# NetApp

# Manage LUNs

REST API reference

NetApp
September 12, 2025

# Table of Contents

# Manage LUNs

## Storage luns endpoint overview

### 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.

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

A LUN can be created to a specified size using thin or thick provisioning. A LUN can then be renamed, resized, cloned, and moved to a different volume. LUNs support the assignment of a quality of service (QoS) policy for performance management or a QoS policy can be assigned to the volume containing the LUN. See the LUN object model to learn more about each of the properties supported by the LUN REST API.

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.

### Performance monitoring

Performance of a LUN can be monitored by observing the `metric.*` and `statistics.*` properties. These properties show the performance of a LUN in terms of IOPS, latency and throughput. The `metric.*` properties denote an average whereas `statistics.*` properies 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": [
  {
```

```json
      "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": {
        "size": 322163441664,
        "used": 0,
        "guarantee": {
          "requested": false,
          "reserved": false
        }
      },
      "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": "regular",
"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": {
    "size": 322163441664,
    "used": 0,
    "guarantee": {
      "requested": false,
      "reserved": false
    }
  }
},
 "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,
```

```json
      "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"
  },
  "_links": {
    "self": {
      "href": "/api/storage/luns/5a24ae5b-28af-47fb-b129-
5adf6cfba0a6?fields=**"
    }
  }
}
```

## 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 also 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.

## Examples

### 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 copy, 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 copy. 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`, `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" } } }'
```

## 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.

## Examples

**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"
  }
}
}
```

**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'
```

# Retrieve LUNs

GET `/storage/luns`

Retrieves LUNs.

## Expensive properties

There is an added 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 DOC Requesting specific fields to learn more.

- `auto_delete`
- `lun_maps.*`
- `movement.*`
- `status.mapped`
- `statistics.*`
- `metrics.*`

## Related ONTAP commands

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

## Learn more

- DOC /storage/luns

## Parameters

| Name | Type | In | Required | Description |
|------|------|-----|----------|-------------|
| qos_policy.uuid | string | query | False | Filter by qos_policy.uuid |
| qos_policy.name | string | query | False | Filter by qos_policy.name |
| svm.uuid | string | query | False | Filter by svm.uuid |
| svm.name | string | query | False | Filter by svm.name |
| os_type | string | query | False | Filter by os_type |

| Name | Type | In | Required | Description |
|---|---|---|---|---|
| enabled | boolean | query | False | Filter by enabled |
| movement.paths.destination | string | query | False | Filter by movement.paths.destination |
| movement.paths.source | string | query | False | Filter by movement.paths.source |
| movement.progress.failure.code | string | query | False | Filter by movement.progress.failure.code |
| movement.progress.failure.message | string | query | False | Filter by movement.progress.failure.message |
| movement.progress.volume_snapshot_blocked | boolean | query | False | Filter by movement.progress.volume_snapshot_blocked |
| movement.progress.percent_complete | integer | query | False | Filter by movement.progress.percent_complete |
| movement.progress.elapsed | integer | query | False | Filter by movement.progress.elapsed |
| movement.progress.state | string | query | False | Filter by movement.progress.state |
| movement.max_throughput | string | query | False | Filter by movement.max_throughput |
| uuid | string | query | False | Filter by uuid |
| auto_delete | boolean | query | False | Filter by auto_delete |
| name | string | query | False | Filter by name |

| Name | Type | In | Required | Description |
|---|---|---|---|---|
| space.guarantee.requested | boolean | query | False | Filter by space.guarantee.requested |
| space.guarantee.reserved | boolean | query | False | Filter by space.guarantee.reserved |
| space.size | integer | query | False | Filter by space.size |
| space.used | integer | query | False | Filter by space.used |
| comment | string | query | False | Filter by comment |
| status.state | string | query | False | Filter by status.state |
| status.container_state | string | query | False | Filter by status.container_state |
| status.mapped | boolean | query | False | Filter by status.mapped |
| status.read_only | boolean | query | False | Filter by status.read_only |
| serial_number | string | query | False | Filter by serial_number |
| location.logical_unit | string | query | False | Filter by location.logical_unit |
| location.volume.uuid | string | query | False | Filter by location.volume.uuid |
| location.volume.name | string | query | False | Filter by location.volume.name |
| location.qtree.name | string | query | False | Filter by location.qtree.name |
| location.qtree.id | integer | query | False | Filter by location.qtree.id |

| Name | Type | In | Required | Description |
|---|---|---|---|---|
| lun_maps.logical_unit_number | integer | query | False | Filter by lun_maps.logical_unit_number |
| 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 |
| class | string | query | False | Filter by class |
| metric.timestamp | string | query | False | Filter by metric.timestamp |
| metric.iops.other | integer | query | False | Filter by metric.iops.other |
| metric.iops.write | integer | query | False | Filter by metric.iops.write |
| metric.iops.read | integer | query | False | Filter by metric.iops.read |
| metric.iops.total | integer | query | False | Filter by metric.iops.total |
| metric.status | string | query | False | Filter by metric.status |
| metric.duration | string | query | False | Filter by metric.duration |
| metric.latency.other | integer | query | False | Filter by metric.latency.other |
| metric.latency.write | integer | query | False | Filter by metric.latency.write |
| metric.latency.read | integer | query | False | Filter by metric.latency.read |
| metric.latency.total | integer | query | False | Filter by metric.latency.total |

