



# Manage NVMe namespaces

ONTAP 9.14.1 REST API reference

NetApp  
May 23, 2024

# Table of Contents

- Manage NVMe namespaces ..... 1
  - Storage namespaces endpoint overview ..... 1
  - Retrieve NVMe namespaces ..... 10
  - Create an NVMe namespace ..... 42
  - Retrieve historical performance metrics for an NVMe namespace ..... 77
  - Retrieve historical performance metrics for an NVMe namespace for a specific time ..... 85
  - Delete an NVMe namespace ..... 91
  - Retrieve an NVMe namespace ..... 93
  - Update an NVMe namespace ..... 116

# Manage NVMe namespaces

## Storage namespaces endpoint overview

### Overview

An NVMe namespace is a collection of addressable logical blocks presented to hosts connected to the storage virtual machine using the NVMe over Fabrics protocol.

The NVMe namespace REST API allows you to create, update, delete and discover NVMe namespaces.

In ONTAP, an NVMe namespace is located within a volume. Optionally, it can be located within a qtree in a volume.

An NVMe namespace is created to a specified size using thin or thick provisioning as determined by the volume on which it is created. NVMe namespaces support being cloned. An NVMe namespace cannot be renamed, resized, or moved to a different volume. NVMe namespaces do not support the assignment of a QoS policy for performance management, but a QoS policy can be assigned to the volume containing the namespace. See the NVMe namespace object model to learn more about each of the properties supported by the NVMe namespace REST API.

An NVMe namespace must be mapped to an NVMe subsystem to grant access to the subsystem's hosts. Hosts can then access the NVMe namespace and perform I/O using the NVMe over Fabrics protocol.

### Performance monitoring

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

### Examples

#### Creating an NVMe namespace

This example creates a 300 gigabyte NVMe namespace, with 4096-byte blocks, in SVM `svm1`, volume `vol1`, configured for use by `linux` hosts. The `return_records` query parameter is used to retrieve properties of the newly created NVMe namespace in the POST response.

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

# The call:
curl -X POST 'https://<mgmt-
ip>/api/storage/namespaces?return_records=true' -H 'Accept:
application/hal+json' -d '{ "svm": { "name": "svm1" }, "os_type": "linux",
"space": { "block_size": "4096", "size": "300G" }, "name" :
"/vol/vol1/namespacel" }'

# The response:
```

```

{
  "num_records": 1,
  "records": [
    {
      "uuid": "dccc3e6-cf4e-498f-bec6-f7897f945669",
      "svm": {
        "uuid": "6bf967fd-2a1c-11e9-b682-005056bbc17d",
        "name": "svm1",
        "_links": {
          "self": {
            "href": "/api/svm/svms/6bf967fd-2a1c-11e9-b682-005056bbc17d"
          }
        }
      },
      "name": "/vol/vol1/namespace1",
      "location": {
        "namespace": "namespace1",
        "volume": {
          "uuid": "71cd0dba-2a1c-11e9-b682-005056bbc17d",
          "name": "vol1",
          "_links": {
            "self": {
              "href": "/api/storage/volumes/71cd0dba-2a1c-11e9-b682-005056bbc17d"
            }
          }
        }
      },
      "enabled": true,
      "os_type": "linux",
      "space": {
        "block_size": 4096,
        "size": 322122547200,
        "used": 0,
        "guarantee": {
          "requested": false,
          "reserved": false
        }
      },
      "status": {
        "container_state": "online",
        "read_only": false,
        "state": "online"
      },
      "_links": {
        "self": {

```

```
        "href": "/api/storage/namespaces/dccdc3e6-cf4e-498f-bec6-
f7897f945669"
      }
    }
  }
]
}
```

### Updating an NVMe namespace comment

This example sets the `comment` property of an NVMe namespace.

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

# The call:
```

### Updating the size of an NVMe namespace

This example increases the size of an NVMe namespace.

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

# The call:
curl -X PATCH 'https://<mgmt-ip>/api/storage/namespaces/dccdc3e6-cf4e-
498f-bec6-f7897f945669' -H 'Accept: application/hal+json' -d '{ "space": {
"size": "1073741824" } }'
```

### Retrieving NVMe namespaces

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

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

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

```
# The response:
{
  "records": [
    {
      "uuid": "5c254d22-96a6-42ac-aad8-0cd9ebd126b6",
      "svm": {
        "name": "svm1"
      },
      "name": "/vol/vol1/namespace2",
      "status": {
        "state": "online"
      },
      "_links": {
        "self": {
          "href": "/api/storage/namespaces/5c254d22-96a6-42ac-aad8-0cd9ebd126b6"
        }
      }
    },
    {
      "uuid": "dccdc3e6-cf4e-498f-bec6-f7897f945669",
      "svm": {
        "name": "svm1"
      },
      "name": "/vol/vol1/namespace1",
      "status": {
        "state": "online"
      },
      "_links": {
        "self": {
          "href": "/api/storage/namespaces/dccdc3e6-cf4e-498f-bec6-f7897f945669"
        }
      }
    },
    {
      "uuid": "be732687-20cf-47d2-a0e2-2a989d15661d",
      "svm": {
        "name": "svm1"
      },
      "name": "/vol/vol2/namespace3",
      "status": {
        "state": "online"
      },
      "_links": {
```

```

    "self": {
      "href": "/api/storage/namespaces/be732687-20cf-47d2-a0e2-
2a989d15661d"
    }
  }
],
"num_records": 3,
"_links": {
  "self": {
    "href": "/api/storage/namespaces?svm.name=svml&status.state=online"
  }
}
}

```

### Retrieving details for a specific NVMe namespace

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 NVMe namespace.

```

# The API:
GET /api/storage/namespaces/{uuid}

# The call:
curl -X GET 'https://<mgmt-ip>/api/storage/namespaces/dccdc3e6-cf4e-498f-
bec6-f7897f945669?fields=**' -H 'Accept: application/hal+json'

# The response:
{
  "uuid": "dccdc3e6-cf4e-498f-bec6-f7897f945669",
  "svm": {
    "uuid": "6bf967fd-2a1c-11e9-b682-005056bbc17d",
    "name": "svml",
    "_links": {
      "self": {
        "href": "/api/svm/svms/6bf967fd-2a1c-11e9-b682-005056bbc17d"
      }
    }
  },
  "name": "/vol/vol1/namespacel",
  "location": {
    "namespace": "namespacel",
    "volume": {
      "uuid": "71cd0dba-2a1c-11e9-b682-005056bbc17d",

```

```

    "name": "voll1",
    "_links": {
      "self": {
        "href": "/api/storage/volumes/71cd0dba-2a1c-11e9-b682-005056bbc17d"
      }
    }
  },
  "auto_delete": false,
  "enabled": true,
  "comment": "Data for the research department.",
  "os_type": "linux",
  "space": {
    "block_size": 4096,
    "size": 322122547200,
    "used": 0,
    "guarantee": {
      "requested": false,
      "reserved": false
    }
  },
  "status": {
    "container_state": "online",
    "mapped": true,
    "read_only": false,
    "state": "online"
  },
  "subsystem_map": {
    "nsid": "00000001h",
    "anagrpid": "00000001h",
    "subsystem": {
      "uuid": "01f17d05-2be9-11e9-bed2-005056bbc17d",
      "name": "subsystem1",
      "_links": {
        "self": {
          "href": "/api/protocols/nvme/subsystems/01f17d05-2be9-11e9-bed2-005056bbc17d"
        }
      }
    }
  },
  "_links": {
    "self": {
      "href": "/api/protocols/nvme/subsystem-maps/dccdc3e6-cf4e-498f-bec6-f7897f945669/01f17d05-2be9-11e9-bed2-005056bbc17d"
    }
  }
}

```



```
    }  
  },  
  "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,  
      "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,  
      "total": 0  
    }  
  },  
  "_links": {
```

```
"self": {
  "href": "/api/storage/namespaces/dccdc3e6-cf4e-498f-bec6-
f7897f945669?fields=**"
}
}
}
```

## Cloning NVMe namespaces

A clone of an NVMe namespace is an independent "copy" of the namespace that shares unchanged data blocks with the original. As blocks of the source and clone are modified, unique blocks are written for each. NVMe namespace 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.

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

### Creating a new NVMe namespace clone

You create an NVMe namespace clone as you create any NVMe namespace — a POST to [/storage/namespaces](#). Set `clone.source.uuid` or `clone.source.name` to identify the source NVMe namespace from which the clone is created. The NVMe namespace clone and its source must reside in the same volume.

The source NVMe namespace 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/namespacel`.

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

# The call:
curl -X POST 'https://<mgmt-ip>/api/storage/namespaces' -H 'Accept:
application/hal+json' -d '{ "svm": { "name": "svm1" }, "name":
"/vol/vol1/namespace2clone1", "clone": { "source": { "name":
"/vol/vol1/namespace2" } } }'
```

### Over-writing an existing NVMe namespace's data as a clone of another

You can over-write an existing NVMe namespace as a clone of another. You do this as a PATCH on the NVMe namespace to overwrite — a PATCH to [/storage/namespaces/{uuid}](#). Set the `clone.source.uuid` or `clone.source.name` property to identify the source NVMe namespace from which the clone data is taken. The NVMe namespace clone and its source must reside in the same volume.

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

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

# The call:
curl -X PATCH 'https://<mgmt-ip>/api/storage/namespaces/dccdc3e6-cf4e-498f-bec6-f7897f945669' -H 'Accept: application/hal+json' -d '{ "clone": { "source": { "name": "/vol/vol1/namespace2" } } }'
```

## Converting a LUN into an NVMe namespace

An existing LUN can be converted in-place to an NVMe namespace with no modification to the data blocks. In other words, there is no additional copy created for the data blocks. There are certain requirements when converting a LUN to an NVMe namespace. For instance, the LUN should not be mapped to an initiator group, or exist as a protocol endpoint LUN, or in a foreign LUN import relationship. If the LUN exists as a VM volume, it should not be bound to a protocol endpoint LUN. Furthermore, only LUN with a supported operating system type for NVMe namespace can be converted.

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

### Convert a LUN into an NVMe namespace

You convert a LUN into an NVMe namespace by calling a POST to [/storage/namespaces](#). Set `convert.lun.uuid` or `convert.lun.name` to identify the source LUN which is to be converted in-place into an NVMe namespace.

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

# The call:
curl -X POST 'https://<mgmt-ip>/api/storage/namespaces' -H 'Accept: application/hal+json' -d '{ "svm": { "name": "svm1" }, "convert": { "lun": { "name": "/vol/vol1/lun1" } } }'
```

