Technical Report

SANtricity Web Services API

Built-In and Central Management Capabilities

Loren Price, Jolie Gallagher, NetApp
August 2020 | TR-4736

Abstract

This document provides an overview of NetApp® SANtricity® Web Services, an API used for configuring and managing NetApp E-Series and EF-Series storage systems.

NetApp®

TABLE OF CONTENTS

LIST OF TABLES

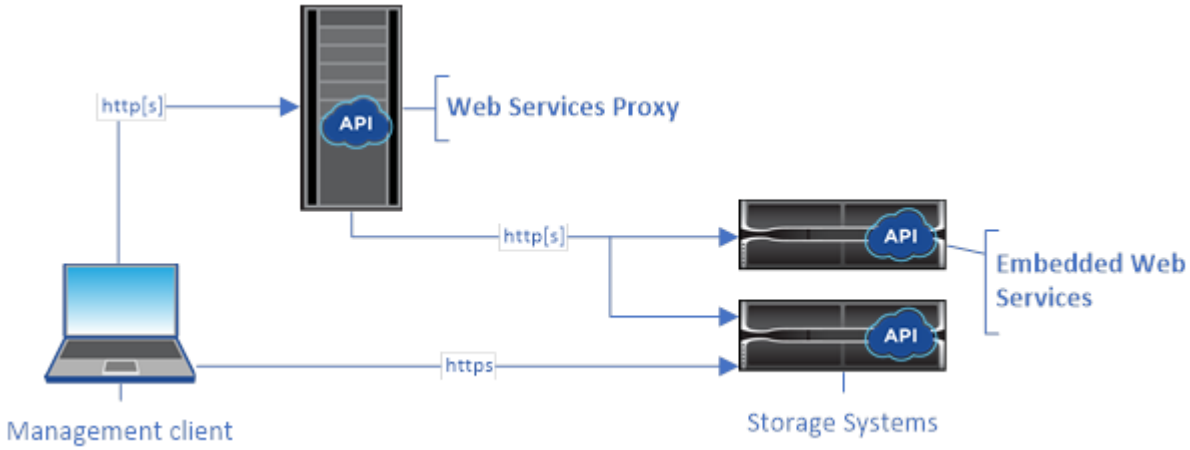
LIST OF FIGURES

1 Overview

NetApp® SANtricity® Web Services is an API that allows you to configure, manage, and monitor NetApp E-Series and EF-Series storage systems. Web Services is a RESTful API that uses the HTTP protocol just like a web browser. The API provides structured output to HTTP requests that use standard HTTP verbs and URL structure.

Web Services is embedded on all E2800/EF280, E5700/EF570, and EF600 series controllers. However, it manages only the array on which the controller is installed. To manage multiple arrays, including a mixture of older and new systems, you can install the SANtricity Web Services Proxy on a Linux or Windows server, and then configure the proxy to communicate with the storage systems.

Figure 1) SANtricity Web Services in a network.



Although Web Services is designed for developers, you do not need to be an experienced API developer to understand and employ its powerful functions. This technical report leads you through the basic concepts and setup tasks so that you can begin using Web Services without first taking an extensive training course.

1.1 Typical Workflows

Whether you intend to write small scripts for single tasks or integrate tasks into a larger application, Web Services provides all the necessary functionality for managing your E-Series storage systems. By issuing API requests against API endpoints, you can complete workflows such as the following:

- **Configuration (initial setup).** After you complete hardware installation, you can use Web Services to push the rest of the configuration from a remote location. Initial setup includes security settings, alerting methods, certificates, and NetApp AutoSupport®. By using the SANtricity Web Services Proxy, you can centralize system configuration and security-related operations for hundreds of storage systems.
- **Provisioning and data protection.** With Web Services, you can streamline the provisioning steps required for providing logical volumes to hosts. You can also streamline other complicated workflows used in data protection functions, such as snapshots and mirroring.
- **Performance monitoring.** You can use Web Services to access the full set of raw data collected in the storage system; then you can perform higher-level data analysis in a more manageable format. Web Services manages all the difficult tasks on the server.
- **Health monitoring.** Web Services provides endpoints to retrieve relevant information about the health status of array components, including state and configuration.