| Name | Type | In | Required | Description |
|---|---|---|---|---|
| metric.throughput.other | integer | query | False | Filter by metric.throughput.other |
| metric.throughput.write | integer | query | False | Filter by metric.throughput.write |
| metric.throughput.read | integer | query | False | Filter by metric.throughput.read |
| metric.throughput.total | integer | query | False | Filter by metric.throughput.total |
| statistics.iops_raw.other | integer | query | False | Filter by statistics.iops_raw.other |
| statistics.iops_raw.write | integer | query | False | Filter by statistics.iops_raw.write |
| statistics.iops_raw.read | integer | query | False | Filter by statistics.iops_raw.read |
| statistics.iops_raw.total | integer | query | False | Filter by statistics.iops_raw.total |
| statistics.timestamp | string | query | False | Filter by statistics.timestamp |
| statistics.throughput_raw.other | integer | query | False | Filter by statistics.throughput_raw.other |
| statistics.throughput_raw.write | integer | query | False | Filter by statistics.throughput_raw.write |
| statistics.throughput_raw.read | integer | query | False | Filter by statistics.throughput_raw.read |

| Name | Type | In | Required | Description |
|---|---|---|---|---|
| statistics.throughput_raw.total | integer | query | False | Filter by statistics.throughput_raw.total |
| statistics.status | string | query | False | Filter by statistics.status |
| statistics.latency_raw.other | integer | query | False | Filter by statistics.latency_raw.other |
| statistics.latency_raw.write | integer | query | False | Filter by statistics.latency_raw.write |
| statistics.latency_raw.read | integer | query | False | Filter by statistics.latency_raw.read |
| statistics.latency_raw.total | integer | query | False | Filter by statistics.latency_raw.total |
| create_time | string | query | False | Filter by create_time |
| 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. |

| Name | Type | In | Required | Description |
|------|------|-----|----------|-------------|
| 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. |
| 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 | Number of records. |
| records | array[lun] | |

**Example response**

```json
{
  "_links": {
    "next": {
      "href": "/api/resourcelink"
    },
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "records": [
    {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "class": "string",
      "comment": "string",
      "create_time": "2018-06-04 19:00:00 UTC",
      "location": {
        "logical_unit": "lun1",
        "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": {
```

```json
            "href": "/api/resourcelink"
          }
        },
        "igroup": {
          "_links": {
            "self": {
              "href": "/api/resourcelink"
            }
          },
          "name": "igroup1",
          "uuid": "4ea7a442-86d1-11e0-ae1c-123478563412"
        },
        "logical_unit_number": 0
      }
    ],
    "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 11:20:13 UTC"
    },
    "movement": {
      "max_throughput": "string",
      "paths": {
        "destination": "/vol/vol1/lun1",
        "source": "/vol/vol2/lun2"
      },
```

```json
          "progress": {
            "elapsed": 0,
            "failure": {
              "code": "4",
              "message": "Destination volume is offline."
            },
            "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",
        "space": {
          "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 11:20:13 UTC"
        },
        "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"
    }
  ]
}
```

## Error

```
Status: Default, Error
```

| Name | Type | Description |
|------|------|-------------|
| error | 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 | |

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`, and `space.guarantee.requested`.

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.

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.uuid` 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. |
| 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. <br><br> • example: 028baa66-41bd-11e9-81d5-00a0986138f7 |

location

The location of the LUN within the ONTAP cluster. Valid in POST and PATCH.

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

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

igroup

The initiator group to which the LUN is mapped.

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

lun_maps

A LUN map with which the LUN is associated.

| 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 for initiators in the initiator group. |

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 | Peformance 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. |

| Name | Type | Description |
| --- | --- | --- |
| write | integer | Peformance 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 | Peformance 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. |

failure

Error information provided if the asynchronous LUN movement operation fails.

| Name | Type | Description |
|------|------|-------------|
| code | string | The error code. |
| message | string | The error message. |

progress

| Name | Type | Description |
|------|------|-------------|
| elapsed | integer | The amount of time, in seconds, that has elapsed since the start of the LUN movement. |
| failure | failure | Error information provided if the asynchronous LUN movement operation fails. |
| percent_complete | integer | The percentage complete of the LUN movement. |
| state | string | The state of the LUN movement.<br><br>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. |
| volume_snapshot_blocked | boolean | This property reports if volume Snapshot copies are blocked by the LUN movement. This property can be polled to identify when volume Snapshot copies 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 added cost to retrieving property values for `movement`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more.

| Name | Type | Description |
|---|---|---|
| max_throughput | string | The maximum data throughput 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.<br><br>If this property is not specified in a POST that begins a LUN movement, throttling is not applied to the data transfer.<br><br>For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation.<br><br>This property is valid only in a POST that begins a LUN movement or a PATCH when a LUN movement is already in process. |
| paths | paths | The fully qualified LUN path names involved in the LUN movement. |
| progress | progress | |

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 property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.

