NAS

ONTAP Automation

NetApp
July 25, 2024

Table of Contents

NAS

File security permissions

Prepare to manage file security and audit policies

You can manage the permissions and audit policies for files available through the SVMs within an ONTAP cluster.

Overview

ONTAP uses System Access Control Lists (SACLs) and Discretionary Access Control Lists (DACLs) to assign permissions to file objects. Beginning with ONTAP 9.9.1, the REST API includes support for managing the SACL and DACL permissions. You can use the API to automate the administration of the file security permissions. In many cases you can use a single REST API call instead of multiple CLI commands or ONTAPI (ZAPI) calls.

For ONTAP releases prior to 9.9.1, you can automate the administration of the SACL and DACL permissions using the CLI passthrough feature. See Migration considerations and Using the private CLI passthrough with the ONTAP REST API for more information.

Several example workflows are available to illustrate how to manage the ONTAP file security services using the REST API. Before using the workflows and issuing any of the REST API calls, make sure to review Prepare to use the workflows.

If you use Python, also see the script file_security_permissions.py for examples of how to automate some of the file security activities.

ONTAP REST API versus ONTAP CLI commands

For many tasks, using the ONTAP REST API requires fewer calls than the equivalent ONTAP CLI commands or ONTAPI (ZAPI) calls. The table below includes a list of API calls and the equivalent the CLI commands needed for each task.

ONTAP REST API	ONTAP CLI
`GET /protocols/file-security/effective-permissions/`	`vserver security file-directory show-effective-permissions`
`POST /protocols/file-security/permissions/`	1. `vserver security file-directory ntfs create` 2. `vserver security file-directory ntfs dacl add` 3. `vserver security file-directory ntfs sacl add` 4. `vserver security file-directory policy create` 5. `vserver security file-directory policy task add` 6. `vserver security file-directory apply`

ONTAP REST API	ONTAP CLI
`PATCH /protocols/file-security/permissions/`	`vserver security file-directory ntfs modify`
`DELETE /protocols/file-security/permissions/`	1. `vserver security file-directory ntfs dacl remove` 2. `vserver security file-directory ntfs sacl remove`

Related information

- Python script illustrating file permissions
- Simplified management of file-security permissions with ONTAP REST APIs
- Using the private CLI passthrough with the ONTAP REST API

Get the effective permissions for a file

You can retrieve the current effective permissions for a specific file or folder.

HTTP method and endpoint

This REST API call uses the following method and endpoint.

HTTP method	Path
GET	/api/protocols/file-security/effective-permissions/{svm.uuid}/{path}

Processing type

Synchronous

Additional input parameters for curl examples

In addition to the parameters common with all REST API calls, the following parameters are also used in the curl example in this step.

Parameter	Type	Required	Description
$SVM_ID	Path	Yes	This is the UUID of the SVM containing the file.
$FILE_PATH	Path	Yes	This is the path to the file or folder.

Curl example

```
curl --request GET \
--location "https://$FQDN_IP/api/protocols/file-security/effective-
permissions/$SVM_ID/$FILE_PATH" \
--include \
--header "Accept: */*" \
--header "Authorization: Basic $BASIC_AUTH"
```

JSON output example


```
{
  "svm": {
    "uuid": "cf5f271a-1beb-11ea-8fad-005056bb645e",
    "name": "vs1"
  },
  "user": "administrator",
  "type": "windows",
  "path": "/",
  "share": {
    "path": "/"
  },
  "file_permission": [
    "read",
    "write",
    "append",
    "read_ea",
    "write_ea",
    "execute",
    "delete_child",
    "read_attributes",
    "write_attributes",
    "delete",
    "read_control",
    "write_dac",
    "write_owner",
    "synchronize",
    "system_security"
  ],
  "share_permission": [
    "read",
    "read_ea",
    "execute",
    "read_attributes",
    "read_control",
    "synchronize"
  ]
}
```


Get the auditing information for a file

You can retrieve the auditing information for a specific file or folder.

