Using App Volumes API

You can find the most up-to-date technical documentation on the website at: https://docs.com/

Contents

About This Programming Guide

1

The *App Volumes® API Programming Guide* provides information about the App Volumes REST APIs, and how to construct and use the APIs.

Intended Audience

This information is intended for administrators and programmers who want to configure and manage App Volumes programmatically using the available REST APIs.

Technical Publications Glossary

VMware Technical Publications provides a glossary of terms that might be unfamiliar to you. For definitions of terms as they are used in technical documentation, go to http://www.com/support/pubs.

Understanding the App Volumes API

2

App Volumes uses REST APIs to make HTTP requests to the server and to retrieve the required information from the server. You can use the App Volumes APIs to automate workflows within your code.

Overview of REST APIs

REST is an acronym for Representational State Transfer and describes an architectural style characteristic of applications that use the Hypertext Transfer Protocol (HTTP) to exchange serialized representations of objects between a client and a server.

App Volumes API Reference is available at: https://code.com/apis.

Connecting to the App Volumes API

To connect to the App Volumes API and retrieve data from the API, you need to have a client which uses the HTTP/URL syntax.

You can use any client that meets this requirement, such as cURL or Postman.

You must first create an authenticated session to establish a connection. After a successful authentication, you will receive a session cookie. Note down the session cookie information for future use.

Note `/cv_api/version` can be used even when App Volumes Manager is not configured. Hence to use this API, an authenticated session is not necessary.

Depending on the client you use, you may have to save the session cookie in a text file and pass the text file in your API requests.

See your client documentation for specific information about saving and referencing the session cookie.

You can perform the following operations using the App Volumes APIs:

- List All Operating Systems
- View System Activity

- AppStack Operations
  - Retrieve AppStack Information
  - List All Available AppStacks
  - List AppStack Assignments
  - Assign or Unassign an AppStack
- Writable Volume Operations
  - List Writable Volumes and Datastore information
  - Retrieve Writable Volume Information
  - Expand a Writable Volume
  - Update a Writable Volume
  - Delete a Writable Volume

App Volumes API Responses

3

All HTTP responses from App Volumes include a status code and, depending on the request, might include a body or a URL. Some responses might be empty.

HTTP Response Codes

The App Volumes API sends the following HTTP response codes when you send an API request.

Table 3-1. HTTP Status Codes

Status Code	Status Description
`200 OK`	The request is valid. Some APIs send this code for both a successful response and if there are errors in completing the request. The accompanying message in the document body indicates success or failure.
`400 Missing ID parameter`	The request body is missing key parameters required to complete the request.
`400 Bad session parameters`	Usually indicates a missing or invalid parameter.
`400 Bad request`	Missing required data.
`500 Server error`	A server error with additional details provided in the response body.
`404 Variable not found`	Unable to locate the desired volume or AppStack, for example.
`403 Session expired. Create a session and make the request with the _session_id cookie.`	Expired session. User must log in again.

Getting Started with the API

4

Before you can begin to use the App Volumes APIs, you must create an authenticated API session and secure a channel between the browser and the App Volumes server.

You must first POST a request with a user name and password to create an authentication session for the user. You can send further API requests only after you receive a successful response.

This chapter includes the following topics:

Create an Authorized API Session

You must create an authorized API session for a user before you can send other API requests.