Note that a QoS policy can be set on a LUN, or a LUN's volume, but not both.

| 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. |

| Name | Type | Description |
|---|---|---|
| reserved | boolean | Reports if the LUN is space guaranteed.<br><br>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 |
|---|---|---|
| guarantee | guarantee | Properties that request and report the space guarantee for the LUN. |
| size | integer | The total provisioned size of the LUN. The LUN size can be increased but not be made smaller using the REST interface.<br><br>For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation.<br><br>• example: 1073741824 |

| Name | Type | Description |
|------|------|-------------|
| used | integer | The amount of space consumed by the main data stream of the LUN.<br><br>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.<br><br>For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation.<br><br>• readOnly: 1 |

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. |
| total | integer | Performance metric aggregated over all types of I/O operations. |

| Name | Type | Description |
|------|------|-------------|
| write | integer | Peformance 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 | Peformance 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. |
| write | integer | Peformance 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. |
| 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 cost to retrieving this property's value. It is not populated for either a collection GET or an instance GET unless it is explicitly requested using the `fields` query parameter. See DOC 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 | string | The name of the SVM. |
| uuid | string | The unique identifier of the SVM. |

lun

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

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

A LUN can be created to a specified size using thin or thick provisioning. A LUN can then be renamed, resized, cloned, and moved to a different volume. LUNs support the assignment of a quality of service

(QoS) policy for performance management or a QoS policy can be assigned to the volume containing the LUN. See the LUN object model to learn more about each of the properties supported by the LUN REST API.

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 Fibre Channel Protocol or a TCP/IP network using iSCSI.

| Name | Type | Description |
|------|------|-------------|
| _links | _links | |
| 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. <br><br> 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. <br><br> This property is optional in POST and PATCH. The default value for a new LUN is *false*. <br><br> There is an added cost to retrieving this property's value. It is not populated for either a collection GET or an instance GET unless it is explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |
| class | string | The class of LUN. Only *regular* LUNs can be created using the REST API. |
| comment | string | A configurable comment available for use by the administrator. Valid in POST and PATCH. |
| 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, you can consult the `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. Valid in POST and PATCH. |
| lun_maps | array[lun_maps] | The LUN maps with which the LUN is associated. There is an added cost to retrieving property values for `lun_maps`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |
| 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.<br><br>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.<br><br>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.<br><br>There is added cost to retrieving property values for `movement`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |

| Name | Type | Description |
|------|------|-------------|
| name | string | The fully qualified path name of the LUN composed of a "/vol" prefix, the volume name, the (optional) qtree name, and base name of the LUN. Valid in POST and PATCH.<br><br>A PATCH that modifies the qtree and/or base name portion of the LUN path is considered a rename operation.<br><br>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.<br><br>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 property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.<br><br>Note that a QoS policy can be set on a LUN, or a LUN's volume, but not both. |

| Name | Type | Description |
|---|---|---|
| serial_number | string | The LUN serial number. The serial number is generated by ONTAP when the LUN is created.<br><br>   • maxLength: 12<br>   • minLength: 12<br>   • readOnly: 1 |
| 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.<br><br>   • example: 1cd8a442-86d1-11e0-ae1c-123478563412<br>   • readOnly: 1 |

error_arguments

| Name | Type | Description |
|---|---|---|
| code | string | Argument code |
| message | string | Message argument |

error

| Name | Type | Description |
|---|---|---|
| arguments | array[error_arguments] | Message arguments |

| Name | Type | Description |
|------|------|-------------|
| code | string | Error code |
| message | string | Error message |
| target | string | The target parameter that caused the error. |

# Create a LUN

POST `/storage/luns`

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`
- `volume file clone autodelete`
- `volume file clone create`

## Learn more

## Parameters

| Name | Type | In | Required | Description |
|------|------|-----|----------|-------------|
| return_records | boolean | query | False | The default is false. If set to true, the records are returned. |

## Request Body

| Name | Type | Description |
|------|------|-------------|
| _links | _links | |
| 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.<br><br>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.<br><br>This property is optional in POST and PATCH. The default value for a new LUN is *false*.<br><br>There is an added cost to retrieving this property's value. It is not populated for either a collection GET or an instance GET unless it is explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |
| class | string | The class of LUN. Only *regular* LUNs can be created using the REST API. |

| 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`, and `space.guarantee.requested`.<br><br>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`.<br><br>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. |
| create_time | string | The time the LUN was created. |
| location | location | The location of the LUN within the ONTAP cluster. Valid in POST and PATCH. |
| lun_maps | array[lun_maps] | The LUN maps with which the LUN is associated.<br><br>There is an added cost to retrieving property values for `lun_maps`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |
| 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 added cost to retrieving property values for `movement`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |

| Name | Type | Description |
|------|------|-------------|
| name | string | The fully qualified path name of the LUN composed of a "/vol" prefix, the volume name, the (optional) qtree name, and base name of the LUN. Valid in POST and PATCH.<br><br>A PATCH that modifies the qtree and/or base name portion of the LUN path is considered a rename operation.<br><br>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.<br><br>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 property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.<br><br>Note that a QoS policy can be set on a LUN, or a LUN's volume, but not both. |
| serial_number | string | The LUN serial number. The serial number is generated by ONTAP when the LUN is created.<br><br>• maxLength: 12<br>• minLength: 12<br>• readOnly: 1 |

| Name | Type | Description |
|------|------|-------------|
| 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.<br><br>• example: 1cd8a442-86d1-11e0-ae1c-123478563412<br>• readOnly: 1 |

