
state	string	The state of the LUN movement. Valid in PATCH when an LUN movement is active. Set to *paused* to pause a LUN movement. Set to *replicating* to resume a paused LUN movement.

Name	Type	Description
volume_snapshot_blocked	boolean	This property reports if volume snapshots are blocked by the LUN movement. This property can be polled to identify when volume snapshots can be resumed after beginning a LUN movement.

movement

This sub-object applies to LUN movement between volumes. A LUN can be moved to a new volume with a PATCH request that changes either the volume portion of property `name`, `location.volume.uuid`, or `location.volume.name`. If the volume is changed using more than one of these properties, the supplied properties used must refer to the same volume.

Moving a LUN between volumes is an asynchronous activity begun by a PATCH request. The data for the LUN is then asynchronously copied from the source volume to the destination volume. The time required to complete the move depends on the size of the LUN and the load on the cluster. The `movement` sub-object is populated while a LUN movement is in progress and for two (2) minutes following completion of a movement.

While the LUN is being moved, the status of the LUN movement operation can be obtained using a GET for the LUN that requests the `movement` properties. The LUN movement operation can be further modified using a PATCH on the properties on the `movement` sub-object.

There is an added computational cost to retrieving property values for `movement`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
max_throughput	integer	The maximum data throughput, in bytes per second, that should be utilized in support of the LUN movement. This property can be used to throttle a transfer and limit its impact on the performance of the source and destination nodes. The specified value will be rounded up to the nearest megabyte. If this property is not specified in a POST that begins a LUN movement, throttling is not applied to the data transfer. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. This property is valid only in a POST that begins a LUN movement or a PATCH when a LUN movement is already in process. • format: int64 • Introduced in: 9.6 • x-nullable: true
paths	paths	The fully qualified LUN path names involved in the LUN movement.
progress	progress	Properties related to the progress of an active or recently completed LUN movement.

qos_policy

The QoS policy for the volume provisioned to host the LUN. This property is only supported when the request provisions a new volume. This property is mutually exclusive with the LUN granular `qos_policy`. If no `qos_policy` is provided at LUN or volume granularity, a volume granular policy is set based on the `storage_service.name`, which defaults to the most performant service available.

Name	Type	Description
name	string	The QoS policy group name. This is mutually exclusive with UUID and other QoS attributes during POST and PATCH.
uuid	string	The QoS policy group UUID. This is mutually exclusive with name and other QoS attributes during POST and PATCH.

snapshot_policy_reference

This is a reference to the snapshot policy.

Name	Type	Description
name	string	
uuid	string	

storage_service

Determines the placement of the LUN based on the value specified. This property is only supported for regular and vvol LUNs. Valid in POST.

object_stores

Name	Type	Description
name	string	The name of the object store to use. Used for placement.

tiering

The tiering placement and policy definitions for the volume provisioned to host the LUN. This property is only supported when the request provisions a new volume.

Name	Type	Description
control	string	Storage tiering placement rules for the object.
object_stores	array[object_stores]	Object stores to use. Used for placement.

provisioning_options

Options that are applied to the operation.

Name	Type	Description
storage_service	storage_service	Determines the placement of the LUN based on the value specified. This property is only supported for regular and vvol LUNs. Valid in POST.

qos_policy

The QoS policy for the LUN. Both traditional and adaptive QoS policies are supported. If both property `qos_policy.uuid` and `qos_policy.name` are specified in the same request, they must refer to the same QoS policy. To remove the QoS policy from a LUN, leaving it with no QoS policy, set the property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.

Name	Type	Description
name	string	The name of the QoS policy. To remove the QoS policy from a LUN, leaving it with no QoS policy, set this property to an empty string ("") in a PATCH request. Valid in POST and PATCH.
uuid	string	The unique identifier of the QoS policy. Valid in POST and PATCH.

guarantee

Properties that request and report the space guarantee for the LUN.