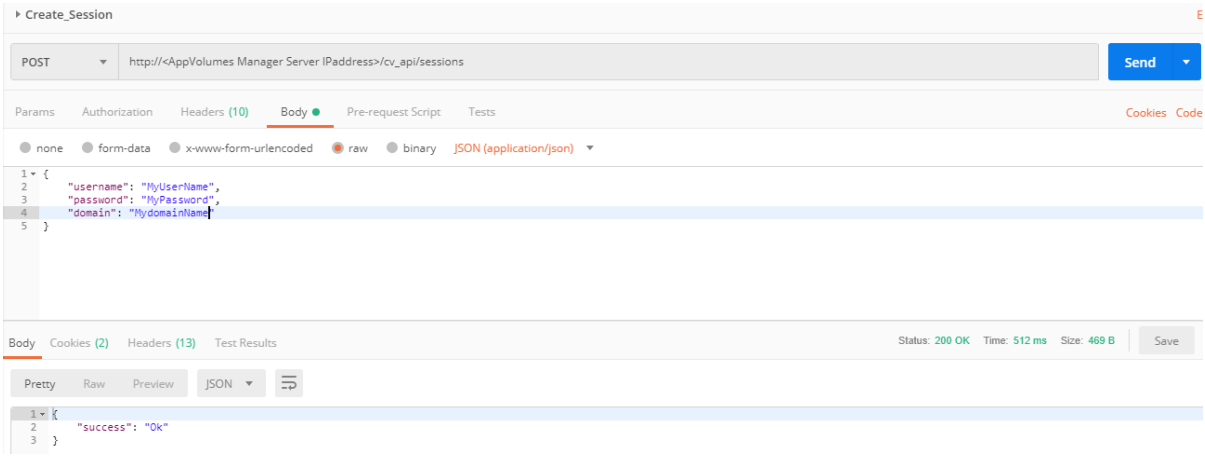
Prerequisites

- You must know the user name and password of the user for whom you want to create the authorized session.
- The account for which you want to create a session must be in the App Volumes administrators group.

Procedure

1. Send the following POST request to the App Volumes server. Include the user name and password in the request body.

   ```
   POST AV_Manager_URL/cv_api/sessions
   ```



2 Examine the response. A successful request returns a session cookie along with the message `"200 OK"`.

If there is an error, a 400 HTTP status code is returned with an error message. You might get an error if one of the following is true, for example:

- If the required parameters are missing.
- If the account for which you are trying to create an authentication does not have sufficient privileges.
- If the user name or password is invalid.

What to do next

Note Depending on the client you are using to connect to the App Volumes API, you might need to save the session cookie as a text file to reference it later.

Delete an Authorized API Session

Delete an authorized session for a user.

Prerequisites

You must know the user name of the user for whom you want to delete the session.

Procedure

1 Send the following DELETE request to the App Volumes server. The *username* is embedded in the cookie header.

`DELETE` *`AV_Manager_URL`*`/cv_api/sessions`

2 Examine the response. A successful request returns HTTP status 200 and a success message. For example:

```
{
"application/json": {
        "success": "Destroying session for \USERNAME\""
}
```

General App Volumes APIs

5

This section lists the general App Volumes APIs.

This chapter includes the following topics:

List All Operating Systems

List all the available operating systems that an AppStack or Writable Volume can be mounted on. You can also see information associated with the OS such as the major and minor version of the OS, the processor architecture, and so on.

Procedure

1 Make the following GET request to the App Volumes server:

`GET AV_Manager_URL/cv_api/operating_systems`

2 Examine the response. A successful response returns HTTP status 200 and includes the OS id, major and minor version of the OS, processor architecture and so on. For example:

```
[
  [
    {
      "id": "string",
      "major_version": "10",
      "minor_version": "0",
      "name": "Windows 10 (x64)",
      "proc_arch": "9",
      "product_type": "1",
      "created_at": "2018-03-29T23:47:32Z",
      "updated_at": "2018-03-29T23:47:32Z"
    }
  ]
]
```

View System Activity

Retrieve the activity logs to view system activity.

The Activity Log contains information about user logins, computer power-ups, and volume attachments. System messages include messages and errors generated from internal events such as polling for domain controllers, Active Directory access, and so on.

You can also filter the logs by user name if you want. Look up the **Down-level Logon Name** of the user in the Active Directory services, and find the user name.

Procedure

1 Make a GET request to retrieve the activity logs.

```
GET AV_Manager_URL/cv_api/activity_logs
```

2 Examine the response.

A successful response returns a list of records with HTTP status code 200.

