Basic operational characteristics

ONTAP Select

David Peterson
November 21, 2019

Table of Contents

Basic operational characteristics 1

Basic operational characteristics

While REST establishes a common set of technologies and best practices, the details of each API can vary based on the design choices. You should be aware of the details and operational characteristics of the ONTAP Select Deploy API before using the API.

Hypervisor host versus ONTAP Select node

A *hypervisor host* is the core hardware platform that hosts an ONTAP Select virtual machine. When an ONTAP Select virtual machine is deployed and active on a hypervisor host, the virtual machine is considered to be an *ONTAP Select node*. With version 3 of the Deploy REST API, the host and node objects are separate and distinct. This allows a one-to-many relationship, where one or more ONTAP Select nodes can run on the same hypervisor host.

Object identifiers

Each resource instance or object is assigned a unique identifier when it is created. These identifiers are globally unique within a specific instance of ONTAP Select Deploy. After issuing an API call that creates a new object instance, the associated id value is returned to the caller in the `location` header of the HTTP response. You can extract the identifier and use it on subsequent calls when referring to the resource instance.

The content and internal structure of the object identifiers can change at any time. You should only use the identifiers on the applicable API calls as needed when referring to the associated objects.

Request identifiers

Every successful API request is assigned a unique identifier. The identifier is returned in the `request-id` header of the associated HTTP response. You can use a request identifier to collectively refer to the activities of a single specific API request-response transaction. For example, you can retrieve all the event messages for a transaction based on the request id.

Synchronous and asynchronous calls

There are two primary ways that a server performs an HTTP request received from a client:

- Synchronous
  The server performs the request immediately and responds with a status code of 200, 201, or 204.
- Asynchronous
  The server accepts the request and responds with a status code of 202. This indicates the server has accepted the client request and started a background task to complete the request. Final success or failure is not immediately available and must be determined through additional API calls.

Confirming the completion of a long-running job

Generally, any operation that can take a long time to complete is processed asynchronously using a background task at the server. With the Deploy REST API, every background task is anchored by a Job object which tracks the task and provides information, such as the current state. A Job object, including its unique identifier, is returned in the HTTP response after a background task is created.

You can query the Job object directly to determine the success or failure of the associated API call. Refer to *asynchronous processing using the Job object* for additional information.

In addition to using the Job object, there are other ways you can determine the success or failure of a request, including:

- Event messages
  You can retrieve all the event messages associated with a specific API call using the request id returned with the original response. The event messages typically contain an indication of success or failure, and can also be useful when debugging an error condition.
- Resource state or status
  Several of the resources maintain a state or status value which you can query to indirectly determine the success or failure of a request.

Security

The Deploy API uses the following security technologies:

- Transport Layer Security
  All traffic sent over the network between the Deploy server and client is encrypted through TLS. Using the HTTP protocol over an unencrypted channel is not supported. TLS version 1.2 is supported.
- HTTP authentication
  Basic authentication is used for every API transaction. An HTTP header, which includes the user name and password in a base64 string, is added to every request.