Name	Type	Description
requested	boolean	The requested space reservation policy for the LUN. If *true*, a space reservation is requested for the LUN; if *false*, the LUN is thin provisioned. Guaranteeing a space reservation request for a LUN requires that the volume in which the LUN resides is also space reserved and that the fractional reserve for the volume is 100%. Valid in POST and PATCH. This property is caller settable as described above.

Name	Type	Description
reserved	boolean	Reports if the LUN is space guaranteed. If *true*, a space guarantee is requested and the containing volume and aggregate support the request. If *false*, a space guarantee is not requested or a space guarantee is requested and either the containing volume or aggregate do not support the request.

space

The storage space related properties of the LUN.

Name	Type	Description
efficiency_ratio	number	The storage efficiency ratio of the LUN without snapshots. (Logical Used / Used) This property is not available on the LUN object in the REST API and is not reported for GET requests. See the containing volume object for this information.
guarantee	guarantee	Properties that request and report the space guarantee for the LUN.
physical_used	integer	The number of bytes consumed on the disk by the LUN, excluding snapshots. This property is not available on the LUN object in the REST API and is not reported for GET requests. See the containing volume object for this information.
physical_used_by_snapshots	integer	The number of bytes consumed on the disk by the LUN's snapshots. This property is not available on the LUN object in the REST API and is not reported for GET requests. See the containing volume object for this information.

Name	Type	Description
scsi_thin_provisioning_support_enabled	boolean	To leverage the benefits of SCSI thin provisioning, it must be supported by your host. SCSI thin provisioning uses the Logical Block Provisioning feature as defined in the SCSI SBC-3 standard. Only hosts that support this standard can use SCSI thin provisioning in ONTAP. When you disable SCSI thin provisioning support in ONTAP, you turn off the following SCSI thin provisioning features: • Unmapping and reporting space usage for space reclamation • Reporting resource exhaustion errors The value of this property is not propagated to the destination when a LUN is cloned as a new LUN or copied; it is reset to false. The value of this property is maintained from the destination LUN when a LUN is overwritten as a clone. Valid in POST and PATCH. • Default value: 1 • Introduced in: 9.10 • x-nullable: true

Name	Type	Description
size	integer	The total provisioned size of the LUN. The LUN size can be increased but not decreased using the REST interface. The maximum and minimum sizes listed here are the absolute maximum and absolute minimum sizes, in bytes. The actual minimum and maximum sizes vary depending on the ONTAP version, ONTAP platform and the available space in the containing volume and aggregate. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. • example: 1073741824 • format: int64 • Max value: 140737488355328 • Min value: 4096 • Introduced in: 9.6 • x-nullable: true

Name	Type	Description
used	integer	The amount of space consumed by the main data stream of the LUN. This value is the total space consumed in the volume by the LUN, including filesystem overhead, but excluding prefix and suffix streams. Due to internal filesystem overhead and the many ways SAN filesystems and applications utilize blocks within a LUN, this value does not necessarily reflect actual consumption/availability from the perspective of the filesystem or application. Without specific knowledge of how the LUN blocks are utilized outside of ONTAP, this property should not be used as an indicator for an out-of-space condition. For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation. • format: int64 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true

iops_raw

The number of I/O operations observed at the storage object. This can be used along with delta time to calculate the rate of I/O operations per unit of time.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.

Name	Type	Description
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

latency_raw

The raw latency in microseconds observed at the storage object. This can be divided by the raw IOPS value to calculate the average latency per I/O operation.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.
write	integer	Performance metric for write I/O operations.

throughput_raw

Throughput bytes observed at the storage object. This can be used along with delta time to calculate the rate of throughput bytes per unit of time.

Name	Type	Description
other	integer	Performance metric for other I/O operations. Other I/O operations can be metadata operations, such as directory lookups and so on.
read	integer	Performance metric for read I/O operations.
total	integer	Performance metric aggregated over all types of I/O operations.

Name	Type	Description
write	integer	Performance metric for write I/O operations.

statistics

These are raw performance numbers, such as IOPS latency and throughput. These numbers are aggregated across all nodes in the cluster and increase with the uptime of the cluster.