For example:


```
{   "actlogs": {
        "logs": [
         {
           "source_id": 0,
           "source_type": "User",
           "source_url": "/directory#/Users/1",
           "source_name": "Administrator",
           "target_id": 0,
           "target_type": "Snapvol",
           "target_url": "/directory#/Users/1",
           "target_name": "Administrator",
           "target_description": "Administrator",
           "action": "Assign",
           "log": "Success",
           "event_time": "2000-01-01 12:00:00 -0700",
           "event_time_human": "January 1, 2000 12:00PM",
           "admin_user_upn": "DOMAIN\\Administrator",
           "admin_user_name": "Administrator",
           "admin_user_url": "/directory#/Users/2"
         }
        ]
      }
 }
```


View Version Details of App Volumes Manager

Use this API to view the version details of App Volumes Manager. In addition to the version details, you can also view other information such as copyright, configuration status of the App

Volumes Manager, uptime (duration for which the App Volumes Manager has been running), and the UUID of the database configured in App Volumes Manager.

You can use this API even when App Volumes Manager is not configured. An authenticated session is not required to run this API.

Procedure

1. Make the following GET request to the App Volumes server:

   `GET AV_Manager_URL/cv_api/version`

2. Examine the response.

   A successful response returns HTTP status `200` and includes the version, copyright, configured status, uptime, and database UUID.

   For example:

```
{
"version": {
  "version": "2.17.0 (development)",
  "internal": "2.17.0",
  "copyright": "Copyright 2011–2019",
  "configured": "true",
  "uptime": "less than a minute",
  "database_uuid": "654723bc-09de-4d9b-806c-9cd1d231a0de"
  }
 }
```

Working with AppStack APIs

6

You can perform various operations on AppStacks using the APIs.

You can get information about an AppStack, assign, or unassign an AppStack to a target, list all available AppStacks, and so on.

This chapter includes the following topics:

Retrieve AppStack Information

You can retrieve information about an AppStack such as the date and time of creation of the AppStack, the total number of assignments, the attachment limit, size, and so on. You can also get the name and ID of the operating system and other optional information if it was included when the AppStack was created.

Prerequisites

You must know the ID of the AppStack for which you want to retrive the information. If you do not know the AppStack ID, use the `GET /cv_api/appstacks` API to retrieve the ID.

Procedure

1 Make a GET request with the AppStack id to retrieve the information:

```
GET AV_Manager_URL/cv_api/appstacks/id
```

2 Examine the response. A successful response returns the AppStack information with an HTTP status 200. For example:

```
[
  {
```

```
    "id": 0,
    "name": "zippertool",
    "name_html": "zippertool",
    "path": "cloudvolumes/apps",
    "datastore_name": "data.1",
    "filename": "zippertool.vmdk",
    "file_location": "[data.1] cloudvolumes/apps/zippertool.vmdk",
    "description": "Archive tool",
    "status": "enabled",
    "created_at": "2016-11-23 13:10:13 -0800",
    "created_at_human": "Nov 23 2016",
    "mounted_at": "2016-11-23 13:10:13 -0800",
    "mounted_at_human": "Nov 23 2016",
    "mount_count": 1,
    "size_mb": 78,
    "template_version": "2.10.0.709",
    "assignments_total": 1,
    "attachments_total": 1,
    "attachment_limit": 10,
    "location_count": 1,
    "application_count": 1,
    "application_icons": [
      "/snapvols/9/zippertool.ico"
    ],
    "volume_guid": "{9b1b40b0-e2f8-4fe6-b743-c55be9c2e000}",
    "template_file_name": "[data.1] cloudvolumes/app_templates/template.vmdk",
    "agent_version": "2.14.0.0",
    "capture_version": "2.10.0.0",
    "primordial_os_id": 4,
    "primordial_os_name": "Windows 8.1 (x64)",
    "oses": [
      {
        "id": 0,
        "name": "Windows 8.1 (x64)"
      }
    ],
    "parent_snapvol": "<a href=\"/volumes#/AppStacks/1\" title=\"\">zippertool</a>",
    "provisioning_duration": "3 minutes",
    "provisioning_since": "Mar 06 2018",
    "provisioning_computer": "<a href=\"/directory#/Computers/1\" title=\"Compu1\">DOMAIN\
\Compu1</a>"
  }
]
```


If App Volumes is unable to locate the AppStack, an HTTP error code 404 is returned with an error message. For example, `Unable to locate the Volume ID`.

List All Available AppStacks

You can get a list of all available AppStacks and the information associated with each AppStack such as the path, associated datastore, size, and so on.