1.2 Terms and Concepts

This document uses the terms and concepts listed in Table 1.

Table 1) Basic terminology.

Term	Definition
API	Application programming interface. A set of protocols and methods that allows developers to communicate with devices. The Web Services API is used to communicate with E-Series storage systems.
Endpoint	Functions that are available through the API. An endpoint includes an HTTP verb plus the URI path. In Web Services, endpoints can include such tasks as discovering storage systems and creating volumes.
HTTP verb	A corresponding action for an endpoint, such as retrieving and creating data. In Web Services, HTTP verbs include POST, GET, and DELETE.
JSON	JavaScript Object Notation. A structured data format, much like XML, that uses a minimal, readable format. Data in Web Services is encoded through JSON.
REST/RESTful	Representational State Transfer (REST). A loose specification that defines an architectural style for an API. Because most REST APIs do not fully adhere to the specification, they are described as “RESTful” or “REST-like.” Generally, a RESTful API is agnostic to programming languages and has the following characteristics: • HTTP-based, following the general semantics of the protocol • Producer and consumer of structured data (JSON, XML, and so on) • Object-oriented (as opposed to operation-oriented) Web Services is a RESTful API that provides access to virtually all the SANtricity management capabilities.
Storage system	Refers to both the physical storage array and the logical representation of the system (its connectivity and state). Web Services can communicate with one or more E-Series storage systems.
SYMbol API	A legacy API for managing E-Series storage systems. The underlying implementation of the Web Services API uses SYMbol.
Web Services	An API that NetApp designed for developers to manage E-Series storage systems. There are two implementations of Web Services; one is embedded on the controller, and one is a separate proxy that can be installed on Linux or Windows.

1.3 Prerequisites

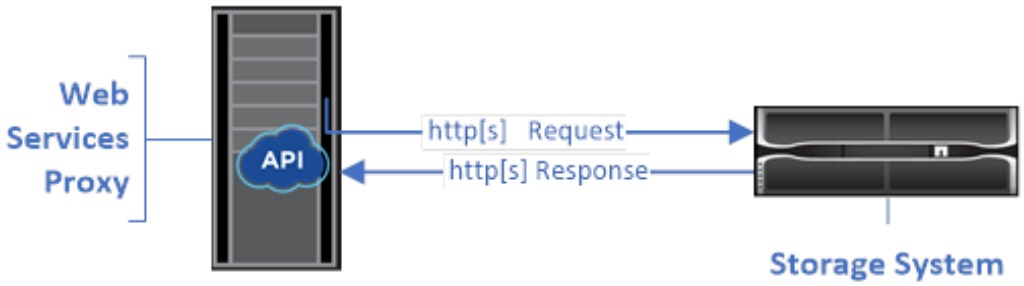
Before using the Web Services API, you should be familiar with the following resources:

- **REST**. For more information about this subject, see the dissertation Architectural Styles and the Design of Network-Based Software Architectures from UC Irvine.
- **JSON**. For more information, see the JSON home page.
- **Programming concepts**. Because Web Services is intended for developer use, you should be familiar with programming language concepts. Java and Python are the most common programming languages used with the Web Services API, but any programming language that can make HTTP requests is sufficient for API interaction.

2 API Basics

In the Web Services API, HTTP communications involve a request-response cycle, as described in this section.

Figure 2) API request and response.



2.1 Requests

Regardless of the programming language or tool used, each call to the Web Services API has a similar structure, with a URL, an HTTP verb, and an Accept header.

URL Path

All requests include a URL, as in the following example, and contain the elements described in Table 2.

```
https://webservices.name.com:8443/devmgr/v2/storage-systems
```

Table 2) Typical URL path.

Area	Description
HTTP transport `https://`	The SANtricity Web Services Proxy allows use of HTTP or HTTPS. The embedded Web Services implementation requires HTTPS for security reasons.
Base URL and port `webservices.name.com:8443`	Every request must be correctly routed to an active instance of Web Services. The FQDN (fully qualified domain name) or the IP address of the instance is required, along with the listening port. By default, Web Services communicates over port 8080 (for HTTP) and port 8443 (for HTTPS). For the Web Services Proxy, both ports can be changed during the proxy installation, because port contention is common on data center hosts running various management applications. For the embedded Web Services implementation, the port on the controller cannot be changed; it defaults to port 8443 for secure connections.