Name	Type	Description
iops_raw	iops_raw	The number of I/O operations observed at the storage object. This can be used along with delta time to calculate the rate of I/O operations per unit of time.
latency_raw	latency_raw	The raw latency in microseconds observed at the storage object. This can be divided by the raw IOPS value to calculate the average latency per I/O operation.
status	string	Errors associated with the sample. For example, if the aggregation of data over multiple nodes fails, then any partial errors might return "ok" on success or "error" on an internal uncategorized failure. Whenever a sample collection is missed but done at a later time, it is back filled to the previous 15 second timestamp and tagged with "backfilled_data". "Inconsistent_ delta_time" is encountered when the time between two collections is not the same for all nodes. Therefore, the aggregated value might be over or under inflated. "Negative_delta" is returned when an expected monotonically increasing value has decreased in value. "Inconsistent_old_data" is returned when one or more nodes do not have the latest data.

Name	Type	Description
throughput_raw	throughput_raw	Throughput bytes observed at the storage object. This can be used along with delta time to calculate the rate of throughput bytes per unit of time.
timestamp	string	The timestamp of the performance data.

status

Status information about the LUN.

Name	Type	Description
container_state	string	The state of the volume and aggregate that contain the LUN. LUNs are only available when their containers are available.
mapped	boolean	Reports if the LUN is mapped to one or more initiator groups. There is an added computational cost to retrieving this property's value. It is not populated for a GET request unless it is explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
read_only	boolean	Reports if the LUN allows only read access.
state	string	The state of the LUN. Normal states for a LUN are *online* and *offline*. Other states indicate errors.

svm

The SVM in which the LUN is located.

Name	Type	Description
name	string	The name of the SVM. This field cannot be specified in a PATCH method.

Name	Type	Description
uuid	string	The unique identifier of the SVM. This field cannot be specified in a PATCH method.

partner

The LUN partner that this LUN is bound to. If this LUN is a `vvol` class LUN, the partner is a `protocol_endpoint` class LUN.

Name	Type	Description
name	string	The name of the partner LUN.
uuid	string	The unique identifier of the partner LUN.

bindings

A vVol binding with which the LUN is associated.

Name	Type	Description
id	integer	The ONTAP internal identifier assigned to the vVol binding. The bind identifier is unique amongst all class `vvol` LUNs bound to the same class `protocol_endpoint` LUN. This property was included in early releases of the REST API for vVols and is maintained for backward compatibility. See the `secondary_id` property, which replaces `id`. • example: 1 • readOnly: 1 • x-ntap-deprecated: 9.13.1 • Introduced in: 9.10 • x-nullable: true
partner	partner	The LUN partner that this LUN is bound to. If this LUN is a `vvol` class LUN, the partner is a `protocol_endpoint` class LUN.

Name	Type	Description
secondary_id	string	The identifier assigned to the vVol binding, known as the secondary LUN ID. The identifier is unique amongst all class `vvol` LUNs bound to the same class `protocol_endpoint` LUN. The format for a secondary LUN ID is 16 hexadecimal digits (zero-filled) followed by a lower case "h".

vvol

A VMware virtual volume (vVol) binding is an association between a LUN of class `protocol_endpoint` and a LUN of class `vvol`. Class `protocol_endpoint` LUNs are mapped to igroups and granted access using the same configuration as class `regular` LUNs. When a class `vvol` LUN is bound to a mapped class `protocol_endpoint` LUN, VMware can access the class `vvol` LUN through the class `protocol_endpoint` LUN mapping.

See DELETE /protocols/san/vvol-bindings to learn more about deleting vVol bindings.