Procedure

1 Make a GET request to retrieve the list of AppStacks.

```
GET AV_Manager_URL/cv_api/appstacks
```

2 Examine the response. A successful response returns an HTTP status 200 with list of available AppStacks and associated information about the AppStacks. For example :


```
[
  [
    {
      "id": 0,
      "name": "zippertool",
      "name_html": "zippertool",
      "path": "cloudvolumes/apps",
      "datastore_name": "data.1",
      "status": "enabled",
      "created_at": "2016-11-23 13:10:13 -0800",
      "created_at_human": "Nov 23 2016",
      "mounted_at": "2016-11-23 13:10:13 -0800",
      "mounted_at_human": "Nov 23 2016",
      "mount_count": 1,
      "size_mb": 78,
      "template_version": "2.10.0.709",
      "assignments_total": 1,
      "attachments_total": 1,
      "attachment_limit": 10
    }
  ]
]
```


Example: List AppStacks For a Specific Operating System

Instead of listing all available AppStacks, you can also list AppStacks that are attached to a specific operating system.

List AppStack Assignments

Get a list of machines that the AppStack is assigned to, and associated AppStacks information such as the mount prefix, the mount keyword that is used to assign the AppStack, and so on.

Prerequisites

You must know the ID of the AppStack. If you do not know the AppStack ID, use the `GET AV_Manager_URL/cv_api/appstacks` API to retrieve the ID.

Procedure

1 Make a GET request to retrieve the list of assignments of an AppStack.

```
GET AV_Manager_URL/cv_api/appstacks/id/assignments
```

2 Examine the response. A successful response returns the list of assignments for the AppStack with HTTP status 200. For example:

```
[
  [
    {
      "name": "<a href=\"/directory#/Users/1\" title=\"DOMAIN\\jdoe\">DOMAIN\\jdoe</a>",
      "entity_type": "user",
      "mount_prefix": "compu",
      "unmount_computer_assignments": true,
      "mount_keyword": "compu"
    }
  ]
]
```

Assign an AppStack

Assign an AppStack to an entity. The entity can be Users, Groups, Computers, or Organizational Units (OUs).

Prerequisites

You must know the value of the following parameters:

- The ID of the AppStack that you want to assign.
- The path of the entity to which you will assign the AppStack.
- (Optional) The *mount_prefix* variable - If you want to limit the AppStack attachment to computers with a specific prefix, use the *mount_prefix* variable and include it in the POST request. Then when the entity, to whom the AppStack is assigned logs in and if the prefix matches the the target machine, the AppStacks are attached and the entity will receive the assigned AppStacks. If you do not include the *mount_prefix* variable, the AppStack will be attached to all entities for whom assignment is available.

Procedure

1 Make a POST request to the App Volumes server to assign an AppStack. Include the type of action (assign in this case), and optionally, the *mount_prefix* of the target in the body of the request.

```
POST AV_Manager_URL/cv_api/assignments
```

2 Examine the response. A successful request returns an HTTP status code 200, and a message indicating success or failure. If you do not provide the required data in the request body, an HTTP status code 400 is returned.

Example: Assign an AppStack

The following example shows an AppStack assignment to a Group entity with target computers which have the prefix *DESK*.

```
Request
POST AV_Manager_URL/cv_api/assignments

Request body:
{
  "id": 0,
  "action_type": "assign"
  "assignments": [
    {
      "entity_type": "Group",
      "path": "CN=test1,CN=Users,DC=test,DC=local"
    }
  ],
  "mount_prefix": "DESK",
  "rtime": false
}
```

Note If you want to assign an AppStack and attach it immediately to a live desktop session, use the parameter `rtime` and set it to **true**.

Unassign an AppStack

Unassign an AppStack from an entity. The entity can be Users, Groups, Computers and Organizational Units(OUs).

Prerequisites

You must know the value of the following parameters:

- The ID of the AppStack that you want to unassign.
- The *path* of the entity that you are going to unassign from the AppStack.
- (Optional) The `mount_prefix` parameter. If the `mount_prefix` parameter is specified in the POST request, only the AppStack assignment that matches this value will be unassigned. If there is no match, no Appstacks will be unassigned.

  If the `mount_prefix` parameter is not specified, then all assignments that match the AppStack IDs will be unassigned, regardless of the `mount_prefix` value specified during the assignment.