Area	Description
API path `devmgr/v2/storage-systems`	A request is made to a specific REST resource or endpoint in the Web Services API. Most endpoints are in the form `devmgr/v2/<resource>/[id]` The API path consists of three parts: • `devmgr` (Device Manager) is the namespace of the Web Services API. • `V2` denotes the version of the API that you are accessing. You can also use `utils` to access login endpoints. • `storage-systems` is a category in the Web Services documentation.

HTTP Verb

REST interfaces support multiple HTTP verbs (GET, POST, DELETE) for each resource:

- GET requests are used for read-only requests.
- POST requests are used for creating and updating objects and for read requests that might have side effects or security implications.
- DELETE requests are typically used to remove an object from management, remove an object entirely, or reset the state of the object.

Currently, the Web Services API does not support PUT or PATCH. The functionality typically provided by these verbs is provided by POST.

Accept Header

When returning a request body, Web Services returns the data in JSON format (unless otherwise specified). Certain clients default to requesting “text/html” or something similar. In these cases, the API responds with an HTTP code 406, denoting that it cannot provide data in this format. As a best practice, define the Accept header as “application/json” for any cases where you expect JSON as the response type. In other cases where a response body is not returned (for example, DELETE), providing the Accept header does not cause any unintended effects.

2.2 Responses

When a request is made to the API, a response returns two critical pieces of information:

- **HTTP status code**. Indicates whether the request was successful.
- **Optional response body**. Usually provides a JSON body representing the state of the resource or a body providing more details on the nature of a failure.

You must check the status code and the content-type header to determine what the resulting response body looks like. For HTTP status codes 200–203 and 422, Web Services returns a JSON body with the response. For other HTTP status codes, Web Services generally does not return an additional JSON body, either because the specification does not allow it (204) or because the status is self-explanatory.

Table 3 lists common HTTP status codes and definitions. It also indicates whether information associated with each HTTP code is returned in a JSON body.

Table 3) Common HTTP status codes and definitions.

HTTP Status Code	Description	JSON Body
200 OK	Denotes a successful response.	Yes

HTTP Status Code	Description	JSON Body
201 Created	Indicates that an object was created. This code is used in a few rare cases instead of a 200 status.	Yes
202 Accepted	Indicates that the request is accepted for processing as an asynchronous request, but you must make a subsequent request to get the actual result.	Yes
203 Non-Authoritative Information	Similar to a 200 response, but Web Services cannot guarantee that the data is up-to-date (for example, only cached data is available at this time).	Yes
204 No Content	Indicates a successful operation, but there is no response body. This code is always used for DELETE operations.	No
400 Bad Request	Indicates that the JSON body provided in the request is not valid.	No
401 Unauthorized	Indicates that an authentication failure has occurred. Either no credentials were provided, or the user name or password was invalid.	No
403 Forbidden	An authorization failure, which indicates that the authenticated user does not have permission to access the requested endpoint.	No
404 Not Found	Indicates that the requested resource could not be located. This code is valid for nonexistent APIs or nonexistent resources requested by the identifier.	No
422 Unprocessable Entity	Indicates that the request is generally well formed, but either the input parameters are invalid or the state of the storage system does not allow Web Services to satisfy the request.	Yes
424 Failed Dependency	Used in the SANtricity Web Services Proxy to indicate that the requested storage system is inaccessible. Therefore, Web Services cannot satisfy the request.	No
429 Too Many Requests	Indicates that a request limit was exceeded and should be retried later.	No

3 Web Services Implementation

The Web Services API is available in two implementations:

- **Embedded**. A RESTful API server is embedded on each controller in an E2800/EF280 storage system running NetApp SANtricity 11.30 or later versions and on an E5700/EF570/EF600 running SANtricity 11.40 or later versions. The API is accessible over the HTTPS protocol on port 8443 (default).
- **Proxy**. The SANtricity Web Services Proxy is a RESTful API server installed separately on a host machine. The API is accessible through either HTTP (8080) or HTTPS (8443) protocols, but the listening ports can be changed to available port numbers for your specific server during the proxy installation. This host-based application can manage hundreds of new and legacy NetApp E-Series storage systems. However, be aware that the proxy uses minimal security settings by default, as described in Table 4.