HTTP method and endpoint

This REST API call uses the following method and endpoint.

HTTP method	Path
GET	/api/protocols/file-security/permissions/{svm.uuid}/{path}

Processing type

Synchronous

Additional input parameters for curl examples

In addition to the parameters common with all REST API calls, the following parameters are also used in the curl example in this step.

Parameter	Type	Required	Description
$SVM_ID	Path	Yes	This is the UUID of the SVM containing the file.
$FILE_PATH	Path	Yes	This is the path to the file or folder.

Curl example

```
curl --request GET \
--location "https://$FQDN_IP/api/protocols/file-
security/permissions/$SVM_ID/$FILE_PATH" \
--include \
--header "Accept: */*" \
--header "Authorization: Basic $BASIC_AUTH"
```

JSON output example

```
{
  "svm": {
    "uuid": "9479099d-5b9f-11eb-9c4e-0050568e8682",
    "name": "vs1"
  },
  "path": "/parent",
  "owner": "BUILTIN\\Administrators",
  "group": "BUILTIN\\Administrators",
  "control_flags": "0x8014",
  "acls": [
    {
      "user": "BUILTIN\\Administrators",
      "access": "access_allow",
      "apply_to": {
        "files": true,
        "sub_folders": true,
        "this_folder": true
      },
      "advanced_rights": {
        "append_data": true,
```

```
        "delete": true,
        "delete_child": true,
        "execute_file": true,
        "full_control": true,
        "read_attr": true,
        "read_data": true,
        "read_ea": true,
        "read_perm": true,
        "write_attr": true,
        "write_data": true,
        "write_ea": true,
        "write_owner": true,
        "synchronize": true,
        "write_perm": true
      },
      "access_control": "file_directory"
    },
    {
      "user": "BUILTIN\\Users",
      "access": "access_allow",
      "apply_to": {
        "files": true,
        "sub_folders": true,
        "this_folder": true
      },
      "advanced_rights": {
        "append_data": true,
        "delete": true,
        "delete_child": true,
        "execute_file": true,
        "full_control": true,
        "read_attr": true,
        "read_data": true,
        "read_ea": true,
        "read_perm": true,
        "write_attr": true,
        "write_data": true,
        "write_ea": true,
        "write_owner": true,
        "synchronize": true,
        "write_perm": true
      },
      "access_control": "file_directory"
    }
  ],
  "inode": 64,
```

```
  "security_style": "mixed",
  "effective_style": "ntfs",
  "dos_attributes": "10",
  "text_dos_attr": "----D---",
  "user_id": "0",
  "group_id": "0",
  "mode_bits": 777,
  "text_mode_bits": "rwxrwxrwx"
}
```

Apply new permissions to a file

You can apply a new security descriptor to a specific file or folder.

Step 1: Apply the new permissions

HTTP method and endpoint

This REST API call uses the following method and endpoint.

HTTP method	Path
POST	/api/protocols/file-security/permissions/{svm.uuid}/{path}

Processing type

Asynchronous

Additional input parameters for curl examples

In addition to the parameters common with all REST API calls, the following parameters are also used in the curl example in this step.

Parameter	Type	Required	Description
$SVM_ID	Path	Yes	This is the UUID of the SVM containing the file.
$FILE_PATH	Path	Yes	This is the path to the file or folder.

Curl example

```
curl --request POST --location "https://$FQDN_IP/api/protocols/file-
security/permissions/$SVM_ID/$FILE_PATH?return_timeout=0" --include
--header "Accept */*" --header "Authorization: Basic $BASIC_AUTH" --data
'{ \"acls\": [ { \"access\": \"access_allow\", \"advanced_rights\": {
\"append_data\": true, \"delete\": true, \"delete_child\": true,
\"execute_file\": true, \"full_control\": true, \"read_attr\": true,
\"read_data\": true, \"read_ea\": true, \"read_perm\": true,
\"write_attr\": true, \"write_data\": true, \"write_ea\": true,
\"write_owner\": true, \"write_perm\": true }, \"apply_to\": { \"files\":
true, \"sub_folders\": true, \"this_folder\": true }, \"user\":
\"administrator\" } ], \"control_flags\": \"32788\", \"group\": \"S-1-5-
21-2233347455-2266964949-1780268902-69700\", \"ignore_paths\": [
\"/parent/child2\" ], \"owner\": \"S-1-5-21-2233347455-2266964949-
1780268902-69304\", \"propagation_mode\": \"propagate\"}'
```