Procedure

1. Make a POST request to the App Volumes server to unassign an AppStack. Include the type of action (unassign in this case), and the *mount_prefix* of the target in the body of the request.

   `POST AV_Manager_URL/cv_api/assignments`

2 Examine the response. A successful request returns an HTTP status code 200, and a message indicating success or failure. If you do not provide the required data in the request body, an HTTP status code 400 is returned.

Example: Unassign an AppStack

The following example shows a POST request to unassign the AppStack from all target entities with prefix DESK.

```
Request
POST AV_Manager_URL/cv_api/assignments

Request body:
{
  "id": 0,
  "action_type": "unassign",
  "assignments": [
    {
      "entity_type": "Group",
      "path": "CN=test1,CN=Users,DC=test,DC=local"
    }
  ],
  "mount_prefix": "DESK",
  "rtime": false
}
```

Note If you want to detach an AppStack immediately and unassign it from a live desktop session, use the parameter `rtime` and set it to **true**.

The following message is displayed after an AppStack is successfully unassigned:

```
{ "successes": "Unassigned 1 AppStack(s)" }
```

Working with Writable Volumes APIs

7

You can perform various operations on the Writable Volumes using the APIs.

You can get detailed information about the Writable Volume, and expand, update, or delete a volume.

This chapter includes the following topics:

- List Writable Volumes and DataStore Information
- Retrieve Writable Volume Information
- Expand a Writable Volume
- Update a Writable Volume
- Delete a Writable Volume

List Writable Volumes and DataStore Information

Retrieve a list of Writable Volumes and the associated datastore information such the volume GUID, the date and time it was created, whether the volume is attached, and so on.

Procedure

1. Send the following GET request to retrieve the list of Writable Volumes.

   `GET AV_Manager_URL/cv_api/writables`

2. Examine the response.

   A list of volumes and associated information is returned. For example:

```
{
  "datastores": {
    "total_count": 1,
    "warning_count": 0,
    "critical_count": 0,
    "writable": {
      "name": "SSD Disk Array RAID0",
      "datacenter": "CloudVolumes",
      "path": "cloudvolumes/writable",
      "current": "true"
```

```
    },
    "writable_volumes": [
      {
        "id": 0,
        "name": "SNAPVOLUMES\test_user",
        "name_html": "SNAPVOLUMES\test_user",
        "title": "SNAPVOLUMES\test_user",
     "title_html": "SNAPVOLUMES\test_user",
        "owner": "<a href=\\\"/directory#/Users/2\\\" title=\\\"test_user\\\">TESTDOMAIN\\\
\test_user</a>",
        "link": "/directory#/Users/2",
        "owner_name": "TestUser",
        "owner_type": "User",
        "owner_upn": "SNAPVOLUMES\test_user",
        "owner_upn_html": "SNAPVOLUMES\test_user",
        "owner_object_guid": "8da04bfa-bbbe-45c3-9cbf-2d3d03897b84",
        "created_at": "2016-11-23 13:10:13 -0800",
        "created_at_human": "Nov 23 2016",
        "updated_at": "2016-11-23 13:10:13 -0800",
        "updated_at_human": "Nov 23 2016",
        "mounted_at": "2016-11-23 13:10:13 -0800",
        "mounted_at_human": "Nov 23 2016",
        "attached": "Attached",
        "status": "missing",
        "size_mb": 0,
        "mount_count": 0,
        "free_mb": 0,
        "total_mb": 0,
        "percent_available": "Unknown",
        "template_version": "2.10.0.709",
        "version_count": 0,
        "filename": "snapvolumes_test_user_15.vmdk",
        "path": "cloudvolumes/writable",
        "datastore_name": "SSD Disk Array RAID8",
        "volume_guid": "volumeguid-8-volumeguid",
        "datastore_host": "null",
        "error-action": "string",
        "busy": false,
        "state": "Moving"
      }
    ]
  }
}
```


Retrieve Writable Volume Information

Retrieve information about a Writable Volume such as the name, owner, size, path to the Writable Volume, and so on.

Prerequisites

You must know the ID of the Writable Volume for which you want to retrieve the information.

Procedure

- Send the following GET request to retrieve the list of volumes.