There is an added computational cost to retrieving property values for `vvol`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
bindings	array[bindings]	Bindings between the LUN, which must be of class `protocol_endpoint` or `vvol`, and LUNs of the opposite class. A class `vvol` LUN must be bound to a class `protocol_endpoint` LUN in order to be accessed. Class `protocol_endpoint` and `vvol` LUNs allow many-to-many bindings. A LUN of one class is allowed to be bound to zero or more LUNs of the opposite class. The binding between any two specific LUNs is reference counted. When a binding is created that already exists, the binding count is incremented. When a binding is deleted, the binding count is decremented, but the LUNs remain bound if the resultant reference count is greater than zero. When the binding count reaches zero, the binding is destroyed. The bindings array contains LUNs of the opposite class of the containing LUN object. There is an added computational cost to retrieving property values for `vvol.bindings`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
is_bound	boolean	Reports if the LUN is part of a VMware virtual volume (vVol) bind relationship. This is `true` if the LUN is of class `protocol_endpoint` or `vvol` and has one or more bindings to a LUN of the opposite class. This is false if the LUN is of class `regular` or unbound.

lun

A LUN is the logical representation of storage in a storage area network (SAN).

A LUN must be mapped to an initiator group to grant access to the initiator group's initiators (client hosts). Initiators can then access the LUN and perform I/O over a Fibre Channel (FC) fabric using the FC Protocol or a TCP/IP network using iSCSI.

See the LUN object model to learn more about each of the properties supported by the LUN REST API.

A LUN is located within a volume. Optionally, it can be located within a qtree in a volume.

LUN names are paths of the form "/vol/<volume>[/<qtree>]/<lun>" where the qtree name is optional.

A LUN can be created to a specified size using thin or thick provisioning. A LUN can then be renamed, resized, cloned, moved to a different volume and copied. LUNs support the assignment of a QoS policy for performance management or a QoS policy can be assigned to a volume containing one or more LUNs.

Name	Type	Description
auto_delete	boolean	This property marks the LUN for auto deletion when the volume containing the LUN runs out of space. This is most commonly set on LUN clones. When set to *true*, the LUN becomes eligible for automatic deletion when the volume runs out of space. Auto deletion only occurs when the volume containing the LUN is also configured for auto deletion and free space in the volume decreases below a particular threshold. This property is optional in POST and PATCH. The default value for a new LUN is *false*. There is an added computational cost to retrieving this property's value. It is not populated for a GET request unless it is explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
clone	clone	This sub-object is used in POST to create a new LUN as a clone of an existing LUN, or PATCH to overwrite an existing LUN as a clone of another. Setting a property in this sub-object indicates that a LUN clone is desired. Consider the following other properties when cloning a LUN: `auto_delete`, `qos_policy`, `space.guarantee.requested` and `space.scsi_thin_provisioning_support_enabled`. When used in a PATCH, the patched LUN's data is over-written as a clone of the source and the following properties are preserved from the patched LUN unless otherwise specified as part of the PATCH: `class`, `auto_delete`, `lun_maps`, `serial_number`, `status.state`, and `uuid`. Persistent reservations for the patched LUN are also preserved.
comment	string	A configurable comment available for use by the administrator. Valid in POST and PATCH.
consistency_group	consistency_group	The LUN's consistency group. This property is populated for LUNs that are members of a consistency group. If the LUN is a member of a child consistency group, the parent consistency group is reported. A LUN's consistency group is the consistency group of its containing volume.

Name	Type	Description
copy	copy	This sub-object applies to LUN copy operations. A LUN can be copied with a POST request that supplies `copy.source` properties. Copying a LUN is an asynchronous activity begun by a POST request that specifies the source of the copy in the `copy.source` properties. The data for the LUN is then asynchronously copied from the source to the destination. The time required to complete the copy depends on the size of the LUN and the load on the cluster. The `copy` sub-object is populated while a LUN copy is in progress and for two (2) minutes following completion of a copy. While LUNs are being copied, the status of the LUN copy operations can be obtained using a GET of the source or destination LUN that requests the `copy` properties. If the LUN is the source LUN for one or more copy operations, the `copy.destinations` array is populated in GET. If the containing LUN is the destination LUN for a copy operation, the `copy.source` sub-object is populated in GET. The LUN copy operation can be further modified using a PATCH on the properties on the `copy.source` sub-object of the copy destination LUN. There is an added computational cost to retrieving property values for `copy`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.
create_time	string	The time the LUN was created.