**Example request**

```json
{
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "class": "string",
  "clone": {
    "source": {
      "name": "/vol/volume1/lun1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    }
  },
  "comment": "string",
  "create_time": "2018-06-04 19:00:00 UTC",
  "enabled": null,
  "location": {
    "logical_unit": "lun1",
    "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"
        }
      },
```

```json
        "igroup": {
          "_links": {
            "self": {
              "href": "/api/resourcelink"
            }
          },
          "name": "igroup1",
          "uuid": "4ea7a442-86d1-11e0-ae1c-123478563412"
        },
        "logical_unit_number": 0
      }
    ],
    "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 11:20:13 UTC"
    },
    "movement": {
      "max_throughput": "string",
      "paths": {
        "destination": "/vol/vol1/lun1",
        "source": "/vol/vol2/lun2"
      }
    },
    "name": "/vol/volume1/qtree1/lun1",
    "os_type": "string",
```

```json
    "qos_policy": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "name": "qos1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    },
    "serial_number": "string",
    "space": {
      "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 11:20:13 UTC"
    },
    "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"
  }
```

## Response

```
Status: 201, Created
```

| Name | Type | Description |
| --- | --- | --- |
| _links | _links | |
| num_records | integer | Number of records. |
| records | array[lun] | |

**Example response**

```json
{
  "_links": {
    "next": {
      "href": "/api/resourcelink"
    },
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "records": [
    {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "class": "string",
      "clone": {
        "source": {
          "name": "/vol/volume1/lun1",
          "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
        }
      },
      "comment": "string",
      "create_time": "2018-06-04 19:00:00 UTC",
      "enabled": null,
      "location": {
        "logical_unit": "lun1",
        "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"
                      }
                  },
                  "name": "igroup1",
                  "uuid": "4ea7a442-86d1-11e0-ae1c-123478563412"
              },
              "logical_unit_number": 0
          }
      ],
      "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 11:20:13 UTC"
```

```json
        },
        "movement": {
          "max_throughput": "string",
          "paths": {
            "destination": "/vol/vol1/lun1",
            "source": "/vol/vol2/lun2"
          }
        },
        "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",
        "space": {
          "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 11:20:13 UTC"
        },
        "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"
      }
    ]
  }
```

## 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. |
| 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. |

| Error Code | Description |
|---|---|
| 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 Snapshot copies. |
| 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. |
| 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 | The property is required except when creating a LUN clone. 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 copy. |
| 13565952 | The LUN clone request failed. |