  `GET AV_Manager_URL/cv_api/writables/id`

  If the request is completed successfully, an HTTP status code of 200 is returned with information about the volume.

  For example:


```
{
  "id": 0,
  "name": "SNAPVOLUMES\test_user",
  "name_html": "SNAPVOLUMES\test_user",
  "title": "SNAPVOLUMES\test_user",
  "title_html": "SNAPVOLUMES\test_user",
  "owner": "<a href=\\\"/directory#/Users/2\\\" title=\\\"test_user\\\">TESTDOMAIN\\\\test_user</
a>",
  "link": "/directory#/Users/2",
  "owner_name": "TestUser",
  "owner_type": "User",
  "owner_upn": "SNAPVOLUMES\test_user",
  "owner_upn_html": "SNAPVOLUMES\test_user",
  "description": "My writable volume.",
  "created_at": "2016-11-23 13:10:13 -0800",
  "created_at_human": "Nov 23 2016",
  "updated_at": "2016-11-23 13:10:13 -0800",
  "updated_at_human": "Nov 23 2016",
  "mounted_at": "2016-11-23 13:10:13 -0800",
  "mounted_at_human": "Nov 23 2016",
  "mounted_on": "Machine <DESKTOP-VM123> (#{1234567812345678})",
  "attached": "Attached",
  "status": "missing",
  "version_count": 0,
  "version_tag": 0,
  "block_login": true,
  "enable_version": true,
  "mount_prefix": "DESKTOP",
  "defer_create": true,
  "size_mb": 0,
  "template_version": "2.10.0.709",
  "datastore_name": "SSD Disk Array RAID8",
  "machine_manager_host": "0.0.0.0",
  "machine_manager_type": "Machine Manager",
  "path": "cloudvolumes/writable",
  "filename": "snapvolumes_test_user_15.vmdk",
  "file_location": "[SSD Disk Array RAID8] cloudvolumes/writable/snapvolumes_test_user_15.vmdk",
  "mount_count": 0,
  "type": "null",
  "display_type": "",
  "template_file_name": "[SSD Disk Array RAID8] cloudvolumes/writable_templates/
template_uia_only_persistent.vmdk",
  "protected": true,
  "free_mb": 0,
  "total_mb": 0,
```

```
  "percent_available": "Unknown",
  "can_expand": true,
  "storage_group": "Storage group 1",
  "storage_group_members": 2,
  "primordial_os_id": 0,
  "primordial_os_name": "Windows 8.1 (x64)",
  "oses": [
    {
      "id" 0,
      "name": "windows 8.1 (x64)"
}
  ],
  "busy": false,
  "state": "Moving"
}
```

Expand a Writable Volume

Increase the size of a Writable Volume.

Prerequisites

Before you expand the Writable Volume, ensure the following:

- The Writable Volume is not attached.
- You have logged out from the Writable Volume.
- The App Volumes Manager is licensed and configured correctly.

Procedure

1 Make a POST request to expand the volume. Include the ID of the volume and the size you want to expand the volume to in the POST request body.

```
POST AV_Manager_URL/cv_api/writables/grow
```

2 Examine the response. A successful request returns 200 with a message. For example:

```
{
"success" :[
        "Successfully expanded the Writable Volume to 50 MB"
]
}
```

You might also get a HTTP response of 200 with warnings or errors if you have either not detached the volume or logged out from the Writable Volume.

```
 "warnings": [ "Writable Volume XYZ is attached. Make sure you shut down/logoff XYZ." ]  }
```

If the request is missing parameters or if App Volumes Manager is not configured correctly, a HTTP response of 400 is returned.

```
"errors": [ "Error expanding Writable Volume XYZ" ]
```

Update a Writable Volume

You can update some Writable Volume settings, such as description, whether users can log in if there is a failure, the prefix value of the computer the volume should be mounted to, and the operating system.

Prerequisites

Ensure that the Writable Volume you want to modify is not attached and that you have logged off from the given volume.