## Deleting an NVMe namespace

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

# The call:
curl -X DELETE 'https://<mgmt-ip>/api/storage/namespaces/5c254d22-96a6-42ac-aad8-0cd9ebd126b6' -H 'Accept: application/hal+json'
```

## Retrieve NVMe namespaces

GET /storage/namespaces

**Introduced In:** 9.6

Retrieves NVMe namespaces.

### Expensive properties

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

- `auto_delete`
- `subsystem_map.*`
- `status.mapped`
- `statistics.*`
- `metric.*`

### Related ONTAP commands

- `vserver nvme namespace show`
- `vserver nvme subsystem map show`

### Learn more

- [DOC /storage/namespaces](#) to learn more and examples.

### Parameters

Name	Type	In	Required	Description
uuid	string	query	False	Filter by uuid

Name	Type	In	Required	Description
statistics.status	string	query	False	Filter by statistics.status  • Introduced in: 9.8
statistics.iops_raw.total	integer	query	False	Filter by statistics.iops_raw.total  • Introduced in: 9.8
statistics.iops_raw.other	integer	query	False	Filter by statistics.iops_raw.other  • Introduced in: 9.8
statistics.iops_raw.write	integer	query	False	Filter by statistics.iops_raw.write  • Introduced in: 9.8
statistics.iops_raw.read	integer	query	False	Filter by statistics.iops_raw.read  • Introduced in: 9.8
statistics.timestamp	string	query	False	Filter by statistics.timestamp  • Introduced in: 9.8
statistics.latency_raw.total	integer	query	False	Filter by statistics.latency_raw.total  • Introduced in: 9.8

Name	Type	In	Required	Description
statistics.latency_raw.other	integer	query	False	Filter by statistics.latency_raw.other  • Introduced in: 9.8
statistics.latency_raw.write	integer	query	False	Filter by statistics.latency_raw.write  • Introduced in: 9.8
statistics.latency_raw.read	integer	query	False	Filter by statistics.latency_raw.read  • Introduced in: 9.8
statistics.throughput_raw.total	integer	query	False	Filter by statistics.throughput_raw.total  • Introduced in: 9.8
statistics.throughput_raw.write	integer	query	False	Filter by statistics.throughput_raw.write  • Introduced in: 9.8
statistics.throughput_raw.read	integer	query	False	Filter by statistics.throughput_raw.read  • Introduced in: 9.8
create_time	string	query	False	Filter by create_time  • Introduced in: 9.7
enabled	boolean	query	False	Filter by enabled

Name	Type	In	Required	Description
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
metric.timestamp	string	query	False	Filter by metric.timestamp  • Introduced in: 9.8
metric.iops.total	integer	query	False	Filter by metric.iops.total  • Introduced in: 9.8
metric.iops.other	integer	query	False	Filter by metric.iops.other  • Introduced in: 9.8
metric.iops.write	integer	query	False	Filter by metric.iops.write  • Introduced in: 9.8
metric.iops.read	integer	query	False	Filter by metric.iops.read  • Introduced in: 9.8
metric.duration	string	query	False	Filter by metric.duration  • Introduced in: 9.8

Name	Type	In	Required	Description
metric.status	string	query	False	Filter by metric.status  • Introduced in: 9.8
metric.latency.total	integer	query	False	Filter by metric.latency.total  • Introduced in: 9.8
metric.latency.other	integer	query	False	Filter by metric.latency.other  • Introduced in: 9.8
metric.latency.write	integer	query	False	Filter by metric.latency.write  • Introduced in: 9.8
metric.latency.read	integer	query	False	Filter by metric.latency.read  • Introduced in: 9.8
metric.throughput.total	integer	query	False	Filter by metric.throughput.total  • Introduced in: 9.8
metric.throughput.write	integer	query	False	Filter by metric.throughput.write  • Introduced in: 9.8



Name	Type	In	Required	Description
metric.throughput.read	integer	query	False	Filter by metric.throughput.read  • Introduced in: 9.8
svm.uuid	string	query	False	Filter by svm.uuid
svm.name	string	query	False	Filter by svm.name
comment	string	query	False	Filter by comment  • maxLength: 254 • minLength: 0
space.size	integer	query	False	Filter by space.size  • Max value: 140737488355328 • Min value: 4096
space.guarantee.requested	boolean	query	False	Filter by space.guarantee.requested
space.guarantee.reserved	boolean	query	False	Filter by space.guarantee.reserved
space.block_size	integer	query	False	Filter by space.block_size
space.used	integer	query	False	Filter by space.used
auto_delete	boolean	query	False	Filter by auto_delete
name	string	query	False	Filter by name
subsystem_map.nsid	string	query	False	Filter by subsystem_map.nsid

Name	Type	In	Required	Description
subsystem_map.subsystem.uuid	string	query	False	Filter by subsystem_map.subsystem.uuid
subsystem_map.subsystem.name	string	query	False	Filter by subsystem_map.subsystem.name <ul style="list-style-type: none"> <li>• maxLength: 64</li> <li>• minLength: 1</li> </ul>
subsystem_map.anagrp.id	string	query	False	Filter by subsystem_map.anagrp.id
location.node.name	string	query	False	Filter by location.node.name <ul style="list-style-type: none"> <li>• Introduced in: 9.10</li> </ul>
location.node.uuid	string	query	False	Filter by location.node.uuid <ul style="list-style-type: none"> <li>• Introduced in: 9.10</li> </ul>
location.qtree.name	string	query	False	Filter by location.qtree.name
location.qtree.id	integer	query	False	Filter by location.qtree.id <ul style="list-style-type: none"> <li>• Max value: 4994</li> <li>• Min value: 0</li> </ul>
location.volume.name	string	query	False	Filter by location.volume.name
location.volume.uuid	string	query	False	Filter by location.volume.uuid
location.namespace	string	query	False	Filter by location.namespace
os_type	string	query	False	Filter by os_type

Name	Type	In	Required	Description
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. <ul style="list-style-type: none"> <li>• Default value: 1</li> </ul>
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. <ul style="list-style-type: none"> <li>• Max value: 120</li> <li>• Min value: 0</li> <li>• Default value: 1</li> </ul>
order_by	array[string]	query	False	Order results by specified fields and optional [asc

## Response

Status: 200, Ok

Name	Type	Description
_links	<a href="#">_links</a>	
num_records	integer	The number of records in the response.

Name	Type	Description
records	array[nvme_namespace]	

## Example response

```
{
  "_links": {
    "next": {
      "href": "/api/resourcelink"
    },
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "num_records": 1,
  "records": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "clone": {
      "source": {
        "name": "/vol/volume1/namespace1",
        "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
      }
    },
    "comment": "string",
    "convert": {
      "lun": {
        "name": "/vol/volume1/lun1",
        "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
      }
    },
    "create_time": "2018-06-04 15:00:00 -0400",
    "location": {
      "namespace": "namespace1",
      "node": {
        "_links": {
          "self": {
            "href": "/api/resourcelink"
          }
        },
        "name": "node1",
        "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
      },
      "qtree": {
        "_links": {
          "self": {
```

```

        "href": "/api/resourcelink"
      }
    },
    "id": 1,
    "name": "qt1"
  },
  "volume": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "name": "volume1",
    "uuid": "028baa66-41bd-11e9-81d5-00a0986138f7"
  }
},
"metric": {
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "duration": "PT15S",
  "iops": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "latency": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "status": "ok",
  "throughput": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "timestamp": "2017-01-25 06:20:13 -0500"
},
"name": "/vol/volume1/mtree1/namespace1",
"os_type": "aix",
"space": {
  "block_size": 512,
  "size": 1073741824,

```

```

    "used": 0
  },
  "statistics": {
    "iops_raw": {
      "read": 200,
      "total": 1000,
      "write": 100
    },
    "latency_raw": {
      "read": 200,
      "total": 1000,
      "write": 100
    },
    "status": "ok",
    "throughput_raw": {
      "read": 200,
      "total": 1000,
      "write": 100
    },
    "timestamp": "2017-01-25 06:20:13 -0500"
  },
  "status": {
    "container_state": "online",
    "state": "online"
  },
  "subsystem_map": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "anagrpid": "00103050h",
    "nsid": "00000001h",
    "subsystem": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "name": "subsystem1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    }
  },
  "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	<a href="#">returned_error</a>	

## 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	<a href="#">href</a>	
self	<a href="#">href</a>	

\_links

Name	Type	Description
self	<a href="#">href</a>	

source

The source NVMe namespace for a namespace 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 namespace.

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

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

Name	Type	Description
name	string	The fully qualified path name of the clone source NVMe namespace composed of a "/vol" prefix, the volume name, the (optional) qtree name and base name of the namespace. Valid in POST and PATCH.
uuid	string	The unique identifier of the clone source NVMe namespace. Valid in POST and PATCH.

clone

This sub-object is used in POST to create a new NVMe namespace as a clone of an existing namespace, or PATCH to overwrite an existing namespace as a clone of another. Setting a property in this sub-object indicates that a namespace clone is desired.

When used in a PATCH, the patched NVMe namespace's data is over-written as a clone of the source and the following properties are preserved from the patched namespace unless otherwise specified as

part of the PATCH: `auto_delete` (unless specified in the request), `subsystem_map`, `status.state`, and `uuid`.

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

#### `lun`

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

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

Name	Type	Description
<code>name</code>	<code>string</code>	The fully qualified path name of the source LUN composed of a "/vol" prefix, the volume name, the (optional) qtree name and base name of the LUN. Valid in POST.
<code>uuid</code>	<code>string</code>	The unique identifier of the source LUN. Valid in POST.

#### `convert`

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

Name	Type	Description
lun	<a href="#">lun</a>	The source LUN for convert operation. This can be specified using property <code>convert.lun.uuid</code> or <code>convert.lun.name</code> . If both properties are supplied, they must refer to the same LUN.  Valid in POST. A convert request from LUN to NVMe namespace cannot be combined with setting any other namespace properties. All other properties of the converted NVMe namespace comes from the source LUN.

#### node

The cluster node that hosts the NVMe namespace.

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
name	string	
uuid	string	

#### qtree

The qtree in which the NVMe namespace is optionally located. Valid in POST.

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.

NVMe namespaces do not support rename.

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
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 NVMe namespace is located. Valid in POST.

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.

NVMe namespaces do not support movement between volumes.

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
name	string	The name of the volume. This field cannot be specified in a POST or PATCH method.
uuid	string	Unique identifier for the volume. This corresponds to the instance-uuid that is exposed in the CLI and ONTAPI. It does not change due to a volume move. <ul style="list-style-type: none"><li>• example: 028baa66-41bd-11e9-81d5-00a0986138f7</li><li>• Introduced in: 9.6</li><li>• x-nullable: true</li></ul>

location

The location of the NVMe namespace within the ONTAP cluster. Valid in POST.

NVMe namespaces do not support rename, or movement between volumes.

Name	Type	Description
namespace	string	The base name component of the NVMe namespace. Valid in POST.  If properties <code>name</code> and <code>location.namespace</code> are specified in the same request, they must refer to the base name.  NVMe namespaces do not support rename.
node	<a href="#">node</a>	The cluster node that hosts the NVMe namespace.

Name	Type	Description
qtree	qtree	<p>The qtree in which the NVMe namespace is optionally located. Valid in POST.</p> <p>If properties <code>name</code> and <code>location.qtree.name</code> and/or <code>location.qtree.uuid</code> are specified in the same request, they must refer to the same qtree.</p> <p>NVMe namespaces do not support rename.</p>
volume	volume	<p>The volume in which the NVMe namespace is located. Valid in POST.</p> <p>If properties <code>name</code> and <code>location.volume.name</code> and/or <code>location.volume.uuid</code> are specified in the same request, they must refer to the same volume.</p> <p>NVMe namespaces do not support movement between volumes.</p>

## iops

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

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

## latency

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

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

## throughput

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

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

## metric

Performance numbers, such as IOPS latency and throughput

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
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:

Name	Type	Description
iops	<a href="#">iops</a>	The rate of I/O operations observed at the storage object.
latency	<a href="#">latency</a>	The round trip latency in microseconds observed at the storage object.
status	string	Any errors associated with the sample. For example, if the aggregation of data over multiple nodes fails then any of the partial errors might be returned, "ok" on success, or "error" on any 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	<a href="#">throughput</a>	The rate of throughput bytes per second observed at the storage object.
timestamp	string	The timestamp of the performance data.

#### guarantee

Properties that request and report the space guarantee for the NVMe namespace.

Name	Type	Description
requested	boolean	<p>The requested space reservation policy for the NVMe namespace. If <i>true</i>, a space reservation is requested for the namespace; if <i>false</i>, the namespace is thin provisioned. Guaranteeing a space reservation request for a namespace requires that the volume in which the namespace resides also be space reserved and that the fractional reserve for the volume be 100%.</p> <p>The space reservation policy for an NVMe namespace is determined by ONTAP.</p> <ul style="list-style-type: none"> <li>• readOnly: 1</li> <li>• Introduced in: 9.6</li> <li>• x-nullable: true</li> </ul>
reserved	boolean	<p>Reports if the NVMe namespace is space guaranteed.</p> <p>This property is <i>true</i> if a space guarantee is requested and the containing volume and aggregate support the request. This property is <i>false</i> if a space guarantee is not requested or if a space guarantee is requested and either the containing volume and aggregate do not support the request.</p>

#### space

The storage space related properties of the NVMe namespace.

Name	Type	Description
block_size	integer	<p>The size of blocks in the namespace in bytes.</p> <p>Valid in POST when creating an NVMe namespace that is not a clone of another. Disallowed in POST when creating a namespace clone. Valid in POST.</p>



Name	Type	Description
guarantee	guarantee	Properties that request and report the space guarantee for the NVMe namespace.
size	integer	<p>The total provisioned size of the NVMe namespace. Valid in POST and PATCH. The NVMe namespace size can be increased but not be made smaller using the REST interface.</p> <p>The maximum and minimum sizes listed here are the absolute maximum and absolute minimum sizes in bytes. The maximum size is variable with respect to large NVMe namespace support in ONTAP. If large namespaces are supported, the maximum size is 128 TB (140737488355328 bytes) and if not supported, the maximum size is just under 16 TB (17557557870592 bytes). The minimum size supported is always 4096 bytes.</p> <p>For more information, see <i>Size properties</i> in the <i>docs</i> section of the ONTAP REST API documentation.</p> <ul style="list-style-type: none"> <li>• example: 1073741824</li> <li>• format: int64</li> <li>• Max value: 140737488355328</li> <li>• Min value: 4096</li> <li>• Introduced in: 9.6</li> <li>• x-nullable: true</li> </ul>

Name	Type	Description
used	integer	<p>The amount of space consumed by the main data stream of the NVMe namespace.</p> <p>This value is the total space consumed in the volume by the NVMe namespace, including filesystem overhead, but excluding prefix and suffix streams. Due to internal filesystem overhead and the many ways NVMe filesystems and applications utilize blocks within a namespace, this value does not necessarily reflect actual consumption/availability from the perspective of the filesystem or application. Without specific knowledge of how the namespace blocks are utilized outside of ONTAP, this property should not be used and an indicator for an out-of-space condition.</p> <p>For more information, see <i>Size properties</i> in the <i>docs</i> section of the ONTAP REST API documentation.</p> <ul style="list-style-type: none"> <li>• format: int64</li> <li>• readOnly: 1</li> <li>• Introduced in: 9.6</li> <li>• x-nullable: true</li> </ul>

iops\_raw

The number of I/O operations observed at the storage object. This should 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.