| Name | Type | Description |
|------|------|-------------|
| error | error | |

**Example error**

```json
{
  "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 | |

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 fully qualified path name of the clone source LUN composed of a "/vol" prefix, the volume name, the (optional) qtree name, and base name of the LUN. 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`, and `space.guarantee.requested`.

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.<br><br>Valid in POST to create a new LUN as a clone of the source.<br><br>Valid in PATCH to overwrite an existing LUN's data as a clone of another. |

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.uuid` 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. |

| Name | Type | Description |
|------|------|-------------|
| 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.<br><br>• example: 028baa66-41bd-11e9-81d5-00a0986138f7 |

location

The location of the LUN within the ONTAP cluster. Valid in POST and PATCH.

| Name | Type | Description |
|------|------|-------------|
| logical_unit | string | The base name component of the LUN. Valid in POST and PATCH.<br><br>If properties `name` and `location.logical_unit` are specified in the same request, they must refer to the base name.<br><br>A PATCH that modifies the base name of the LUN is considered a rename operation. |
| qtree | qtree | The qtree in which the LUN is optionally located. Valid in POST and PATCH.<br><br>If properties `name` and `location.qtree.name` and/or `location.qtree.uuid` are specified in the same request, they must refer to the same qtree.<br><br>A PATCH that modifies the qtree of the LUN is considered a rename operation. |

| Name | Type | Description |
|---|---|---|
| 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. |

## igroup

The initiator group to which the LUN is mapped.

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

## lun_maps

A LUN map with which the LUN is associated.

| 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 for initiators in the initiator group. |

## 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 | Peformance 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 | Peformance 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 | Peformance 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. |

failure

Error information provided if the asynchronous LUN movement operation fails.

| Name | Type | Description |
|------|------|-------------|
| code | string | The error code. |
| message | string | The error message. |

progress

| Name | Type | Description |
|------|------|-------------|
| elapsed | integer | The amount of time, in seconds, that has elapsed since the start of the LUN movement. |
| failure | failure | Error information provided if the asynchronous LUN movement operation fails. |
| percent_complete | integer | The percentage complete of the LUN movement. |
| volume_snapshot_blocked | boolean | This property reports if volume Snapshot copies are blocked by the LUN movement. This property can be polled to identify when volume Snapshot copies 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 added cost to retrieving property values for `movement`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more.

| Name | Type | Description |
|------|------|-------------|
| max_throughput | string | The maximum data throughput 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.<br><br>If this property is not specified in a POST that begins a LUN movement, throttling is not applied to the data transfer.<br><br>For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation.<br><br>This property is valid only in a POST that begins a LUN movement or a PATCH when a LUN movement is already in process. |
| paths | paths | The fully qualified LUN path names involved in the LUN movement. |

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 property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.

Note that a QoS policy can be set on a LUN, or a LUN's volume, but not both.

| 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. |
| 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 |
|---|---|---|
| guarantee | guarantee | Properties that request and report the space guarantee for the LUN. |
| size | integer | The total provisioned size of the LUN. The LUN size can be increased but not be made smaller using the REST interface.<br><br>For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation.<br><br>• example: 1073741824 |
| used | integer | The amount of space consumed by the main data stream of the LUN.<br><br>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.<br><br>For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation.<br><br>• readOnly: 1 |

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. |
| total | integer | Performance metric aggregated over all types of I/O operations. |
| write | integer | Peformance 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 | Peformance 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. |
| write | integer | Peformance 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. |

| 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_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. |

| Name | Type | Description |
|------|------|-------------|
| mapped | boolean | Reports if the LUN is mapped to one or more initiator groups.<br><br>There is an added cost to retrieving this property's value. It is not populated for either a collection GET or an instance GET unless it is explicitly requested using the `fields` query parameter. See DOC 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 | string | The name of the SVM. |
| uuid | string | The unique identifier of the SVM. |

lun

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

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

A LUN can be created to a specified size using thin or thick provisioning. A LUN can then be renamed, resized, cloned, and moved to a different volume. LUNs support the assignment of a quality of service (QoS) policy for performance management or a QoS policy can be assigned to the volume containing the LUN. See the LUN object model to learn more about each of the properties supported by the LUN REST API.

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 Fibre Channel Protocol or a TCP/IP network using iSCSI.

| Name | Type | Description |
|------|------|-------------|
| _links | _links | |
| 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.<br><br>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.<br><br>This property is optional in POST and PATCH. The default value for a new LUN is *false*.<br><br>There is an added cost to retrieving this property's value. It is not populated for either a collection GET or an instance GET unless it is explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |
| class | string | The class of LUN. Only *regular* LUNs can be created using the REST API. |

| 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`, and `space.guarantee.requested`.<br><br>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`.<br><br>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. |
| create_time | string | The time the LUN was created. |
| location | location | The location of the LUN within the ONTAP cluster. Valid in POST and PATCH. |
| lun_maps | array[lun_maps] | The LUN maps with which the LUN is associated.<br><br>There is an added cost to retrieving property values for `lun_maps`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |

| Name | Type | Description |
|------|------|-------------|
| metric | metric | Performance numbers, such as IOPS latency and throughput. |
| 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.<br><br>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.<br><br>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.<br><br>There is added cost to retrieving property values for `movement`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |

| Name | Type | Description |
|------|------|-------------|
| name | string | The fully qualified path name of the LUN composed of a "/vol" prefix, the volume name, the (optional) qtree name, and base name of the LUN. Valid in POST and PATCH.<br><br>A PATCH that modifies the qtree and/or base name portion of the LUN path is considered a rename operation.<br><br>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.<br><br>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 property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.<br><br>Note that a QoS policy can be set on a LUN, or a LUN's volume, but not both. |

| Name | Type | Description |
|---|---|---|
| serial_number | string | The LUN serial number. The serial number is generated by ONTAP when the LUN is created.<br><br>• maxLength: 12<br>• minLength: 12<br>• readOnly: 1 |
| 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.<br><br>• example: 1cd8a442-86d1-11e0-ae1c-123478563412<br>• readOnly: 1 |

_links

| Name | Type | Description |
|---|---|---|
| next | href | |
| self | href | |

error_arguments

| Name | Type | Description |
|---|---|---|
| code | string | Argument code |
| message | string | Message argument |

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. |

# Delete a LUN

DELETE /storage/luns/{uuid}

Deletes a LUN.

## Related ONTAP commands

- `lun delete`

## Learn more

- DOC /storage/luns

## Parameters

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

| Name | Type | In | Required | Description |
|------|------|-----|----------|-------------|
| 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.**<br><br>• Default value: |

## Response

```
Status: 200, Ok
```

## 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. |

| Name | Type | Description |
|------|------|-------------|
| error | 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 |

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}`

Retrieves a LUN.

## Expensive properties

There is an added 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 DOC Requesting specific fields to learn more.

- `auto_delete`
- `lun_maps.*`
- `movement.*`
- `status.mapped`
- `statistics.*`
- `metrics.*`

## Related ONTAP commands

- `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. |
| fields | array[string] | query | False | Specify the fields to return. |

## Response