Name	Type	Description
enabled	boolean	The enabled state of the LUN. LUNs can be disabled to prevent access to the LUN. Certain error conditions also cause the LUN to become disabled. If the LUN is disabled, check the `status.state` property to determine if the LUN is administratively disabled (*offline*) or has become disabled as a result of an error. A LUN in an error condition can be brought online by setting the `enabled` property to *true* or brought administratively offline by setting the `enabled` property to *false*. Upon creation, a LUN is enabled by default. Valid in PATCH.
location	location	The location of the LUN within the ONTAP cluster. LUNs support rename and move between volumes. Valid in POST and PATCH. • Introduced in: 9.6

Name	Type	Description
movement	movement	This sub-object applies to LUN movement between volumes. A LUN can be moved to a new volume with a PATCH request that changes either the volume portion of property `name`, `location.volume.uuid`, or `location.volume.name`. If the volume is changed using more than one of these properties, the supplied properties used must refer to the same volume. Moving a LUN between volumes is an asynchronous activity begun by a PATCH request. The data for the LUN is then asynchronously copied from the source volume to the destination volume. The time required to complete the move depends on the size of the LUN and the load on the cluster. The `movement` sub-object is populated while a LUN movement is in progress and for two (2) minutes following completion of a movement. While the LUN is being moved, the status of the LUN movement operation can be obtained using a GET for the LUN that requests the `movement` properties. The LUN movement operation can be further modified using a PATCH on the properties on the `movement` sub-object. There is an added computational cost to retrieving property values for `movement`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

Name	Type	Description
name	string	The name of the LUN. Valid in POST and PATCH. A LUN is located within a volume. Optionally, it can be located within a qtree in a volume. LUN names are paths of the form "/vol/<volume>[/<qtree>]/<lun>" where the qtree name is optional. A PATCH that modifies the qtree and/or base name portion of the LUN path is considered a rename operation. A PATCH that modifies the volume portion of the LUN path begins an asynchronous LUN movement operation.
qos_policy	qos_policy	The QoS policy for the LUN. Both traditional and adaptive QoS policies are supported. If both property `qos_policy.uuid` and `qos_policy.name` are specified in the same request, they must refer to the same QoS policy. To remove the QoS policy from a LUN, leaving it with no QoS policy, set the property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.
serial_number	string	The LUN serial number. The serial number is generated by ONTAP when the LUN is created. • maxLength: 12 • minLength: 12 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true

Name	Type	Description
serial_number_hex	string	The LUN serial number encoded in hexadecimal format. The serial number is generated by ONTAP when the LUN is created. • readOnly: 1 • Introduced in: 9.17 • x-nullable: true
space	space	The storage space related properties of the LUN.
status	status	Status information about the LUN.
uuid	string	The unique identifier of the LUN. The UUID is generated by ONTAP when the LUN is created. • example: 1cd8a442-86d1-11e0-ae1c-123478563412 • readOnly: 1 • Introduced in: 9.6 • x-nullable: true

Name	Type	Description
vvol	vvol	A VMware virtual volume (vVol) binding is an association between a LUN of class `protocol_endpoint` and a LUN of class `vvol`. Class `protocol_endpoint` LUNs are mapped to igroups and granted access using the same configuration as class `regular` LUNs. When a class `vvol` LUN is bound to a mapped class `protocol_endpoint` LUN, VMware can access the class `vvol` LUN through the class `protocol_endpoint` LUN mapping. See DELETE /protocols/san/vvol-bindings to learn more about deleting vVol bindings. There is an added computational cost to retrieving property values for `vvol`. They are not populated for a GET request unless explicitly requested using the `fields` query parameter. See Requesting specific fields to learn more.

returned_error

Name	Type	Description
arguments	array[error_arguments]	Message arguments
code	string	Error code
message	string	Error message
target	string	The target parameter that caused the error.

Copyright information

Trademark information