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

#### latency\_raw

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

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

#### throughput\_raw

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

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

## 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	<a href="#">iops_raw</a>	The number of I/O operations observed at the storage object. This should be used along with delta time to calculate the rate of I/O operations per unit of time.
latency_raw	<a href="#">latency_raw</a>	The raw latency in microseconds observed at the storage object. This should be divided by the raw IOPS value to calculate the average latency per I/O operation.
status	string	Any errors associated with the sample. For example, if the aggregation of data over multiple nodes fails then any of the partial errors might be returned, "ok" on success, or "error" on any 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	<a href="#">throughput_raw</a>	Throughput bytes observed at the storage object. This should be used along with delta time to calculate the rate of throughput bytes per unit of time.

Name	Type	Description
timestamp	string	The timestamp of the performance data.

status

Status information about the NVMe namespace.

Name	Type	Description
container_state	string	The state of the volume and aggregate that contain the NVMe namespace. Namespaces are only available when their containers are available.
mapped	boolean	Reports if the NVMe namespace is mapped to an NVMe subsystem.  There is an added computational 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 <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.
read_only	boolean	Reports if the NVMe namespace allows only read access.
state	string	The state of the NVMe namespace. Normal states for a namespace are <i>online</i> and <i>offline</i> . Other states indicate errors.

subsystem

The NVMe subsystem to which the NVMe namespace is mapped.

Name	Type	Description
<code>_links</code>	<a href="#">_links</a>	
name	string	The name of the NVMe subsystem.

Name	Type	Description
uuid	string	The unique identifier of the NVMe subsystem.

#### subsystem\_map

The NVMe subsystem with which the NVMe namespace is associated. A namespace can be mapped to zero (0) or one (1) subsystems.

There is an added computational cost to retrieving property values for `subsystem_map`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See [Requesting specific fields](#) to learn more.

Name	Type	Description
<code>_links</code>	<a href="#">_links</a>	
anagrpId	string	The Asymmetric Namespace Access Group ID (ANAGRPID) of the NVMe namespace.  The format for an ANAGRPID is 8 hexadecimal digits (zero-filled) followed by a lower case "h".
nsid	string	The NVMe namespace identifier. This is an identifier used by an NVMe controller to provide access to the NVMe namespace.  The format for an NVMe namespace identifier is 8 hexadecimal digits (zero-filled) followed by a lower case "h".
subsystem	<a href="#">subsystem</a>	The NVMe subsystem to which the NVMe namespace is mapped.

#### svm

SVM, applies only to SVM-scoped objects.

Name	Type	Description
<code>_links</code>	<a href="#">_links</a>	
name	string	The name of the SVM. This field cannot be specified in a PATCH method.

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

### nvme\_namespace

An NVMe namespace is a collection of addressable logical blocks presented to hosts connected to the storage virtual machine using the NVMe over Fabrics protocol.

In ONTAP, an NVMe namespace is located within a volume. Optionally, it can be located within a qtree in a volume.

An NVMe namespace is created to a specified size using thin or thick provisioning as determined by the volume on which it is created. NVMe namespaces support being cloned. An NVMe namespace cannot be renamed, resized, or moved to a different volume. NVMe namespaces do not support the assignment of a QoS policy for performance management, but a QoS policy can be assigned to the volume containing the namespace. See the NVMe namespace object model to learn more about each of the properties supported by the NVMe namespace REST API.

An NVMe namespace must be mapped to an NVMe subsystem to grant access to the subsystem's hosts. Hosts can then access the NVMe namespace and perform I/O using the NVMe over Fabrics protocol.

Name	Type	Description
_links	<a href="#">_links</a>	

Name	Type	Description
auto_delete	boolean	<p>This property marks the NVMe namespace for auto deletion when the volume containing the namespace runs out of space. This is most commonly set on namespace clones.</p> <p>When set to <i>true</i>, the NVMe namespace becomes eligible for automatic deletion when the volume runs out of space. Auto deletion only occurs when the volume containing the namespace is also configured for auto deletion and free space in the volume decreases below a particular threshold.</p> <p>This property is optional in POST and PATCH. The default value for a new NVMe namespace is <i>false</i>.</p> <p>There is an added computational 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 <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.</p>



Name	Type	Description
clone	clone	<p>This sub-object is used in POST to create a new NVMe namespace as a clone of an existing namespace, or PATCH to overwrite an existing namespace as a clone of another. Setting a property in this sub-object indicates that a namespace clone is desired.</p> <p>When used in a PATCH, the patched NVMe namespace's data is over-written as a clone of the source and the following properties are preserved from the patched namespace unless otherwise specified as part of the PATCH: <code>auto_delete</code> (unless specified in the request), <code>subsystem_map</code>, <code>status.state</code>, and <code>uuid</code>.</p>
comment	string	A configurable comment available for use by the administrator. Valid in POST and PATCH.
convert	convert	This sub-object is used in POST to convert a valid in-place LUN to an NVMe namespace. Setting a property in this sub-object indicates that a conversion from the specified LUN to NVMe namespace is desired.
create_time	string	The time the NVMe namespace was created.
enabled	boolean	The enabled state of the NVMe namespace. Certain error conditions cause the namespace to become disabled. If the namespace is disabled, you can check the <code>state</code> property to determine what error disabled the namespace. An NVMe namespace is enabled automatically when it is created.

Name	Type	Description
location	location	<p>The location of the NVMe namespace within the ONTAP cluster. Valid in POST.</p> <p>NVMe namespaces do not support rename, or movement between volumes.</p> <ul style="list-style-type: none"> <li>• Introduced in: 9.6</li> <li>• readCreate: 1</li> </ul>
metric	metric	Performance numbers, such as IOPS latency and throughput
name	string	<p>The fully qualified path name of the NVMe namespace composed of a "/vol" prefix, the volume name, the (optional) qtree name and base name of the namespace. Valid in POST.</p> <p>NVMe namespaces do not support rename, or movement between volumes.</p>
os_type	string	<p>The operating system type of the NVMe namespace.</p> <p>Required in POST when creating an NVMe namespace that is not a clone of another. Disallowed in POST when creating a namespace clone.</p>
space	space	The storage space related properties of the NVMe namespace.
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 NVMe namespace.

Name	Type	Description
subsystem_map	<a href="#">subsystem_map</a>	<p>The NVMe subsystem with which the NVMe namespace is associated. A namespace can be mapped to zero (0) or one (1) subsystems.</p> <p>There is an added computational cost to retrieving property values for <code>subsystem_map</code>. They are not populated for either a collection GET or an instance GET unless explicitly requested using the <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.</p>
svm	<a href="#">svm</a>	SVM, applies only to SVM-scoped objects.
uuid	string	The unique identifier of the NVMe namespace.

#### error\_arguments

Name	Type	Description
code	string	Argument code
message	string	Message argument

#### returned\_error

Name	Type	Description
arguments	array[ <a href="#">error_arguments</a> ]	Message arguments
code	string	Error code
message	string	Error message
target	string	The target parameter that caused the error.

# Create an NVMe namespace

POST /storage/namespaces

Introduced In: 9.6

Creates an NVMe namespace.

## Required properties

- `svm.uuid` or `svm.name` - Existing SVM in which to create the NVMe namespace.
- `name`, `location.volume.name` or `location.volume.uuid` - Existing volume in which to create the NVMe namespace.
- `name` or `location.namespace` - Base name for the NVMe namespace.
- `os_type` - Operating system from which the NVMe namespace will be accessed. (Not used for clones, which are created based on the `os_type` of the source NVMe namespace.)
- `space.size` - Size for the NVMe namespace. (Not used for clones, which are created based on the size of the source NVMe namespace.)

## Default property values

If not specified in POST, the following default property values are assigned:

- `auto_delete` - *false*
- `space.block_size` - *4096* ( *512* when 'os\_type' is *vmware* )

## Related ONTAP commands

- `volume file clone autodelete`
- `volume file clone create`
- `vserver nvme namespace convert-from-lun`
- `vserver nvme namespace create`

## Learn more

- [DOC /storage/namespaces](#)

## Parameters

Name	Type	In	Required	Description
<code>return_records</code>	boolean	query	False	The default is false. If set to true, the records are returned. <ul style="list-style-type: none"><li>• Default value:</li></ul>

## Request Body

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
auto_delete	boolean	<p>This property marks the NVMe namespace for auto deletion when the volume containing the namespace runs out of space. This is most commonly set on namespace clones.</p> <p>When set to <i>true</i>, the NVMe namespace becomes eligible for automatic deletion when the volume runs out of space. Auto deletion only occurs when the volume containing the namespace is also configured for auto deletion and free space in the volume decreases below a particular threshold.</p> <p>This property is optional in POST and PATCH. The default value for a new NVMe namespace is <i>false</i>.</p> <p>There is an added computational 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 <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.</p>

Name	Type	Description
clone	clone	<p>This sub-object is used in POST to create a new NVMe namespace as a clone of an existing namespace, or PATCH to overwrite an existing namespace as a clone of another. Setting a property in this sub-object indicates that a namespace clone is desired.</p> <p>When used in a PATCH, the patched NVMe namespace's data is over-written as a clone of the source and the following properties are preserved from the patched namespace unless otherwise specified as part of the PATCH: <code>auto_delete</code> (unless specified in the request), <code>subsystem_map</code>, <code>status.state</code>, and <code>uuid</code>.</p>
comment	string	A configurable comment available for use by the administrator. Valid in POST and PATCH.
convert	convert	This sub-object is used in POST to convert a valid in-place LUN to an NVMe namespace. Setting a property in this sub-object indicates that a conversion from the specified LUN to NVMe namespace is desired.
create_time	string	The time the NVMe namespace was created.
enabled	boolean	The enabled state of the NVMe namespace. Certain error conditions cause the namespace to become disabled. If the namespace is disabled, you can check the <code>state</code> property to determine what error disabled the namespace. An NVMe namespace is enabled automatically when it is created.

Name	Type	Description
location	<a href="#">location</a>	<p>The location of the NVMe namespace within the ONTAP cluster. Valid in POST.</p> <p>NVMe namespaces do not support rename, or movement between volumes.</p> <ul style="list-style-type: none"> <li>• Introduced in: 9.6</li> <li>• readCreate: 1</li> </ul>
metric	<a href="#">metric</a>	Performance numbers, such as IOPS latency and throughput
name	string	<p>The fully qualified path name of the NVMe namespace composed of a "/vol" prefix, the volume name, the (optional) qtree name and base name of the namespace. Valid in POST.</p> <p>NVMe namespaces do not support rename, or movement between volumes.</p>
os_type	string	<p>The operating system type of the NVMe namespace.</p> <p>Required in POST when creating an NVMe namespace that is not a clone of another. Disallowed in POST when creating a namespace clone.</p>
space	<a href="#">space</a>	The storage space related properties of the NVMe namespace.
statistics	<a href="#">statistics</a>	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	<a href="#">status</a>	Status information about the NVMe namespace.

Name	Type	Description
subsystem_map	<a href="#">subsystem_map</a>	<p>The NVMe subsystem with which the NVMe namespace is associated. A namespace can be mapped to zero (0) or one (1) subsystems.</p> <p>There is an added computational cost to retrieving property values for <code>subsystem_map</code>. They are not populated for either a collection GET or an instance GET unless explicitly requested using the <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.</p>
svm	<a href="#">svm</a>	SVM, applies only to SVM-scoped objects.
uuid	string	The unique identifier of the NVMe namespace.