Procedure

1 Make a PUT request to update the volume. Include the properties of the volume that you want to update in the request body.

```
PUT AV_Manager_URL/cv_api/writables/id
```

You can update the following properties of the Writable Volume:

Option	Description
Description (string)	Description of the Writable Volume.
error-action (string)	■ `continue_silently` - skip attachments and do not alert the user. ■ **continue_alert** - alert the end user that the writable was not attached. ■ **disable_and_alert** - disable all virtualization and display an alert. ■ **disable_and_alert_on_error** - disable all virtualization and display an alert only for error cases (conflicts due to multiple logins are not considered an error). Must set as "" if a) owner_type is "Computer" or b) block_login parameter is set to "1".
block_login (integer)	■ Set to '0' to prevent users from logging in if there are failures related to this Writable Volume. ■ Set to '1' to allow users to log in even if there are failures related to this Writable Volume.
mount_prefix	Prefix used to filter which computer the Writable Volume can be mounted to.
OS	The IDs of OSes the Writable Volume can be mounted to.

2 Examine the response. A successful request returns HTTP response 200 with the following message:

```
{
"success":OK"[
    "Successfully updated the Writable Volume.
]
}
```

Delete a Writable Volume

You can delete a Writable Volume. Once a volume is deleted , it is immediately detached from all computers. All associated data and settings are also deleted permanently.

Prerequisites

- Ensure that the Writable Volume you want to delete is not in use by any user or computer.
- You must know the ID of the Writable Volume that you want to delete.

Procedure

1 Send the following DELETE request to delete a volume. You can also set the volume to be deleted in the background.

```
DELETE /cv_api/writables/id
```

2 Examine the response. If the volume is deleted successfully, you will receive an HTTP status code 200 with detailed information about the deleted volume. For example:

```
"success": [
      {
        "id": 0,
        "name": "SNAPVOLUMES\test_user",
        "name_html": "SNAPVOLUMES\test_user",
        "title": "SNAPVOLUMES\test_user",
        "title_html": "SNAPVOLUMES\test_user",
        "owner": "<a href=\\\"/directory#/Users/2\\\" title=\\\"test_user\\\">TESTDOMAIN\\\
\test_user</a>",
        "link": "/directory#/Users/2",
        "owner_name": "TestUser",
        "owner_type": "User",
        "owner_upn": "SNAPVOLUMES\test_user",
        "owner_upn_html": "SNAPVOLUMES\test_user",
        "description": "My writable volume.",
        "created_at": "2016-11-23 13:10:13 -0800",
        "created_at_human": "Nov 23 2016",
        "updated_at": "2016-11-23 13:10:13 -0800",
        "updated_at_human": "Nov 23 2016",
        "mounted_at": "2016-11-23 13:10:13 -0800",
        "mounted_at_human": "Nov 23 2016",
        "mounted_on": "Machine <DESKTOP-VM123> (#{1234567812345678})",
        "attached": "Attached",
```

```
        "status": "missing",
        "version_count": 0,
        "version_tag": 0,
        "block_login": true,
        "enable_version": true,
        "mount_prefix": "DESKTOP",
        "defer_create": true,
        "size_mb": 0,
        "template_version": "2.10.0.709",
        "datastore_name": "SSD Disk Array RAID8",
        "machine_manager_host": "0.0.0.0",
        "machine_manager_type": "Machine Manager",
        "path": "cloudvolumes/writable",
        "filename": "snapvolumes_test_user_15.vmdk",
        "file_location": "[SSD Disk Array RAID8] cloudvolumes/writable/
snapvolumes_test_user_15.vmdk",
        "mount_count": 0,
        "type": "null",
        "display_type": "",
        "template_file_name": "[SSD Disk Array RAID8] cloudvolumes/writable_templates/
template_uia_only_persistent.vmdk",
        "protected": true,
        "free_mb": 0,
        "total_mb": 0,
        "percent_available": "Unknown",
        "can_expand": true,
        "storage_group": "Storage group 1",
        "storage_group_members": 2,
        "primordial_os_id": 0,
        "primordial_os_name": "Windows 8.1 (x64)",
        "oses": [
          {}
        ]
      }
    ]
```