JSON output example

```
{
  "job": {
    "uuid": "3015c294-5bbc-11eb-9c4e-0050568e8682",
    "_links": {
      "self": {
        "href": "/api/cluster/jobs/3015c294-5bbc-11eb-9c4e-0050568e8682"
      }
    }
  }
}
```

Step 2: Retrieve the status of the job

Perform the workflow Get job instance and confirm the `state` value is `success`.

Update security descriptor information

You can update a specific security descriptor to a specific file or folder, including the primary owner, group, or control flags.

Step 1: Update the security descriptor

HTTP method and endpoint

This REST API call uses the following method and endpoint.

HTTP method	Path
PATCH	/api/protocols/file-security/permissions/{svm.uuid}/{path}

Processing type

Asynchronous

Additional input parameters for curl examples

In addition to the parameters common with all REST API calls, the following parameters are also used in the curl example in this step.

Parameter	Type	Required	Description
$SVM_ID	Path	Yes	This is the UUID of the SVM containing the file.
$FILE_PATH	Path	Yes	This is the path to the file or folder.

Curl example

```
curl --request POST --location "https://$FQDN_IP/api/protocols/file-
security/permissions/$SVM_ID/$FILE_PATH?return_timeout=0" --include
--header "Accept */*" --header "Authorization: Basic $BASIC_AUTH" --data
'{ \"control_flags\": \"32788\", \"group\": \"everyone\", \"owner\":
\"user1\"}'
```

JSON output example

```
{
  "job": {
    "uuid": "6f89e612-5bbd-11eb-9c4e-0050568e8682",
    "_links": {
      "self": {
        "href": "/api/cluster/jobs/6f89e612-5bbd-11eb-9c4e-0050568e8682"
      }
    }
  }
}
```

Step 2: Retrieve the status of the job

Perform the workflow Get job instance and confirm the `state` value is `success`.

Delete an access control entry

You can delete an existing Access Control Entry (ACE) from a specific file or folder. The change propagates to any child objects.

Step 1: Delete the ACE

HTTP method and endpoint

This REST API call uses the following method and endpoint.

HTTP method	Path
DELETE	/api/protocols/file-security/permissions/{svm.uuid}/{path}

Processing type

Asynchronous

Additional input parameters for curl examples

In addition to the parameters common with all REST API calls, the following parameters are also used in the curl example in this step.

Parameter	Type	Required	Description
$SVM_ID	Path	Yes	This is the UUID of the SVM containing the file.
$FILE_PATH	Path	Yes	This is the path to the file or folder.

Curl example

```
curl --request DELETE --location "https://$FQDN_IP/api/protocols/file-
security/permissions/$SVM_ID/$FILE_PATH?return_timeout=0" --include
--header "Accept */*" --header "Authorization: Basic $BASIC_AUTH" --data
'{ \"access\": \"access_allow\", \"apply_to\": { \"files\": true,
\"sub_folders\": true, \"this_folder\": true }, \"ignore_paths\": [
\"/parent/child2\" ], \"propagation_mode\": \"propagate\"}'
```

JSON output example

```
{
  "job": {
    "uuid": "3015c294-5bbc-11eb-9c4e-0050568e8682",
    "_links": {
      "self": {
        "href": "/api/cluster/jobs/3015c294-5bbc-11eb-9c4e-0050568e8682"
      }
    }
  }
}
```

Step 2: Retrieve the status of the job

Perform the workflow Get job instance and confirm the `state` value is `success`.

Copyright information

Trademark information