## Example request

```
{
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "clone": {
    "source": {
      "name": "/vol/volume1/namespace1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    }
  },
  "comment": "string",
  "convert": {
    "lun": {
      "name": "/vol/volume1/lun1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    }
  },
  "create_time": "2018-06-04 15:00:00 -0400",
  "location": {
    "namespace": "namespace1",
    "node": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "name": "node1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    },
    "qtree": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "id": 1,
      "name": "qt1"
    },
    "volume": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      }
    }
  }
}
```

```

    }
  },
  "name": "volume1",
  "uuid": "028baa66-41bd-11e9-81d5-00a0986138f7"
}
},
"metric": {
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "duration": "PT15S",
  "iops": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "latency": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "status": "ok",
  "throughput": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "timestamp": "2017-01-25 06:20:13 -0500"
},
"name": "/vol/volume1/qtree1/namespace1",
"os_type": "aix",
"space": {
  "block_size": 512,
  "size": 1073741824,
  "used": 0
},
"statistics": {
  "iops_raw": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "latency_raw": {
    "read": 200,

```

```
    "total": 1000,
    "write": 100
  },
  "status": "ok",
  "throughput_raw": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "timestamp": "2017-01-25 06:20:13 -0500"
},
"status": {
  "container_state": "online",
  "state": "online"
},
"subsystem_map": {
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "anagrpid": "00103050h",
  "nsid": "00000001h",
  "subsystem": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "name": "subsystem1",
    "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
  }
},
"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
<a href="#">_links</a>	<a href="#">_links</a>	
num_records	integer	The number of records in the response.
records	array[ <a href="#">nvme_namespace</a> ]	

## Example response

```
{
  "_links": {
    "next": {
      "href": "/api/resourcelink"
    },
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "num_records": 1,
  "records": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "clone": {
      "source": {
        "name": "/vol/volume1/namespace1",
        "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
      }
    },
    "comment": "string",
    "convert": {
      "lun": {
        "name": "/vol/volume1/lun1",
        "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
      }
    },
    "create_time": "2018-06-04 15:00:00 -0400",
    "location": {
      "namespace": "namespace1",
      "node": {
        "_links": {
          "self": {
            "href": "/api/resourcelink"
          }
        },
        "name": "node1",
        "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
      },
      "qtree": {
        "_links": {
          "self": {
```

```

        "href": "/api/resourcelink"
      }
    },
    "id": 1,
    "name": "qt1"
  },
  "volume": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "name": "volume1",
    "uuid": "028baa66-41bd-11e9-81d5-00a0986138f7"
  }
},
"metric": {
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "duration": "PT15S",
  "iops": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "latency": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "status": "ok",
  "throughput": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "timestamp": "2017-01-25 06:20:13 -0500"
},
"name": "/vol/volume1/mtree1/namespace1",
"os_type": "aix",
"space": {
  "block_size": 512,
  "size": 1073741824,

```

```

    "used": 0
  },
  "statistics": {
    "iops_raw": {
      "read": 200,
      "total": 1000,
      "write": 100
    },
    "latency_raw": {
      "read": 200,
      "total": 1000,
      "write": 100
    },
    "status": "ok",
    "throughput_raw": {
      "read": 200,
      "total": 1000,
      "write": 100
    },
    "timestamp": "2017-01-25 06:20:13 -0500"
  },
  "status": {
    "container_state": "online",
    "state": "online"
  },
  "subsystem_map": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "anagrpid": "00103050h",
    "nsid": "00000001h",
    "subsystem": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "name": "subsystem1",
      "uuid": "1cd8a442-86d1-11e0-aelc-123478563412"
    }
  },
  "svm": {
    "_links": {
      "self": {

```

```

        "href": "/api/resourcelink"
      }
    },
    "name": "svm1",
    "uuid": "02c9e252-41be-11e9-81d5-00a0986138f7"
  },
  "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
}
}

```

## Headers

Name	Description	Type
Location	Useful for tracking the resource location	string

## Error

Status: Default

## ONTAP Error Response Codes

Error Code	Description
917927	The specified volume was not found.
918236	The specified <code>location.volume.uuid</code> and <code>location.volume.name</code> do not refer to the same volume.
1254197	The LUN specified for conversion to a namespace is mapped.
2621462	The supplied SVM does not exist.
2621706	The specified <code>svm.uuid</code> and <code>svm.name</code> do not refer to the same SVM.
2621707	No SVM was specified. Either <code>svm.name</code> or <code>svm.uuid</code> must be supplied.
5242927	The specified <code>qtree</code> was not found.
5242950	The specified <code>location.qtree.id</code> and <code>location.qtree.name</code> do not refer to the same <code>qtree</code> .
5374127	An invalid namespace name was specified.
5374140	LUN has a non-zero prefix and/or suffix size.



Error Code	Description
5374141	LUN is part of a SnapMirror Business Continuity (SMBC) relationship.
5374156	A protocol endpoint LUN cannot be converted to an NVMe namespace.
5374157	LUN in an SVM with MetroCluster configured cannot be converted to an NVMe namespace.
5374158	LUN contains an operating system type that is not supported for NVMe namespace.
5374352	An invalid name was provided for the NVMe namespace.
5374858	The volume specified by <code>name</code> is not the same as that specified by <code>location.volume</code> .
5374860	The <code>qtree</code> specified by <code>name</code> is not the same as that specified by <code>location.qtree</code> .
5374861	The NVMe namespace base name specified by <code>name</code> is not the same as that specified by <code>location.name</code> .
5374862	No NVMe namespace path base name was provided for the namespace.
5374876	The LUN specified for conversion to a namespace was not found.
13565952	The NVMe namespace clone request failed.
72089636	Creating NVMe namespaces with <code>os_type</code> AIX is not supported until the effective cluster version is 9.13.1 or later.
72089720	NVMe namespaces cannot be created in Snapshot copies.
72089721	The volume specified is in a load sharing mirror relationship. Namespaces are not supported in load sharing mirrors.
72089722	A negative size was provided for the NVMe namespace.
72089723	The specified size is too small for the NVMe namespace.
72089724	The specified size is too large for the NVMe namespace.
72089725	A LUN or NVMe namespace already exists at the specified path.
72089727	NVMe namespaces cannot be created on an SVM root volume.
72089728	NVMe namespaces cannot be created on a FlexGroup volume.

Error Code	Description
72089732	An NVMe namespace name can only contain characters A-Z, a-z, 0-9, "-", ".", "_", "{" and "}".
72090005	The specified <code>clone.source.uuid</code> and <code>clone.source.name</code> do not refer to the same NVMe namespace.
72090006	The specified <code>clone.source</code> was not found.
72090007	The specified <code>clone.source</code> was not found.
72090009	An error occurred after successfully creating the NVMe namespace. Some properties were not set.
72090012	The property cannot be specified when creating an NVMe namespace clone. The <code>target</code> property of the error object identifies the property.
72090013	The property is required except when creating an NVMe namespace clone. The <code>target</code> property of the error object identifies the property.
72090014	No volume was specified for the NVMe namespace.
72090015	An error occurred after successfully creating the NVMe namespace preventing the retrieval of its properties.
72090033	The <code>clone.source.uuid</code> property is not supported when specifying a source NVMe namespace from a Snapshot copy.
72090039	The property cannot be specified at the same time when creating an NVMe namespace as a clone. The <code>target</code> property of the error object identifies the other property given with <code>clone</code> .
72090040	The property cannot be specified when converting a LUN into an NVMe namespace. The <code>target</code> property of the error object identifies the property.

Also see the table of common errors in the [Response body](#) overview section of this documentation.

Name	Type	Description
error	<a href="#">returned_error</a>	

## 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	<a href="#">href</a>	

source

The source NVMe namespace for a namespace 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 namespace.

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

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

Name	Type	Description
name	string	The fully qualified path name of the clone source NVMe namespace composed of a "/vol" prefix, the volume name, the (optional) qtree name and base name of the namespace. Valid in POST and PATCH.
uuid	string	The unique identifier of the clone source NVMe namespace. Valid in POST and PATCH.

clone

This sub-object is used in POST to create a new NVMe namespace as a clone of an existing namespace, or PATCH to overwrite an existing namespace as a clone of another. Setting a property in this sub-object indicates that a namespace clone is desired.

When used in a PATCH, the patched NVMe namespace's data is over-written as a clone of the source and the following properties are preserved from the patched namespace unless otherwise specified as part of the PATCH: `auto_delete` (unless specified in the request), `subsystem_map`, `status.state`, and `uuid`.

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

#### lun

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

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

Name	Type	Description
name	string	The fully qualified path name of the source LUN composed of a "/vol" prefix, the volume name, the (optional) qtree name and base name of the LUN. Valid in POST.
uuid	string	The unique identifier of the source LUN. Valid in POST.

#### convert

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

Name	Type	Description
lun	<a href="#">lun</a>	The source LUN for convert operation. This can be specified using property <code>convert.lun.uuid</code> or <code>convert.lun.name</code> . If both properties are supplied, they must refer to the same LUN.  Valid in POST. A convert request from LUN to NVMe namespace cannot be combined with setting any other namespace properties. All other properties of the converted NVMe namespace comes from the source LUN.

#### node

The cluster node that hosts the NVMe namespace.

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
name	string	
uuid	string	

#### qtree

The qtree in which the NVMe namespace is optionally located. Valid in POST.

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.

NVMe namespaces do not support rename.

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
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 NVMe namespace is located. Valid in POST.

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.

NVMe namespaces do not support movement between volumes.

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
name	string	The name of the volume. This field cannot be specified in a POST or PATCH method.
uuid	string	Unique identifier for the volume. This corresponds to the instance-uuid that is exposed in the CLI and ONTAPI. It does not change due to a volume move. <ul style="list-style-type: none"><li>• example: 028baa66-41bd-11e9-81d5-00a0986138f7</li><li>• Introduced in: 9.6</li><li>• x-nullable: true</li></ul>

location

The location of the NVMe namespace within the ONTAP cluster. Valid in POST.

NVMe namespaces do not support rename, or movement between volumes.

Name	Type	Description
namespace	string	The base name component of the NVMe namespace. Valid in POST.  If properties <code>name</code> and <code>location.namespace</code> are specified in the same request, they must refer to the base name.  NVMe namespaces do not support rename.
node	<a href="#">node</a>	The cluster node that hosts the NVMe namespace.

Name	Type	Description
qtree	qtree	<p>The qtree in which the NVMe namespace is optionally located. Valid in POST.</p> <p>If properties <code>name</code> and <code>location.qtree.name</code> and/or <code>location.qtree.uuid</code> are specified in the same request, they must refer to the same qtree.</p> <p>NVMe namespaces do not support rename.</p>
volume	volume	<p>The volume in which the NVMe namespace is located. Valid in POST.</p> <p>If properties <code>name</code> and <code>location.volume.name</code> and/or <code>location.volume.uuid</code> are specified in the same request, they must refer to the same volume.</p> <p>NVMe namespaces do not support movement between volumes.</p>

## iops

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

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



## latency

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

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

## throughput

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

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

## metric

Performance numbers, such as IOPS latency and throughput

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
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:

Name	Type	Description
iops	<a href="#">iops</a>	The rate of I/O operations observed at the storage object.
latency	<a href="#">latency</a>	The round trip latency in microseconds observed at the storage object.
status	string	Any errors associated with the sample. For example, if the aggregation of data over multiple nodes fails then any of the partial errors might be returned, "ok" on success, or "error" on any 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	<a href="#">throughput</a>	The rate of throughput bytes per second observed at the storage object.
timestamp	string	The timestamp of the performance data.

#### guarantee

Properties that request and report the space guarantee for the NVMe namespace.

Name	Type	Description
requested	boolean	<p>The requested space reservation policy for the NVMe namespace. If <i>true</i>, a space reservation is requested for the namespace; if <i>false</i>, the namespace is thin provisioned. Guaranteeing a space reservation request for a namespace requires that the volume in which the namespace resides also be space reserved and that the fractional reserve for the volume be 100%.</p> <p>The space reservation policy for an NVMe namespace is determined by ONTAP.</p> <ul style="list-style-type: none"> <li>• readOnly: 1</li> <li>• Introduced in: 9.6</li> <li>• x-nullable: true</li> </ul>
reserved	boolean	<p>Reports if the NVMe namespace is space guaranteed.</p> <p>This property is <i>true</i> if a space guarantee is requested and the containing volume and aggregate support the request. This property is <i>false</i> if a space guarantee is not requested or if a space guarantee is requested and either the containing volume and aggregate do not support the request.</p>

## space

The storage space related properties of the NVMe namespace.

Name	Type	Description
block_size	integer	<p>The size of blocks in the namespace in bytes.</p> <p>Valid in POST when creating an NVMe namespace that is not a clone of another. Disallowed in POST when creating a namespace clone. Valid in POST.</p>

Name	Type	Description
guarantee	<a href="#">guarantee</a>	Properties that request and report the space guarantee for the NVMe namespace.
size	integer	<p>The total provisioned size of the NVMe namespace. Valid in POST and PATCH. The NVMe namespace size can be increased but not be made smaller using the REST interface.</p> <p>The maximum and minimum sizes listed here are the absolute maximum and absolute minimum sizes in bytes. The maximum size is variable with respect to large NVMe namespace support in ONTAP. If large namespaces are supported, the maximum size is 128 TB (140737488355328 bytes) and if not supported, the maximum size is just under 16 TB (17557557870592 bytes). The minimum size supported is always 4096 bytes.</p> <p>For more information, see <i>Size properties</i> in the <i>docs</i> section of the ONTAP REST API documentation.</p> <ul style="list-style-type: none"> <li>• example: 1073741824</li> <li>• format: int64</li> <li>• Max value: 140737488355328</li> <li>• Min value: 4096</li> <li>• Introduced in: 9.6</li> <li>• x-nullable: true</li> </ul>

Name	Type	Description
used	integer	<p>The amount of space consumed by the main data stream of the NVMe namespace.</p> <p>This value is the total space consumed in the volume by the NVMe namespace, including filesystem overhead, but excluding prefix and suffix streams. Due to internal filesystem overhead and the many ways NVMe filesystems and applications utilize blocks within a namespace, this value does not necessarily reflect actual consumption/availability from the perspective of the filesystem or application. Without specific knowledge of how the namespace blocks are utilized outside of ONTAP, this property should not be used and an indicator for an out-of-space condition.</p> <p>For more information, see <i>Size properties</i> in the <i>docs</i> section of the ONTAP REST API documentation.</p> <ul style="list-style-type: none"> <li>• format: int64</li> <li>• readOnly: 1</li> <li>• Introduced in: 9.6</li> <li>• x-nullable: true</li> </ul>

#### iops\_raw

The number of I/O operations observed at the storage object. This should 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.

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

#### latency\_raw

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

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

#### throughput\_raw

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

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

## 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	<a href="#">iops_raw</a>	The number of I/O operations observed at the storage object. This should be used along with delta time to calculate the rate of I/O operations per unit of time.
latency_raw	<a href="#">latency_raw</a>	The raw latency in microseconds observed at the storage object. This should be divided by the raw IOPS value to calculate the average latency per I/O operation.
status	string	Any errors associated with the sample. For example, if the aggregation of data over multiple nodes fails then any of the partial errors might be returned, "ok" on success, or "error" on any 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	<a href="#">throughput_raw</a>	Throughput bytes observed at the storage object. This should be used along with delta time to calculate the rate of throughput bytes per unit of time.

Name	Type	Description
timestamp	string	The timestamp of the performance data.

status

Status information about the NVMe namespace.

Name	Type	Description
container_state	string	The state of the volume and aggregate that contain the NVMe namespace. Namespaces are only available when their containers are available.
mapped	boolean	Reports if the NVMe namespace is mapped to an NVMe subsystem.  There is an added computational 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 <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.
read_only	boolean	Reports if the NVMe namespace allows only read access.
state	string	The state of the NVMe namespace. Normal states for a namespace are <i>online</i> and <i>offline</i> . Other states indicate errors.

subsystem

The NVMe subsystem to which the NVMe namespace is mapped.

Name	Type	Description
<code>_links</code>	<a href="#">_links</a>	
name	string	The name of the NVMe subsystem.



Name	Type	Description
uuid	string	The unique identifier of the NVMe subsystem.

#### subsystem\_map

The NVMe subsystem with which the NVMe namespace is associated. A namespace can be mapped to zero (0) or one (1) subsystems.

There is an added computational cost to retrieving property values for `subsystem_map`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See [Requesting specific fields](#) to learn more.

Name	Type	Description
<code>_links</code>	<a href="#">_links</a>	
anagrpId	string	The Asymmetric Namespace Access Group ID (ANAGRPID) of the NVMe namespace.  The format for an ANAGRPID is 8 hexadecimal digits (zero-filled) followed by a lower case "h".
nsid	string	The NVMe namespace identifier. This is an identifier used by an NVMe controller to provide access to the NVMe namespace.  The format for an NVMe namespace identifier is 8 hexadecimal digits (zero-filled) followed by a lower case "h".
subsystem	<a href="#">subsystem</a>	The NVMe subsystem to which the NVMe namespace is mapped.

#### svm

SVM, applies only to SVM-scoped objects.

Name	Type	Description
<code>_links</code>	<a href="#">_links</a>	
name	string	The name of the SVM. This field cannot be specified in a PATCH method.

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

### nvme\_namespace

An NVMe namespace is a collection of addressable logical blocks presented to hosts connected to the storage virtual machine using the NVMe over Fabrics protocol.

In ONTAP, an NVMe namespace is located within a volume. Optionally, it can be located within a qtree in a volume.

An NVMe namespace is created to a specified size using thin or thick provisioning as determined by the volume on which it is created. NVMe namespaces support being cloned. An NVMe namespace cannot be renamed, resized, or moved to a different volume. NVMe namespaces do not support the assignment of a QoS policy for performance management, but a QoS policy can be assigned to the volume containing the namespace. See the NVMe namespace object model to learn more about each of the properties supported by the NVMe namespace REST API.

An NVMe namespace must be mapped to an NVMe subsystem to grant access to the subsystem's hosts. Hosts can then access the NVMe namespace and perform I/O using the NVMe over Fabrics protocol.

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	

Name	Type	Description
auto_delete	boolean	<p>This property marks the NVMe namespace for auto deletion when the volume containing the namespace runs out of space. This is most commonly set on namespace clones.</p> <p>When set to <i>true</i>, the NVMe namespace becomes eligible for automatic deletion when the volume runs out of space. Auto deletion only occurs when the volume containing the namespace is also configured for auto deletion and free space in the volume decreases below a particular threshold.</p> <p>This property is optional in POST and PATCH. The default value for a new NVMe namespace is <i>false</i>.</p> <p>There is an added computational 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 <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.</p>

Name	Type	Description
clone	clone	<p>This sub-object is used in POST to create a new NVMe namespace as a clone of an existing namespace, or PATCH to overwrite an existing namespace as a clone of another. Setting a property in this sub-object indicates that a namespace clone is desired.</p> <p>When used in a PATCH, the patched NVMe namespace's data is over-written as a clone of the source and the following properties are preserved from the patched namespace unless otherwise specified as part of the PATCH: <code>auto_delete</code> (unless specified in the request), <code>subsystem_map</code>, <code>status.state</code>, and <code>uuid</code>.</p>
comment	string	A configurable comment available for use by the administrator. Valid in POST and PATCH.
convert	convert	This sub-object is used in POST to convert a valid in-place LUN to an NVMe namespace. Setting a property in this sub-object indicates that a conversion from the specified LUN to NVMe namespace is desired.
create_time	string	The time the NVMe namespace was created.
enabled	boolean	The enabled state of the NVMe namespace. Certain error conditions cause the namespace to become disabled. If the namespace is disabled, you can check the <code>state</code> property to determine what error disabled the namespace. An NVMe namespace is enabled automatically when it is created.

Name	Type	Description
location	location	<p>The location of the NVMe namespace within the ONTAP cluster. Valid in POST.</p> <p>NVMe namespaces do not support rename, or movement between volumes.</p> <ul style="list-style-type: none"> <li>• Introduced in: 9.6</li> <li>• readCreate: 1</li> </ul>
metric	metric	Performance numbers, such as IOPS latency and throughput
name	string	<p>The fully qualified path name of the NVMe namespace composed of a "/vol" prefix, the volume name, the (optional) qtree name and base name of the namespace. Valid in POST.</p> <p>NVMe namespaces do not support rename, or movement between volumes.</p>
os_type	string	<p>The operating system type of the NVMe namespace.</p> <p>Required in POST when creating an NVMe namespace that is not a clone of another. Disallowed in POST when creating a namespace clone.</p>
space	space	The storage space related properties of the NVMe namespace.
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 NVMe namespace.

Name	Type	Description
subsystem_map	<a href="#">subsystem_map</a>	<p>The NVMe subsystem with which the NVMe namespace is associated. A namespace can be mapped to zero (0) or one (1) subsystems.</p> <p>There is an added computational cost to retrieving property values for <code>subsystem_map</code>. They are not populated for either a collection GET or an instance GET unless explicitly requested using the <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.</p>
svm	<a href="#">svm</a>	SVM, applies only to SVM-scoped objects.
uuid	string	The unique identifier of the NVMe namespace.

#### `_links`

Name	Type	Description
next	<a href="#">href</a>	
self	<a href="#">href</a>	

#### `error_arguments`

Name	Type	Description
code	string	Argument code
message	string	Message argument

#### `returned_error`

Name	Type	Description
arguments	array[ <a href="#">error_arguments</a> ]	Message arguments
code	string	Error code
message	string	Error message

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

## Retrieve historical performance metrics for an NVMe namespace

GET /storage/namespaces/{nvme\_namespace.uuid}/metrics

Introduced In: 9.14

Retrieves historical performance metrics for an NVMe namespace.

### Related ONTAP commands

- `statistics namespace show`

### Parameters

Name	Type	In	Required	Description
timestamp	string	query	False	Filter by timestamp
status	string	query	False	Filter by status
iops.total	integer	query	False	Filter by iops.total
iops.other	integer	query	False	Filter by iops.other
iops.write	integer	query	False	Filter by iops.write
iops.read	integer	query	False	Filter by iops.read
duration	string	query	False	Filter by duration
latency.total	integer	query	False	Filter by latency.total
latency.other	integer	query	False	Filter by latency.other
latency.write	integer	query	False	Filter by latency.write
latency.read	integer	query	False	Filter by latency.read

Name	Type	In	Required	Description
uuid	string	query	False	Filter by uuid
throughput.total	integer	query	False	Filter by throughput.total
throughput.write	integer	query	False	Filter by throughput.write
throughput.read	integer	query	False	Filter by throughput.read
nvme_namespace.uuid	string	path	True	The unique identifier of the NVMe namespace.
interval	string	query	False	<p>The time range for the data. Examples can be 1h, 1d, 1m, 1w, 1y. The period for each time range is as follows:</p> <ul style="list-style-type: none"> <li>• 1h: Metrics over the most recent hour sampled over 15 seconds.</li> <li>• 1d: Metrics over the most recent day sampled over 5 minutes.</li> <li>• 1w: Metrics over the most recent week sampled over 30 minutes.</li> <li>• 1m: Metrics over the most recent month sampled over 2 hours.</li> <li>• 1y: Metrics over the most recent year sampled over a day.</li> <li>• Default value: 1</li> <li>• enum: ["1h", "1d", "1w", "1m", "1y"]</li> </ul>



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. <ul style="list-style-type: none"> <li>• Default value: 1</li> <li>• Max value: 120</li> <li>• Min value: 0</li> </ul>
fields	array[string]	query	False	Specify the fields to return.
max_records	integer	query	False	Limit the number of records returned.
order_by	array[string]	query	False	Order results by specified fields and optional [asc
desc] direction. Default direction is 'asc' for ascending.	return_records	boolean	query	False

## Response

Status: 200, Ok

Name	Type	Description
_links	<a href="#">_links</a>	
num_records	integer	Number of records
records	array[ <a href="#">records</a> ]	

## Example response