You may also receive a status 200 if the operation has errors, if the volume is missing, for example.


```
{
  "error": "Unable to delete 1 volume",
   {
        "id": 0,
        "name": "SNAPVOLUMES\test_user",
        "name_html": "string",
        "title": "SNAPVOLUMES\test_user",
        "title_html": "string",
        "owner": "string",
        "link": "/directory#/Users/2",
        "owner_name": "TestUser",
        "owner_type": "User",
        "owner_upn": "SNAPVOLUMES\test_user",
        "owner_upn_html": "SNAPVOLUMES\test_user",
        "description": "string",
        "created_at": "string",
```

```
          "created_at_human": "string",
          "updated_at": "string",
          "updated_at_human": "string",
          "mounted_at": "string",
          "mounted_at_human": "string",
          "mounted_on": "Machine <DESKTOP-VM123> (#{1234567812345678})",
          "attached": "Attached",
          "status": "missing",
          "version_count": 0,
          "version_tag": 0,
          "block_login": true,
          "enable_version": true,
          "mount_prefix": "string",
          "defer_create": true,
          "size_mb": 0,
          "template_version": "string",
          "datastore_name": "SSD Disk Array RAID8",
          "machine_manager_host": "string",
          "machine_manager_type": "string",
          "path": "cloudvolumes/writable",
          "filename": "snapvolumes_test_user_15.vmdk",
          "file_location": "string",
          "mount_count": 0,
          "type": "string",
          "display_type": "string",
          "template_file_name": "string",
          "protected": true,
          "free_mb": 0,
          "total_mb": 0,
          "percent_available": "string",
          "can_expand": true,
          "storage_group": "string",
          "storage_group_members": "string",
          "primordial_os_id": 0,
          "primordial_os_name": "string",
          "oses": [
            {}
          ]
        }
      ],
       }
```


Example: Schedule a Writable Volume to be Deleted


```
"scheduled": [
  {
         "id": 0,
       "name": "SNAPVOLUMES\test_user",
      "name_html": "SNAPVOLUMES\test_user",
      "title": "SNAPVOLUMES\test_user",
      "title_html": "SNAPVOLUMES\test_user",
        "owner": "<a href=\\\"/directory#/Users/2\\\" title=\\\"test_user\\\">TESTDOMAIN\\\
\test_user</a>",
```

```
      "link": "/directory#/Users/2",
      "owner_name": "TestUser",
      "owner_type": "User",
      "owner_upn": "SNAPVOLUMES\test_user",
      "owner_upn_html": "SNAPVOLUMES\test_user",
      "description": "My writable volume", "created_at": "2016-11-23 13:10:13 -0800",
      "created_at_human":
      "Nov 23 2016",
       "updated_at": "2016-11-23 13:10:13 -0800",
      "updated_at_human": "Nov 23 2016",
      "mounted_at": "2016-11-23 13:10:13 -0800",
      "mounted_at_human": "Nov 23 2016",
     "mounted_on": "Machine <DESKTOP-VM123>
      (#{1234567812345678})",
      "mount_count": 0,
      "attached": "Attached",
      "status": "enabled",
      "version_count": 0,
      "version_tag": 0,
      "block_login": true,
       "error_action": "",
       "enable_version": false,
      "mount_prefix": "DESKTOP",
      "defer_create": true,
      "size_mb": 0,
      "template_version": "2.10.0.709",
      "datastore_name": "SSD Disk Array RAID8",
      "machine_manager_host": "0.0.0.0",
      "machine_manager_type": "Machine Manager",
      "path": "cloudvolumes/writable",
      "filename": "snapvolumes_test_user_15.vmdk",
      "file_location": "[SSD Disk Array RAID8]
      cloudvolumes/writable/snapvolumes_test_user_15.vmdk",
      "template_file_name": "[SSD Disk Array RAID8] cloudvolumes/writable_templates/
template_uia_only_persistent.vmdk",
      "protected": true,
      "free_mb": 0,
      "total_mb": 0,
      "percent_available": "Unknown",
       "can_expand": true,
       "storage_group":
      "Storage group 1",
      "storage_group_members": 2,
      "primordial_os_id": 15,
      "primordial_os_name": "Windows 8.1 (x64)",
      "oses": [
      {
      "id": 0,
      "name": "Windows 8.1 (x64)"
      }
    ],
   "busy": false,
   "state": "Moving"
 }
]
```

```
}
 }
```