```
Status: 200, Ok
```

| Name | Type | Description |
|------|------|-------------|
| _links | _links | |
| 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.<br><br>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.<br><br>This property is optional in POST and PATCH. The default value for a new LUN is *false*.<br><br>There is an added cost to retrieving this property's value. It is not populated for either a collection GET or an instance GET unless it is explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |
| class | string | The class of LUN. Only *regular* LUNs can be created using the REST API. |
| comment | string | A configurable comment available for use by the administrator. Valid in POST and PATCH. |
| 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, you can consult the `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. Valid in POST and PATCH. |
| lun_maps | array[lun_maps] | The LUN maps with which the LUN is associated.

There is an added cost to retrieving property values for `lun_maps`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |
| 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.<br><br>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.<br><br>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.<br><br>There is added cost to retrieving property values for `movement`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |

| Name | Type | Description |
|------|------|-------------|
| name | string | The fully qualified path name of the LUN composed of a "/vol" prefix, the volume name, the (optional) qtree name, and base name of the LUN. Valid in POST and PATCH. 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](#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 property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH. Note that a QoS policy can be set on a LUN, or a LUN's volume, but not both. |
| serial_number | string | The LUN serial number. The serial number is generated by ONTAP when the LUN is created. <ul><li>maxLength: 12</li><li>minLength: 12</li><li>readOnly: 1</li></ul> |

| Name | Type | Description |
|------|------|-------------|
| 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.<br><br>• example: 1cd8a442-86d1-11e0-ae1c-123478563412<br>• readOnly: 1 |

**Example response**

```json
{
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "class": "string",
  "comment": "string",
  "create_time": "2018-06-04 19:00:00 UTC",
  "location": {
    "logical_unit": "lun1",
    "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"
          }
        },
        "name": "igroup1",
```

```json
        "uuid": "4ea7a442-86d1-11e0-ae1c-123478563412"
      },
      "logical_unit_number": 0
    }
  ],
  "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 11:20:13 UTC"
  },
  "movement": {
    "max_throughput": "string",
    "paths": {
      "destination": "/vol/vol1/lun1",
      "source": "/vol/vol2/lun2"
    },
    "progress": {
      "elapsed": 0,
      "failure": {
        "code": "4",
        "message": "Destination volume is offline."
      },
      "percent_complete": 0,
      "state": "string"
    }
  },
```

```json
      "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",
      "space": {
        "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 11:20:13 UTC"
      },
      "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"
}
```

## Error

```
Status: Default
```

ONTAP Error Response Codes

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

| Name | Type | Description |
|---|---|---|
| error | 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 | |

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`, and `space.guarantee.requested`.

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.

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.uuid` 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. |
| 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.<br><br>• example: 028baa66-41bd-11e9-81d5-00a0986138f7 |

location

The location of the LUN within the ONTAP cluster. Valid in POST and PATCH.

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

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

igroup

The initiator group to which the LUN is mapped.

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

lun_maps

A LUN map with which the LUN is associated.

| 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 for initiators in the initiator group. |

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 | Peformance 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. |

| Name | Type | Description |
| --- | --- | --- |
| write | integer | Peformance 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 | Peformance 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. |

**failure**

Error information provided if the asynchronous LUN movement operation fails.

| Name | Type | Description |
|---|---|---|
| code | string | The error code. |
| message | string | The error message. |

**progress**

| Name | Type | Description |
|---|---|---|
| elapsed | integer | The amount of time, in seconds, that has elapsed since the start of the LUN movement. |
| failure | [failure](#) | Error information provided if the asynchronous LUN movement operation fails. |
| percent_complete | integer | The percentage complete of the LUN movement. |
| state | string | The state of the LUN movement.<br><br>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. |
| volume_snapshot_blocked | boolean | This property reports if volume Snapshot copies are blocked by the LUN movement. This property can be polled to identify when volume Snapshot copies 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 added cost to retrieving property values for `movement`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more.

| Name | Type | Description |
| --- | --- | --- |
| max_throughput | string | The maximum data throughput 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.<br><br>If this property is not specified in a POST that begins a LUN movement, throttling is not applied to the data transfer.<br><br>For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation.<br><br>This property is valid only in a POST that begins a LUN movement or a PATCH when a LUN movement is already in process. |
| paths | paths | The fully qualified LUN path names involved in the LUN movement. |
| progress | progress | |

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 property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.

Note that a QoS policy can be set on a LUN, or a LUN's volume, but not both.

| 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. |

| Name | Type | Description |
|---|---|---|
| reserved | boolean | Reports if the LUN is space guaranteed.<br><br>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 |
|---|---|---|
| guarantee | guarantee | Properties that request and report the space guarantee for the LUN. |
| size | integer | The total provisioned size of the LUN. The LUN size can be increased but not be made smaller using the REST interface.<br><br>For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation.<br><br>• example: 1073741824 |

| Name | Type | Description |
|------|------|-------------|
| used | integer | The amount of space consumed by the main data stream of the LUN.<br><br>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.<br><br>For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation.<br><br>• readOnly: 1 |

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. |
| total | integer | Performance metric aggregated over all types of I/O operations. |

| Name | Type | Description |
|------|------|-------------|
| write | integer | Peformance 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 | Peformance 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. |
| write | integer | Peformance 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. |
| 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 cost to retrieving this property's value. It is not populated for either a collection GET or an instance GET unless it is explicitly requested using the `fields` query parameter. See DOC 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 | string | The name of the SVM. |
| uuid | string | The unique identifier of the SVM. |

## error_arguments

| Name | Type | Description |
|------|------|-------------|
| code | string | Argument code |
| message | string | Message argument |

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}`

Updates the properties of a LUN. A PATCH request can also be be used to overwrite the contents of a LUN as a clone of another, to begin movement of a LUN between volumes, and to pause and resume the movement of a LUN between volumes.

## Related ONTAP commands

- `lun modify`
- `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 update. |

## Request Body

| Name | Type | Description |
| --- | --- | --- |
| _links | _links | |
| 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.<br><br>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.<br><br>This property is optional in POST and PATCH. The default value for a new LUN is *false*.<br><br>There is an added cost to retrieving this property's value. It is not populated for either a collection GET or an instance GET unless it is explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |
| class | string | The class of LUN. Only *regular* LUNs can be created using the REST API. |

| 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`, and `space.guarantee.requested`. 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. |
| create_time | string | The time the LUN was created. |
| 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, you can consult the `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. |

| Name | Type | Description |
|------|------|-------------|
| location | location | The location of the LUN within the ONTAP cluster. Valid in POST and PATCH. |
| lun_maps | array[lun_maps] | The LUN maps with which the LUN is associated.<br><br>There is an added cost to retrieving property values for `lun_maps`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |
| 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.<br><br>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.<br><br>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.<br><br>There is added cost to retrieving property values for `movement`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |

| Name | Type | Description |
| --- | --- | --- |
| name | string | The fully qualified path name of the LUN composed of a "/vol" prefix, the volume name, the (optional) qtree name, and base name of the LUN. Valid in POST and PATCH.<br><br>A PATCH that modifies the qtree and/or base name portion of the LUN path is considered a rename operation.<br><br>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 property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.<br><br>Note that a QoS policy can be set on a LUN, or a LUN's volume, but not both. |
| serial_number | string | The LUN serial number. The serial number is generated by ONTAP when the LUN is created.<br><br>• maxLength: 12<br>• minLength: 12<br>• readOnly: 1 |
| space | space | The storage space related properties of the LUN. |

| Name | Type | Description |
|------|------|-------------|
| 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.<br><br>• example: 1cd8a442-86d1-11e0-ae1c-123478563412<br><br>• readOnly: 1 |

**Example request**