```
{
  "_links": {
    "next": {
      "href": "/api/resourcelink"
    },
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "num_records": 1,
  "records": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "duration": "PT15S",
    "iops": {
      "read": 200,
      "total": 1000,
      "write": 100
    },
    "latency": {
      "read": 200,
      "total": 1000,
      "write": 100
    },
    "status": "ok",
    "throughput": {
      "read": 200,
      "total": 1000,
      "write": 100
    },
    "timestamp": "2017-01-25 06:20:13 -0500",
    "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
  }
}
```

## Error

Status: Default

## ONTAP Error Response Codes

Error Code	Description
8585947	No metrics are available for the requested object.
8586225	An unexpected error occurred retrieving metrics for the requested object.

Also see the table of common errors in the [Response body](#) overview section of this documentation.

Name	Type	Description
error	<a href="#">returned_error</a>	

### 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	<a href="#">href</a>	
self	<a href="#">href</a>	

\_links

Name	Type	Description
self	<a href="#">href</a>	

iops

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

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

latency

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

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

### throughput

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

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

### records

Performance numbers, such as IOPS latency and throughput, for SVM protocols.

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
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	<a href="#">iops</a>	The rate of I/O operations observed at the storage object.

Name	Type	Description
latency	<a href="#">latency</a>	The round trip latency in microseconds observed at the storage object.
status	string	Any errors associated with the sample. For example, if the aggregation of data over multiple nodes fails then any of the partial errors might be returned, "ok" on success, or "error" on any 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	<a href="#">throughput</a>	The rate of throughput bytes per second observed at the storage object.
timestamp	string	The timestamp of the performance data.
uuid	string	The unique identifier of the NVMe namespace.

#### error\_arguments

Name	Type	Description
code	string	Argument code
message	string	Message argument

#### returned\_error

Name	Type	Description
arguments	array[ <a href="#">error_arguments</a> ]	Message arguments
code	string	Error code
message	string	Error message
target	string	The target parameter that caused the error.

## Retrieve historical performance metrics for an NVMe namespace for a specific time

GET /storage/namespaces/{nvme\_namespace.uuid}/metrics/{timestamp}

**Introduced In:** 9.14

Retrieves historical performance metrics for a NVMe namespace for a specific time.

### Related ONTAP commands

- `statistics namespace show`

### Parameters

Name	Type	In	Required	Description
nvme_namespace.uuid	string	path	True	The unique identifier of the NVMe namespace.
timestamp	string	path	True	The timestamp of the performance data. <ul style="list-style-type: none"> <li>• format: date-time</li> </ul>
fields	array[string]	query	False	Specify the fields to return.

### Response

Status: 200, Ok

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
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	<a href="#">iops</a>	The rate of I/O operations observed at the storage object.
latency	<a href="#">latency</a>	The round trip latency in microseconds observed at the storage object.
status	string	Any errors associated with the sample. For example, if the aggregation of data over multiple nodes fails then any of the partial errors might be returned, "ok" on success, or "error" on any 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	<a href="#">throughput</a>	The rate of throughput bytes per second observed at the storage object.
timestamp	string	The timestamp of the performance data.
uuid	string	The unique identifier of the NVMe namespace.



## Example response

```
{
  "_links": {
    "self": {
      "href": "/api/resourceLink"
    }
  },
  "duration": "PT15S",
  "iops": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "latency": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "status": "ok",
  "throughput": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "timestamp": "2017-01-25 06:20:13 -0500",
  "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
}
```

## Error

Status: Default

### ONTAP Error Response Codes

Error Code	Description
8585947	No metrics are available for the requested object.
8586225	An unexpected error occurred retrieving metrics for the requested object.

Also see the table of common errors in the [Response body](#) overview section of this documentation.

Name	Type	Description
error	<a href="#">returned_error</a>	

### 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	<a href="#">href</a>	

iops

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

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

latency

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

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

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

#### throughput

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

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

#### error\_arguments

Name	Type	Description
code	string	Argument code
message	string	Message argument

#### returned\_error

Name	Type	Description
arguments	array[ <a href="#">error_arguments</a> ]	Message arguments
code	string	Error code
message	string	Error message
target	string	The target parameter that caused the error.

# Delete an NVMe namespace

DELETE /storage/namespaces/{uuid}

Introduced In: 9.6

Deletes an NVMe namespace.

## Related ONTAP commands

- `vserver nvme namespace delete`

## Learn more

- [DOC /storage/namespaces](#)

## Parameters

Name	Type	In	Required	Description
uuid	string	path	True	The unique identifier of the NVMe namespace to delete.
allow_delete_while_mapped	boolean	query	False	Allows deletion of a mapped NVMe namespace. A mapped NVMe namespace might be in use. Deleting a mapped namespace also deletes the namespace map and makes the data no longer available, possibly causing a disruption in the availability of data. <b>This parameter should be used with caution.</b> <ul style="list-style-type: none"><li>• Default value:</li></ul>

## Response

Status: 200, Ok

## Error

Status: Default

### ONTAP Error Response Codes

Error Code	Description
4	The specified namespace was not found.
72089796	The namespace must be unmapped before deletion.
72090016	The namespace's aggregate is offline. The aggregate must be online to modify or remove the namespace.
72090017	The namespace's volume is offline. The volume must be online to modify or remove the namespace.

Also see the table of common errors in the [Response body](#) overview section of this documentation.

Name	Type	Description
error	<a href="#">returned_error</a>	

### Example error

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

## Definitions

## See Definitions

error\_arguments

Name	Type	Description
code	string	Argument code
message	string	Message argument

returned\_error

Name	Type	Description
arguments	array[ <a href="#">error_arguments</a> ]	Message arguments
code	string	Error code
message	string	Error message
target	string	The target parameter that caused the error.

## Retrieve an NVMe namespace

GET /storage/namespaces/{uuid}

**Introduced In:** 9.6

Retrieves an NVMe namespace.

### Expensive properties

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

- `auto_delete`
- `subsystem_map.*`
- `status.mapped`
- `statistics.*`
- `metric.*`

### Related ONTAP commands

- `vserver nvme namespace show`

- `vserver nvme subsystem map show`

## Learn more

- [DOC /storage/namespaces](#)

## Parameters

Name	Type	In	Required	Description
uuid	string	path	True	The unique identifier of the NVMe namespace to retrieve.
fields	array[string]	query	False	Specify the fields to return.

## Response

Status: 200, Ok

Name	Type	Description
<code>_links</code>	<code><a href="#">_links</a></code>	



Name	Type	Description
auto_delete	boolean	<p>This property marks the NVMe namespace for auto deletion when the volume containing the namespace runs out of space. This is most commonly set on namespace clones.</p> <p>When set to <i>true</i>, the NVMe namespace becomes eligible for automatic deletion when the volume runs out of space. Auto deletion only occurs when the volume containing the namespace is also configured for auto deletion and free space in the volume decreases below a particular threshold.</p> <p>This property is optional in POST and PATCH. The default value for a new NVMe namespace is <i>false</i>.</p> <p>There is an added computational 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 <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.</p>
clone	<a href="#">clone</a>	<p>This sub-object is used in POST to create a new NVMe namespace as a clone of an existing namespace, or PATCH to overwrite an existing namespace as a clone of another. Setting a property in this sub-object indicates that a namespace clone is desired.</p> <p>When used in a PATCH, the patched NVMe namespace's data is over-written as a clone of the source and the following properties are preserved from the patched namespace unless otherwise specified as part of the PATCH: <code>auto_delete</code> (unless specified in the request), <code>subsystem_map</code>, <code>status.state</code>, and <code>uuid</code>.</p>

Name	Type	Description
comment	string	A configurable comment available for use by the administrator. Valid in POST and PATCH.
convert	<a href="#">convert</a>	This sub-object is used in POST to convert a valid in-place LUN to an NVMe namespace. Setting a property in this sub-object indicates that a conversion from the specified LUN to NVMe namespace is desired.
create_time	string	The time the NVMe namespace was created.
enabled	boolean	The enabled state of the NVMe namespace. Certain error conditions cause the namespace to become disabled. If the namespace is disabled, you can check the <code>state</code> property to determine what error disabled the namespace. An NVMe namespace is enabled automatically when it is created.
location	<a href="#">location</a>	<p>The location of the NVMe namespace within the ONTAP cluster. Valid in POST.</p> <p>NVMe namespaces do not support rename, or movement between volumes.</p> <ul style="list-style-type: none"> <li>• Introduced in: 9.6</li> <li>• readCreate: 1</li> </ul>
metric	<a href="#">metric</a>	Performance numbers, such as IOPS latency and throughput
name	string	<p>The fully qualified path name of the NVMe namespace composed of a <code>"/vol"</code> prefix, the volume name, the (optional) <code>qtree</code> name and base name of the namespace. Valid in POST.</p> <p>NVMe namespaces do not support rename, or movement between volumes.</p>

Name	Type	Description
os_type	string	The operating system type of the NVMe namespace.  Required in POST when creating an NVMe namespace that is not a clone of another. Disallowed in POST when creating a namespace clone.
space	<a href="#">space</a>	The storage space related properties of the NVMe namespace.
statistics	<a href="#">statistics</a>	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	<a href="#">status</a>	Status information about the NVMe namespace.
subsystem_map	<a href="#">subsystem_map</a>	The NVMe subsystem with which the NVMe namespace is associated. A namespace can be mapped to zero (0) or one (1) subsystems.  There is an added computational cost to retrieving property values for <code>subsystem_map</code> . They are not populated for either a collection GET or an instance GET unless explicitly requested using the <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.
svm	<a href="#">svm</a>	SVM, applies only to SVM-scoped objects.
uuid	string	The unique identifier of the NVMe namespace.

## Example response

```
{
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "clone": {
    "source": {
      "name": "/vol/volume1/namespace1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    }
  },
  "comment": "string",
  "convert": {
    "lun": {
      "name": "/vol/volume1/lun1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    }
  },
  "create_time": "2018-06-04 15:00:00 -0400",
  "location": {
    "namespace": "namespace1",
    "node": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "name": "node1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    },
    "qtree": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "id": 1,
      "name": "qt1"
    },
    "volume": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      }
    }
  }
}
```

```

    }
  },
  "name": "volume1",
  "uuid": "028baa66-41bd-11e9-81d5-00a0986138f7"
}
},
"metric": {
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
},
"duration": "PT15S",
"iops": {
  "read": 200,
  "total": 1000,
  "write": 100
},
"latency": {
  "read": 200,
  "total": 1000,
  "write": 100
},
"status": "ok",
"throughput": {
  "read": 200,
  "total": 1000,
  "write": 100
},
"timestamp": "2017-01-25 06:20:13 -0500"
},
"name": "/vol/volume1/qtree1/namespace1",
"os_type": "aix",
"space": {
  "block_size": 512,
  "size": 1073741824,
  "used": 0
},
"statistics": {
  "iops_raw": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "latency_raw": {
    "read": 200,

```

```
    "total": 1000,
    "write": 100
  },
  "status": "ok",
  "throughput_raw": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "timestamp": "2017-01-25 06:20:13 -0500"
},
"status": {
  "container_state": "online",
  "state": "online"
},
"subsystem_map": {
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "anagrpid": "00103050h",
  "nsid": "00000001h",
  "subsystem": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "name": "subsystem1",
    "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
  }
},
"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
72090006	The specified namespace was not found.
72090007	The specified namespace was not found.

Also see the table of common errors in the [Response body](#) overview section of this documentation.

Name	Type	Description
error	<a href="#">returned_error</a>	

### 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	<a href="#">href</a>	

source

The source NVMe namespace for a namespace 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 namespace.

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

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

Name	Type	Description
name	string	The fully qualified path name of the clone source NVMe namespace composed of a "/vol" prefix, the volume name, the (optional) qtree name and base name of the namespace. Valid in POST and PATCH.
uuid	string	The unique identifier of the clone source NVMe namespace. Valid in POST and PATCH.

clone

This sub-object is used in POST to create a new NVMe namespace as a clone of an existing namespace, or PATCH to overwrite an existing namespace as a clone of another. Setting a property in this sub-object indicates that a namespace clone is desired.

When used in a PATCH, the patched NVMe namespace's data is over-written as a clone of the source and the following properties are preserved from the patched namespace unless otherwise specified as part of the PATCH: `auto_delete` (unless specified in the request), `subsystem_map`, `status.state`, and `uuid`.



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

#### lun

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

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

Name	Type	Description
name	string	The fully qualified path name of the source LUN composed of a "/vol" prefix, the volume name, the (optional) qtree name and base name of the LUN. Valid in POST.
uuid	string	The unique identifier of the source LUN. Valid in POST.

#### convert

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

Name	Type	Description
lun	<a href="#">lun</a>	The source LUN for convert operation. This can be specified using property <code>convert.lun.uuid</code> or <code>convert.lun.name</code> . If both properties are supplied, they must refer to the same LUN.  Valid in POST. A convert request from LUN to NVMe namespace cannot be combined with setting any other namespace properties. All other properties of the converted NVMe namespace comes from the source LUN.

#### node

The cluster node that hosts the NVMe namespace.

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
name	string	
uuid	string	

#### qtree

The qtree in which the NVMe namespace is optionally located. Valid in POST.

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.

NVMe namespaces do not support rename.

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
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 NVMe namespace is located. Valid in POST.

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.

NVMe namespaces do not support movement between volumes.

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
name	string	The name of the volume. This field cannot be specified in a POST or PATCH method.
uuid	string	Unique identifier for the volume. This corresponds to the instance-uuid that is exposed in the CLI and ONTAPI. It does not change due to a volume move. <ul style="list-style-type: none"><li>• example: 028baa66-41bd-11e9-81d5-00a0986138f7</li><li>• Introduced in: 9.6</li><li>• x-nullable: true</li></ul>

location

The location of the NVMe namespace within the ONTAP cluster. Valid in POST.

NVMe namespaces do not support rename, or movement between volumes.

Name	Type	Description
namespace	string	The base name component of the NVMe namespace. Valid in POST.  If properties <code>name</code> and <code>location.namespace</code> are specified in the same request, they must refer to the base name.  NVMe namespaces do not support rename.
node	<a href="#">node</a>	The cluster node that hosts the NVMe namespace.

Name	Type	Description
qtree	qtree	<p>The qtree in which the NVMe namespace is optionally located. Valid in POST.</p> <p>If properties <code>name</code> and <code>location.qtree.name</code> and/or <code>location.qtree.uuid</code> are specified in the same request, they must refer to the same qtree.</p> <p>NVMe namespaces do not support rename.</p>
volume	volume	<p>The volume in which the NVMe namespace is located. Valid in POST.</p> <p>If properties <code>name</code> and <code>location.volume.name</code> and/or <code>location.volume.uuid</code> are specified in the same request, they must refer to the same volume.</p> <p>NVMe namespaces do not support movement between volumes.</p>

## iops

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

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

## latency

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

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

## throughput

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

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

## metric

Performance numbers, such as IOPS latency and throughput

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
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:

Name	Type	Description
iops	<a href="#">iops</a>	The rate of I/O operations observed at the storage object.
latency	<a href="#">latency</a>	The round trip latency in microseconds observed at the storage object.
status	string	Any errors associated with the sample. For example, if the aggregation of data over multiple nodes fails then any of the partial errors might be returned, "ok" on success, or "error" on any 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	<a href="#">throughput</a>	The rate of throughput bytes per second observed at the storage object.
timestamp	string	The timestamp of the performance data.

#### guarantee

Properties that request and report the space guarantee for the NVMe namespace.

Name	Type	Description
requested	boolean	<p>The requested space reservation policy for the NVMe namespace. If <i>true</i>, a space reservation is requested for the namespace; if <i>false</i>, the namespace is thin provisioned. Guaranteeing a space reservation request for a namespace requires that the volume in which the namespace resides also be space reserved and that the fractional reserve for the volume be 100%.</p> <p>The space reservation policy for an NVMe namespace is determined by ONTAP.</p> <ul style="list-style-type: none"> <li>• readOnly: 1</li> <li>• Introduced in: 9.6</li> <li>• x-nullable: true</li> </ul>
reserved	boolean	<p>Reports if the NVMe namespace is space guaranteed.</p> <p>This property is <i>true</i> if a space guarantee is requested and the containing volume and aggregate support the request. This property is <i>false</i> if a space guarantee is not requested or if a space guarantee is requested and either the containing volume and aggregate do not support the request.</p>

## space

The storage space related properties of the NVMe namespace.