```json
{
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "class": "string",
  "clone": {
    "source": {
      "name": "/vol/volume1/lun1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    }
  },
  "comment": "string",
  "create_time": "2018-06-04 19:00:00 UTC",
  "location": {
    "logical_unit": "lun1",
    "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": {
```

```json
        "_links": {
          "self": {
            "href": "/api/resourcelink"
          }
        },
        "name": "igroup1",
        "uuid": "4ea7a442-86d1-11e0-ae1c-123478563412"
      },
      "logical_unit_number": 0
    }
  ],
  "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 11:20:13 UTC"
  },
  "movement": {
    "max_throughput": "string",
    "paths": {
      "destination": "/vol/vol1/lun1",
      "source": "/vol/vol2/lun2"
    },
    "progress": {
      "elapsed": 0,
      "failure": {
        "code": "4",
```

```json
        "message": "Destination volume is offline."
      },
      "percent_complete": 0,
      "state": "string"
    }
  },
  "name": "/vol/volume1/qtree1/lun1",
  "qos_policy": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "name": "qos1",
    "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
  },
  "serial_number": "string",
  "space": {
    "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 11:20:13 UTC"
  },
  "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"
}
```

## Response

```
Status: 200, Ok
```

## 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. |
| 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. |
| 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`. |

| Error Code | Description |
|---|---|
| 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. |
| 13565952 | The LUN clone request failed. |

| Name | Type | Description |
|---|---|---|
| error | error | |

**Example error**

```json
{
  "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 | |

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 fully qualified path name of the clone source LUN composed of a "/vol" prefix, the volume name, the (optional) qtree name, and base name of the LUN. 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`, and `space.guarantee.requested`.

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.<br><br>Valid in POST to create a new LUN as a clone of the source.<br><br>Valid in PATCH to overwrite an existing LUN's data as a clone of another. |

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.uuid` 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. |

| Name | Type | Description |
|------|------|-------------|
| 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.<br><br>&bull; example: 028baa66-41bd-11e9-81d5-00a0986138f7 |

location

The location of the LUN within the ONTAP cluster. Valid in POST and PATCH.

| Name | Type | Description |
|------|------|-------------|
| logical_unit | string | The base name component of the LUN. Valid in POST and PATCH.<br><br>If properties `name` and `location.logical_unit` are specified in the same request, they must refer to the base name.<br><br>A PATCH that modifies the base name of the LUN is considered a rename operation. |
| qtree | qtree | The qtree in which the LUN is optionally located. Valid in POST and PATCH.<br><br>If properties `name` and `location.qtree.name` and/or `location.qtree.uuid` are specified in the same request, they must refer to the same qtree.<br><br>A PATCH that modifies the qtree of the LUN is considered a rename operation. |

| Name | Type | Description |
|------|------|-------------|
| volume | volume | The volume in which the LUN is located. Valid in POST and PATCH.<br><br>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.<br><br>A PATCH that modifies the volume of the LUN begins an asynchronous LUN movement operation. |

igroup

The initiator group to which the LUN is mapped.