Name	Type	Description
block_size	integer	<p>The size of blocks in the namespace in bytes.</p> <p>Valid in POST when creating an NVMe namespace that is not a clone of another. Disallowed in POST when creating a namespace clone. Valid in POST.</p>

Name	Type	Description
guarantee	guarantee	Properties that request and report the space guarantee for the NVMe namespace.
size	integer	<p>The total provisioned size of the NVMe namespace. Valid in POST and PATCH. The NVMe namespace size can be increased but not be made smaller using the REST interface.</p> <p>The maximum and minimum sizes listed here are the absolute maximum and absolute minimum sizes in bytes. The maximum size is variable with respect to large NVMe namespace support in ONTAP. If large namespaces are supported, the maximum size is 128 TB (140737488355328 bytes) and if not supported, the maximum size is just under 16 TB (17557557870592 bytes). The minimum size supported is always 4096 bytes.</p> <p>For more information, see <i>Size properties</i> in the <i>docs</i> section of the ONTAP REST API documentation.</p> <ul style="list-style-type: none"> <li>• example: 1073741824</li> <li>• format: int64</li> <li>• Max value: 140737488355328</li> <li>• Min value: 4096</li> <li>• Introduced in: 9.6</li> <li>• x-nullable: true</li> </ul>



Name	Type	Description
used	integer	<p>The amount of space consumed by the main data stream of the NVMe namespace.</p> <p>This value is the total space consumed in the volume by the NVMe namespace, including filesystem overhead, but excluding prefix and suffix streams. Due to internal filesystem overhead and the many ways NVMe filesystems and applications utilize blocks within a namespace, this value does not necessarily reflect actual consumption/availability from the perspective of the filesystem or application. Without specific knowledge of how the namespace blocks are utilized outside of ONTAP, this property should not be used and an indicator for an out-of-space condition.</p> <p>For more information, see <i>Size properties</i> in the <i>docs</i> section of the ONTAP REST API documentation.</p> <ul style="list-style-type: none"> <li>• format: int64</li> <li>• readOnly: 1</li> <li>• Introduced in: 9.6</li> <li>• x-nullable: true</li> </ul>

#### iops\_raw

The number of I/O operations observed at the storage object. This should 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.

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

#### latency\_raw

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

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

#### throughput\_raw

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

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

## 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	<a href="#">iops_raw</a>	The number of I/O operations observed at the storage object. This should be used along with delta time to calculate the rate of I/O operations per unit of time.
latency_raw	<a href="#">latency_raw</a>	The raw latency in microseconds observed at the storage object. This should be divided by the raw IOPS value to calculate the average latency per I/O operation.
status	string	Any errors associated with the sample. For example, if the aggregation of data over multiple nodes fails then any of the partial errors might be returned, "ok" on success, or "error" on any 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	<a href="#">throughput_raw</a>	Throughput bytes observed at the storage object. This should be used along with delta time to calculate the rate of throughput bytes per unit of time.

Name	Type	Description
timestamp	string	The timestamp of the performance data.

status

Status information about the NVMe namespace.

Name	Type	Description
container_state	string	The state of the volume and aggregate that contain the NVMe namespace. Namespaces are only available when their containers are available.
mapped	boolean	Reports if the NVMe namespace is mapped to an NVMe subsystem.  There is an added computational 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 <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.
read_only	boolean	Reports if the NVMe namespace allows only read access.
state	string	The state of the NVMe namespace. Normal states for a namespace are <i>online</i> and <i>offline</i> . Other states indicate errors.

subsystem

The NVMe subsystem to which the NVMe namespace is mapped.

Name	Type	Description
<code>_links</code>	<a href="#">_links</a>	
name	string	The name of the NVMe subsystem.

Name	Type	Description
uuid	string	The unique identifier of the NVMe subsystem.

#### subsystem\_map

The NVMe subsystem with which the NVMe namespace is associated. A namespace can be mapped to zero (0) or one (1) subsystems.

There is an added computational cost to retrieving property values for `subsystem_map`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See [Requesting specific fields](#) to learn more.

Name	Type	Description
<code>_links</code>	<a href="#">_links</a>	
anagrpId	string	The Asymmetric Namespace Access Group ID (ANAGRPID) of the NVMe namespace.  The format for an ANAGRPID is 8 hexadecimal digits (zero-filled) followed by a lower case "h".
nsid	string	The NVMe namespace identifier. This is an identifier used by an NVMe controller to provide access to the NVMe namespace.  The format for an NVMe namespace identifier is 8 hexadecimal digits (zero-filled) followed by a lower case "h".
subsystem	<a href="#">subsystem</a>	The NVMe subsystem to which the NVMe namespace is mapped.

#### svm

SVM, applies only to SVM-scoped objects.

Name	Type	Description
<code>_links</code>	<a href="#">_links</a>	
name	string	The name of the SVM. This field cannot be specified in a PATCH method.

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

error\_arguments

Name	Type	Description
code	string	Argument code
message	string	Message argument

returned\_error

Name	Type	Description
arguments	array[ <a href="#">error_arguments</a> ]	Message arguments
code	string	Error code
message	string	Error message
target	string	The target parameter that caused the error.

## Update an NVMe namespace

PATCH /storage/namespaces/{uuid}

**Introduced In:** 9.6

Updates an NVMe namespace.

### Related ONTAP commands

- `volume file clone autodelete`
- `vserver nvme namespace modify`

### Learn more

- [DOC /storage/namespaces](#)

### Parameters

Name	Type	In	Required	Description
uuid	string	path	True	The unique identifier of the NVMe namespace to update.

## Request Body

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
auto_delete	boolean	<p>This property marks the NVMe namespace for auto deletion when the volume containing the namespace runs out of space. This is most commonly set on namespace clones.</p> <p>When set to <i>true</i>, the NVMe namespace becomes eligible for automatic deletion when the volume runs out of space. Auto deletion only occurs when the volume containing the namespace is also configured for auto deletion and free space in the volume decreases below a particular threshold.</p> <p>This property is optional in POST and PATCH. The default value for a new NVMe namespace is <i>false</i>.</p> <p>There is an added computational 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 <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.</p>

Name	Type	Description
clone	clone	<p>This sub-object is used in POST to create a new NVMe namespace as a clone of an existing namespace, or PATCH to overwrite an existing namespace as a clone of another. Setting a property in this sub-object indicates that a namespace clone is desired.</p> <p>When used in a PATCH, the patched NVMe namespace's data is over-written as a clone of the source and the following properties are preserved from the patched namespace unless otherwise specified as part of the PATCH: <code>auto_delete</code> (unless specified in the request), <code>subsystem_map</code>, <code>status.state</code>, and <code>uuid</code>.</p>
comment	string	A configurable comment available for use by the administrator. Valid in POST and PATCH.
convert	convert	This sub-object is used in POST to convert a valid in-place LUN to an NVMe namespace. Setting a property in this sub-object indicates that a conversion from the specified LUN to NVMe namespace is desired.
create_time	string	The time the NVMe namespace was created.
enabled	boolean	The enabled state of the NVMe namespace. Certain error conditions cause the namespace to become disabled. If the namespace is disabled, you can check the <code>state</code> property to determine what error disabled the namespace. An NVMe namespace is enabled automatically when it is created.



Name	Type	Description
location	<a href="#">location</a>	<p>The location of the NVMe namespace within the ONTAP cluster. Valid in POST.</p> <p>NVMe namespaces do not support rename, or movement between volumes.</p> <ul style="list-style-type: none"> <li>• Introduced in: 9.6</li> <li>• readCreate: 1</li> </ul>
metric	<a href="#">metric</a>	Performance numbers, such as IOPS latency and throughput
name	string	<p>The fully qualified path name of the NVMe namespace composed of a "/vol" prefix, the volume name, the (optional) qtree name and base name of the namespace. Valid in POST.</p> <p>NVMe namespaces do not support rename, or movement between volumes.</p>
os_type	string	<p>The operating system type of the NVMe namespace.</p> <p>Required in POST when creating an NVMe namespace that is not a clone of another. Disallowed in POST when creating a namespace clone.</p>
space	<a href="#">space</a>	The storage space related properties of the NVMe namespace.
statistics	<a href="#">statistics</a>	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	<a href="#">status</a>	Status information about the NVMe namespace.

Name	Type	Description
subsystem_map	<a href="#">subsystem_map</a>	<p>The NVMe subsystem with which the NVMe namespace is associated. A namespace can be mapped to zero (0) or one (1) subsystems.</p> <p>There is an added computational cost to retrieving property values for <code>subsystem_map</code>. They are not populated for either a collection GET or an instance GET unless explicitly requested using the <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.</p>
svm	<a href="#">svm</a>	SVM, applies only to SVM-scoped objects.
uuid	string	The unique identifier of the NVMe namespace.

## Example request

```
{
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "clone": {
    "source": {
      "name": "/vol/volume1/namespace1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    }
  },
  "comment": "string",
  "convert": {
    "lun": {
      "name": "/vol/volume1/lun1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    }
  },
  "create_time": "2018-06-04 15:00:00 -0400",
  "location": {
    "namespace": "namespace1",
    "node": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "name": "node1",
      "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
    },
    "qtree": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "id": 1,
      "name": "qt1"
    },
    "volume": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      }
    }
  }
}
```

```

    }
  },
  "name": "volume1",
  "uuid": "028baa66-41bd-11e9-81d5-00a0986138f7"
}
},
"metric": {
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
},
"duration": "PT15S",
"iops": {
  "read": 200,
  "total": 1000,
  "write": 100
},
"latency": {
  "read": 200,
  "total": 1000,
  "write": 100
},
"status": "ok",
"throughput": {
  "read": 200,
  "total": 1000,
  "write": 100
},
"timestamp": "2017-01-25 06:20:13 -0500"
},
"name": "/vol/volume1/qtree1/namespace1",
"os_type": "aix",
"space": {
  "block_size": 512,
  "size": 1073741824,
  "used": 0
},
"statistics": {
  "iops_raw": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "latency_raw": {
    "read": 200,

```

```

    "total": 1000,
    "write": 100
  },
  "status": "ok",
  "throughput_raw": {
    "read": 200,
    "total": 1000,
    "write": 100
  },
  "timestamp": "2017-01-25 06:20:13 -0500"
},
"status": {
  "container_state": "online",
  "state": "online"
},
"subsystem_map": {
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "anagrpid": "00103050h",
  "nsid": "00000001h",
  "subsystem": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "name": "subsystem1",
    "uuid": "1cd8a442-86d1-11e0-ae1c-123478563412"
  }
},
"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
5374127	An invalid namespace name was specified.
13565952	The namespace clone request failed.
72089724	The specified namespace size is too large.
72089730	The specified namespace cannot be updated as it resides in a Snapshot copy.
72090005	The specified <code>clone.source.uuid</code> and <code>clone.source.name</code> do not refer to the same LUN.
72090006	The specified namespace was not found. This can apply to <code>clone.source</code> or the target namespace. The <code>target</code> property of the error object identifies the property.
72090007	The specified namespace was not found. This can apply to <code>clone.source</code> or the target namespace. The <code>target</code> property of the error object identifies the property.
72090010	An error occurred after successfully overwriting data for the namespace as a clone. Some properties were not modified.
72090011	An error occurred after successfully modifying some of the properties of the namespace. Some properties were not modified.
72090016	The namespace's aggregate is offline. The aggregate must be online to modify or remove the namespace.
72090017	The namespace's volume is offline. The volume must be online to modify or remove the namespace.
72090038	An attempt was made to reduce the size of the specified namespace.

Also see the table of common errors in the [Response body](#) overview section of this documentation.

Name	Type	Description
error	<a href="#">returned_error</a>	

### 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	<a href="#">href</a>	

source

The source NVMe namespace for a namespace 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 namespace.

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

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

Name	Type	Description
name	string	The fully qualified path name of the clone source NVMe namespace composed of a "/vol" prefix, the volume name, the (optional) qtree name and base name of the namespace. Valid in POST and PATCH.
uuid	string	The unique identifier of the clone source NVMe namespace. Valid in POST and PATCH.

clone

This sub-object is used in POST to create a new NVMe namespace as a clone of an existing namespace, or PATCH to overwrite an existing namespace as a clone of another. Setting a property in this sub-object indicates that a namespace clone is desired.

When used in a PATCH, the patched NVMe namespace's data is over-written as a clone of the source and the following properties are preserved from the patched namespace unless otherwise specified as part of the PATCH: `auto_delete` (unless specified in the request), `subsystem_map`, `status.state`, and `uuid`.



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

#### lun

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

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

Name	Type	Description
name	string	The fully qualified path name of the source LUN composed of a "/vol" prefix, the volume name, the (optional) qtree name and base name of the LUN. Valid in POST.
uuid	string	The unique identifier of the source LUN. Valid in POST.

#### convert

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

Name	Type	Description
lun	<a href="#">lun</a>	The source LUN for convert operation. This can be specified using property <code>convert.lun.uuid</code> or <code>convert.lun.name</code> . If both properties are supplied, they must refer to the same LUN.  Valid in POST. A convert request from LUN to NVMe namespace cannot be combined with setting any other namespace properties. All other properties of the converted NVMe namespace comes from the source LUN.

#### node

The cluster node that hosts the NVMe namespace.

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
name	string	
uuid	string	

#### qtree

The qtree in which the NVMe namespace is optionally located. Valid in POST.

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.

NVMe namespaces do not support rename.

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
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 NVMe namespace is located. Valid in POST.

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.

NVMe namespaces do not support movement between volumes.

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
name	string	The name of the volume. This field cannot be specified in a POST or PATCH method.
uuid	string	Unique identifier for the volume. This corresponds to the instance-uuid that is exposed in the CLI and ONTAPI. It does not change due to a volume move. <ul style="list-style-type: none"><li>• example: 028baa66-41bd-11e9-81d5-00a0986138f7</li><li>• Introduced in: 9.6</li><li>• x-nullable: true</li></ul>

location

The location of the NVMe namespace within the ONTAP cluster. Valid in POST.

NVMe namespaces do not support rename, or movement between volumes.

Name	Type	Description
namespace	string	The base name component of the NVMe namespace. Valid in POST.  If properties <code>name</code> and <code>location.namespace</code> are specified in the same request, they must refer to the base name.  NVMe namespaces do not support rename.
node	<a href="#">node</a>	The cluster node that hosts the NVMe namespace.

Name	Type	Description
qtree	qtree	<p>The qtree in which the NVMe namespace is optionally located. Valid in POST.</p> <p>If properties <code>name</code> and <code>location.qtree.name</code> and/or <code>location.qtree.uuid</code> are specified in the same request, they must refer to the same qtree.</p> <p>NVMe namespaces do not support rename.</p>
volume	volume	<p>The volume in which the NVMe namespace is located. Valid in POST.</p> <p>If properties <code>name</code> and <code>location.volume.name</code> and/or <code>location.volume.uuid</code> are specified in the same request, they must refer to the same volume.</p> <p>NVMe namespaces do not support movement between volumes.</p>

## iops

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

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

## latency

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

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

## throughput

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

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

## metric

Performance numbers, such as IOPS latency and throughput

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	
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:

Name	Type	Description
iops	<a href="#">iops</a>	The rate of I/O operations observed at the storage object.
latency	<a href="#">latency</a>	The round trip latency in microseconds observed at the storage object.
status	string	Any errors associated with the sample. For example, if the aggregation of data over multiple nodes fails then any of the partial errors might be returned, "ok" on success, or "error" on any 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	<a href="#">throughput</a>	The rate of throughput bytes per second observed at the storage object.
timestamp	string	The timestamp of the performance data.

#### guarantee

Properties that request and report the space guarantee for the NVMe namespace.

Name	Type	Description
requested	boolean	<p>The requested space reservation policy for the NVMe namespace. If <i>true</i>, a space reservation is requested for the namespace; if <i>false</i>, the namespace is thin provisioned. Guaranteeing a space reservation request for a namespace requires that the volume in which the namespace resides also be space reserved and that the fractional reserve for the volume be 100%.</p> <p>The space reservation policy for an NVMe namespace is determined by ONTAP.</p> <ul style="list-style-type: none"> <li>• readOnly: 1</li> <li>• Introduced in: 9.6</li> <li>• x-nullable: true</li> </ul>
reserved	boolean	<p>Reports if the NVMe namespace is space guaranteed.</p> <p>This property is <i>true</i> if a space guarantee is requested and the containing volume and aggregate support the request. This property is <i>false</i> if a space guarantee is not requested or if a space guarantee is requested and either the containing volume and aggregate do not support the request.</p>

space

The storage space related properties of the NVMe namespace.

Name	Type	Description
block_size	integer	<p>The size of blocks in the namespace in bytes.</p> <p>Valid in POST when creating an NVMe namespace that is not a clone of another. Disallowed in POST when creating a namespace clone. Valid in POST.</p>

Name	Type	Description
guarantee	<a href="#">guarantee</a>	Properties that request and report the space guarantee for the NVMe namespace.
size	integer	<p>The total provisioned size of the NVMe namespace. Valid in POST and PATCH. The NVMe namespace size can be increased but not be made smaller using the REST interface.</p> <p>The maximum and minimum sizes listed here are the absolute maximum and absolute minimum sizes in bytes. The maximum size is variable with respect to large NVMe namespace support in ONTAP. If large namespaces are supported, the maximum size is 128 TB (140737488355328 bytes) and if not supported, the maximum size is just under 16 TB (17557557870592 bytes). The minimum size supported is always 4096 bytes.</p> <p>For more information, see <i>Size properties</i> in the <i>docs</i> section of the ONTAP REST API documentation.</p> <ul style="list-style-type: none"> <li>• example: 1073741824</li> <li>• format: int64</li> <li>• Max value: 140737488355328</li> <li>• Min value: 4096</li> <li>• Introduced in: 9.6</li> <li>• x-nullable: true</li> </ul>



Name	Type	Description
used	integer	<p>The amount of space consumed by the main data stream of the NVMe namespace.</p> <p>This value is the total space consumed in the volume by the NVMe namespace, including filesystem overhead, but excluding prefix and suffix streams. Due to internal filesystem overhead and the many ways NVMe filesystems and applications utilize blocks within a namespace, this value does not necessarily reflect actual consumption/availability from the perspective of the filesystem or application. Without specific knowledge of how the namespace blocks are utilized outside of ONTAP, this property should not be used and an indicator for an out-of-space condition.</p> <p>For more information, see <i>Size properties</i> in the <i>docs</i> section of the ONTAP REST API documentation.</p> <ul style="list-style-type: none"> <li>• format: int64</li> <li>• readOnly: 1</li> <li>• Introduced in: 9.6</li> <li>• x-nullable: true</li> </ul>

#### iops\_raw

The number of I/O operations observed at the storage object. This should 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.

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

#### latency\_raw

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

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

#### throughput\_raw

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

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

## 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	<a href="#">iops_raw</a>	The number of I/O operations observed at the storage object. This should be used along with delta time to calculate the rate of I/O operations per unit of time.
latency_raw	<a href="#">latency_raw</a>	The raw latency in microseconds observed at the storage object. This should be divided by the raw IOPS value to calculate the average latency per I/O operation.
status	string	Any errors associated with the sample. For example, if the aggregation of data over multiple nodes fails then any of the partial errors might be returned, "ok" on success, or "error" on any 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	<a href="#">throughput_raw</a>	Throughput bytes observed at the storage object. This should be used along with delta time to calculate the rate of throughput bytes per unit of time.

Name	Type	Description
timestamp	string	The timestamp of the performance data.

status

Status information about the NVMe namespace.

Name	Type	Description
container_state	string	The state of the volume and aggregate that contain the NVMe namespace. Namespaces are only available when their containers are available.
mapped	boolean	Reports if the NVMe namespace is mapped to an NVMe subsystem.  There is an added computational 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 <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.
read_only	boolean	Reports if the NVMe namespace allows only read access.
state	string	The state of the NVMe namespace. Normal states for a namespace are <i>online</i> and <i>offline</i> . Other states indicate errors.

subsystem

The NVMe subsystem to which the NVMe namespace is mapped.

Name	Type	Description
<code>_links</code>	<a href="#">_links</a>	
name	string	The name of the NVMe subsystem.

Name	Type	Description
uuid	string	The unique identifier of the NVMe subsystem.

#### subsystem\_map

The NVMe subsystem with which the NVMe namespace is associated. A namespace can be mapped to zero (0) or one (1) subsystems.

There is an added computational cost to retrieving property values for `subsystem_map`. They are not populated for either a collection GET or an instance GET unless explicitly requested using the `fields` query parameter. See [Requesting specific fields](#) to learn more.

Name	Type	Description
<code>_links</code>	<a href="#">_links</a>	
anagrpId	string	The Asymmetric Namespace Access Group ID (ANAGRPID) of the NVMe namespace.  The format for an ANAGRPID is 8 hexadecimal digits (zero-filled) followed by a lower case "h".
nsid	string	The NVMe namespace identifier. This is an identifier used by an NVMe controller to provide access to the NVMe namespace.  The format for an NVMe namespace identifier is 8 hexadecimal digits (zero-filled) followed by a lower case "h".
subsystem	<a href="#">subsystem</a>	The NVMe subsystem to which the NVMe namespace is mapped.

#### svm

SVM, applies only to SVM-scoped objects.

Name	Type	Description
<code>_links</code>	<a href="#">_links</a>	
name	string	The name of the SVM. This field cannot be specified in a PATCH method.

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

### nvme\_namespace

An NVMe namespace is a collection of addressable logical blocks presented to hosts connected to the storage virtual machine using the NVMe over Fabrics protocol.

In ONTAP, an NVMe namespace is located within a volume. Optionally, it can be located within a qtree in a volume.

An NVMe namespace is created to a specified size using thin or thick provisioning as determined by the volume on which it is created. NVMe namespaces support being cloned. An NVMe namespace cannot be renamed, resized, or moved to a different volume. NVMe namespaces do not support the assignment of a QoS policy for performance management, but a QoS policy can be assigned to the volume containing the namespace. See the NVMe namespace object model to learn more about each of the properties supported by the NVMe namespace REST API.

An NVMe namespace must be mapped to an NVMe subsystem to grant access to the subsystem's hosts. Hosts can then access the NVMe namespace and perform I/O using the NVMe over Fabrics protocol.

Name	Type	Description
<a href="#">_links</a>	<a href="#">_links</a>	

Name	Type	Description
auto_delete	boolean	<p>This property marks the NVMe namespace for auto deletion when the volume containing the namespace runs out of space. This is most commonly set on namespace clones.</p> <p>When set to <i>true</i>, the NVMe namespace becomes eligible for automatic deletion when the volume runs out of space. Auto deletion only occurs when the volume containing the namespace is also configured for auto deletion and free space in the volume decreases below a particular threshold.</p> <p>This property is optional in POST and PATCH. The default value for a new NVMe namespace is <i>false</i>.</p> <p>There is an added computational 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 <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.</p>

Name	Type	Description
clone	clone	<p>This sub-object is used in POST to create a new NVMe namespace as a clone of an existing namespace, or PATCH to overwrite an existing namespace as a clone of another. Setting a property in this sub-object indicates that a namespace clone is desired.</p> <p>When used in a PATCH, the patched NVMe namespace's data is over-written as a clone of the source and the following properties are preserved from the patched namespace unless otherwise specified as part of the PATCH: <code>auto_delete</code> (unless specified in the request), <code>subsystem_map</code>, <code>status.state</code>, and <code>uuid</code>.</p>
comment	string	A configurable comment available for use by the administrator. Valid in POST and PATCH.
convert	convert	This sub-object is used in POST to convert a valid in-place LUN to an NVMe namespace. Setting a property in this sub-object indicates that a conversion from the specified LUN to NVMe namespace is desired.
create_time	string	The time the NVMe namespace was created.
enabled	boolean	The enabled state of the NVMe namespace. Certain error conditions cause the namespace to become disabled. If the namespace is disabled, you can check the <code>state</code> property to determine what error disabled the namespace. An NVMe namespace is enabled automatically when it is created.



Name	Type	Description
location	location	<p>The location of the NVMe namespace within the ONTAP cluster. Valid in POST.</p> <p>NVMe namespaces do not support rename, or movement between volumes.</p> <ul style="list-style-type: none"> <li>• Introduced in: 9.6</li> <li>• readCreate: 1</li> </ul>
metric	metric	Performance numbers, such as IOPS latency and throughput
name	string	<p>The fully qualified path name of the NVMe namespace composed of a "/vol" prefix, the volume name, the (optional) qtree name and base name of the namespace. Valid in POST.</p> <p>NVMe namespaces do not support rename, or movement between volumes.</p>
os_type	string	<p>The operating system type of the NVMe namespace.</p> <p>Required in POST when creating an NVMe namespace that is not a clone of another. Disallowed in POST when creating a namespace clone.</p>
space	space	The storage space related properties of the NVMe namespace.
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 NVMe namespace.

Name	Type	Description
subsystem_map	<a href="#">subsystem_map</a>	<p>The NVMe subsystem with which the NVMe namespace is associated. A namespace can be mapped to zero (0) or one (1) subsystems.</p> <p>There is an added computational cost to retrieving property values for <code>subsystem_map</code>. They are not populated for either a collection GET or an instance GET unless explicitly requested using the <code>fields</code> query parameter. See <a href="#">Requesting specific fields</a> to learn more.</p>
svm	<a href="#">svm</a>	SVM, applies only to SVM-scoped objects.
uuid	string	The unique identifier of the NVMe namespace.

#### error\_arguments

Name	Type	Description
code	string	Argument code
message	string	Message argument

#### returned\_error

Name	Type	Description
arguments	array[ <a href="#">error_arguments</a> ]	Message arguments
code	string	Error code
message	string	Error message
target	string	The target parameter that caused the error.

## Copyright information

Copyright © 2024 NetApp, Inc. All Rights Reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

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

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

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

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

LIMITED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data -Noncommercial Items at DFARS 252.227-7013 (FEB 2014) and FAR 52.227-19 (DEC 2007).

Data contained herein pertains to a commercial product and/or commercial service (as defined in FAR 2.101) and is proprietary to NetApp, Inc. All NetApp technical data and computer software provided under this Agreement is commercial in nature and developed solely at private expense. The U.S. Government has a non-exclusive, non-transferrable, nonsublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b) (FEB 2014).

## Trademark information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.