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

lun_maps

A LUN map with which the LUN is associated.

| 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 for initiators in the initiator group. |

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 | Peformance 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 | Peformance 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 | Peformance 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. |

**failure**

Error information provided if the asynchronous LUN movement operation fails.

| Name | Type | Description |
|------|------|-------------|
| code | string | The error code. |
| message | string | The error message. |

**progress**

| Name | Type | Description |
|------|------|-------------|
| elapsed | integer | The amount of time, in seconds, that has elapsed since the start of the LUN movement. |
| failure | failure | Error information provided if the asynchronous LUN movement operation fails. |
| percent_complete | integer | The percentage complete of the LUN movement. |
| state | string | The state of the LUN movement.<br><br>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. |
| volume_snapshot_blocked | boolean | This property reports if volume Snapshot copies are blocked by the LUN movement. This property can be polled to identify when volume Snapshot copies 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 added cost to retrieving property values for `movement`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more.

| Name | Type | Description |
|------|------|-------------|
| max_throughput | string | The maximum data throughput 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.<br><br>If this property is not specified in a POST that begins a LUN movement, throttling is not applied to the data transfer.<br><br>For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation.<br><br>This property is valid only in a POST that begins a LUN movement or a PATCH when a LUN movement is already in process. |
| paths | paths | The fully qualified LUN path names involved in the LUN movement. |
| progress | progress | |

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 property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.

Note that a QoS policy can be set on a LUN, or a LUN's volume, but not both.

| 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. |

| Name | Type | Description |
|---|---|---|
| reserved | boolean | Reports if the LUN is space guaranteed.<br><br>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 |
|---|---|---|
| guarantee | guarantee | Properties that request and report the space guarantee for the LUN. |
| size | integer | The total provisioned size of the LUN. The LUN size can be increased but not be made smaller using the REST interface.<br><br>For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation.<br><br>• example: 1073741824 |

| Name | Type | Description |
|------|------|-------------|
| used | integer | The amount of space consumed by the main data stream of the LUN.<br><br>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.<br><br>For more information, see *Size properties* in the *docs* section of the ONTAP REST API documentation.<br><br>• readOnly: 1 |

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. |
| total | integer | Performance metric aggregated over all types of I/O operations. |

| Name | Type | Description |
| --- | --- | --- |
| write | integer | Peformance 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 | Peformance 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. |
| write | integer | Peformance 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. |
| 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 cost to retrieving this property's value. It is not populated for either a collection GET or an instance GET unless it is explicitly requested using the `fields` query parameter. See DOC 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 | string | The name of the SVM. |
| uuid | string | The unique identifier of the SVM. |

lun

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

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

A LUN can be created to a specified size using thin or thick provisioning. A LUN can then be renamed, resized, cloned, and moved to a different volume. LUNs support the assignment of a quality of service

(QoS) policy for performance management or a QoS policy can be assigned to the volume containing the LUN. See the LUN object model to learn more about each of the properties supported by the LUN REST API.

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 Fibre Channel Protocol or a TCP/IP network using iSCSI.

| Name | Type | Description |
|------|------|-------------|
| _links | _links | |
| 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.<br><br>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.<br><br>This property is optional in POST and PATCH. The default value for a new LUN is *false*.<br><br>There is an added cost to retrieving this property's value. It is not populated for either a collection GET or an instance GET unless it is explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |
| class | string | The class of LUN. Only *regular* LUNs can be created using the REST API. |

| 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`, and `space.guarantee.requested`.<br><br>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`.<br><br>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. |
| 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, you can consult the `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. Valid in POST and PATCH. |
| lun_maps | array[lun_maps] | The LUN maps with which the LUN is associated. There is an added cost to retrieving property values for `lun_maps`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |
| 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.<br><br>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.<br><br>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.<br><br>There is added cost to retrieving property values for `movement`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See DOC Requesting specific fields to learn more. |

| Name | Type | Description |
|------|------|-------------|
| name | string | The fully qualified path name of the LUN composed of a "/vol" prefix, the volume name, the (optional) qtree name, and base name of the LUN. Valid in POST and PATCH.<br><br>A PATCH that modifies the qtree and/or base name portion of the LUN path is considered a rename operation.<br><br>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 property `qos_policy.name` to an empty string ("") in a PATCH request. Valid in POST and PATCH.<br><br>Note that a QoS policy can be set on a LUN, or a LUN's volume, but not both. |
| serial_number | string | The LUN serial number. The serial number is generated by ONTAP when the LUN is created.<br><br>• maxLength: 12<br>• minLength: 12<br>• readOnly: 1 |
| space | space | The storage space related properties of the LUN. |

| Name | Type | Description |
|------|------|-------------|
| 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.<br><br>• example: 1cd8a442-86d1-11e0-ae1c-123478563412<br>• readOnly: 1 |

## error_arguments

| Name | Type | Description |
|------|------|-------------|
| code | string | Argument code |
| message | string | Message argument |

## 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. |