The core of the API is available in both implementations.

3.1 Comparison of Embedded and Proxy Implementations

In general, you should use the proxy for networks with more than 10 storage systems. The proxy can handle numerous requests more efficiently than the embedded API. For more information, see Table 4.

Table 4) Comparison of Web Services implementations.

Consideration	Proxy (Linux and Windows hosts)	Embedded (E2800 and E5700 series)
Installation	Requires a host system (Linux or Windows). The proxy is available for download at the NetApp Support site or on Docker Hub.	No installation or enablement required.
Security	Minimal security settings by default. Security settings are low so that developers can get started with the API quickly and easily. If desired, you can configure the proxy with the same security profile as the embedded version.	High security settings by default. Security settings are high because the API runs directly on the controllers. For example, it does not allow HTTP access, and it disables all SSL and older TLS encryption protocols for HTTPS.
Lightweight Directory Access Protocol (LDAP)	Available by file configuration.	Available through API access.
Central management	Capable of managing all supported systems from one server.	Requires communication with individual systems and configuration of each.
Mirroring	Support is included.	Not available.
Discovery	Discovery of newly deployed systems is supported.	N/A

3.2 Proxy Installation

If you plan to manage multiple systems and you have a mixture of old and new E-Series models, you can download the SANtricity Web Services Proxy, install it on a host system with IP management access to the arrays, and then manage both the legacy arrays and the new E2800/EF280, E5700/EF570, and EF600 array models. A generic installation package is available for Windows and Linux; you can download it from the NetApp Support site. Additionally, an RPM package is available for RPM-based platforms (Red Hat and SUSE).

When installing the proxy, make sure your system meets the requirements listed in Table 5.

Table 5) Proxy installation requirements.

Requirement	Instructions
Hostname limitations	Be sure that the host name of the server where you will install Web Services Proxy contains only ASCII letters, numerical digits, and hyphens (-). This requirement is due to a limitation of Java Keytool, which is used in generating a self-signed certificate for the server. If the hostname of your server contains any other characters, such as an underscore (_), the Web Server will fail to start after installation.

Requirement	Instructions
Compatibility and modes	You can install the proxy on the following operating systems: • Linux • Windows For a complete list of operating systems and firmware compatibility, see the NetApp Interoperability Matrix Tool.
Linux: Additional Considerations	Linux Standard Base (LSB): Make sure that the LSB package for your Linux distribution is installed. You need this package to properly start and shut down the Web Services Proxy.
Capacity planning	Web Services Proxy requires adequate space for logging. Make sure that your system meets the following available disk space requirements: • Required installation space – 275MB • Minimum logging space – 200MB • System memory – 2GB; heap space is 1GB by default You can use a disk-space monitoring tool to verify available disk space for persistent storage and logging.

The Installation Guide details the procedure for installing the files. The installation path defaults to:

- Windows – `C:\Program Files\NetApp\SANtricity Web Services Proxy`
- Linux – `/opt/netapp/santricity_web_services_proxy`

Under these installation paths, you can find the `wsconfig.xml` configuration file with various startup and runtime configuration options. The User Guide details the configuration options and procedures.

4 Interactive API Documentation

Web Services includes API documentation that allows you to directly interact with the API. This documentation runs with each instance of Web Services, and it is also available in a static PDF format from the NetApp Support site.

Note: The Web Services API implements the OpenAPI specification (originally called the Swagger specification).

To access the interactive API documentation, complete the following steps:

1. Open a browser.
2. Enter the URL for the embedded or proxy implementation:
   - **Embedded:** `https://<controller>:<port>/devmgr/docs/`

     In this URL, <`controller`> is the IP address or FQDN of the controller, and <`port`> is the management port number of the controller (defaults to 8443).
   - **Proxy:** `http[s]://<server>:<port>/devmgr/docs/`

     In this URL, <`server`> is the IP address or FQDN of the server where the proxy is installed, and <`port`> is the listening port number (defaults to 8080 for HTTP or 8443 for HTTPS).

The API documentation defaults to version 2, as shown in Figure 3.

Figure 3) Default API documentation landing page.

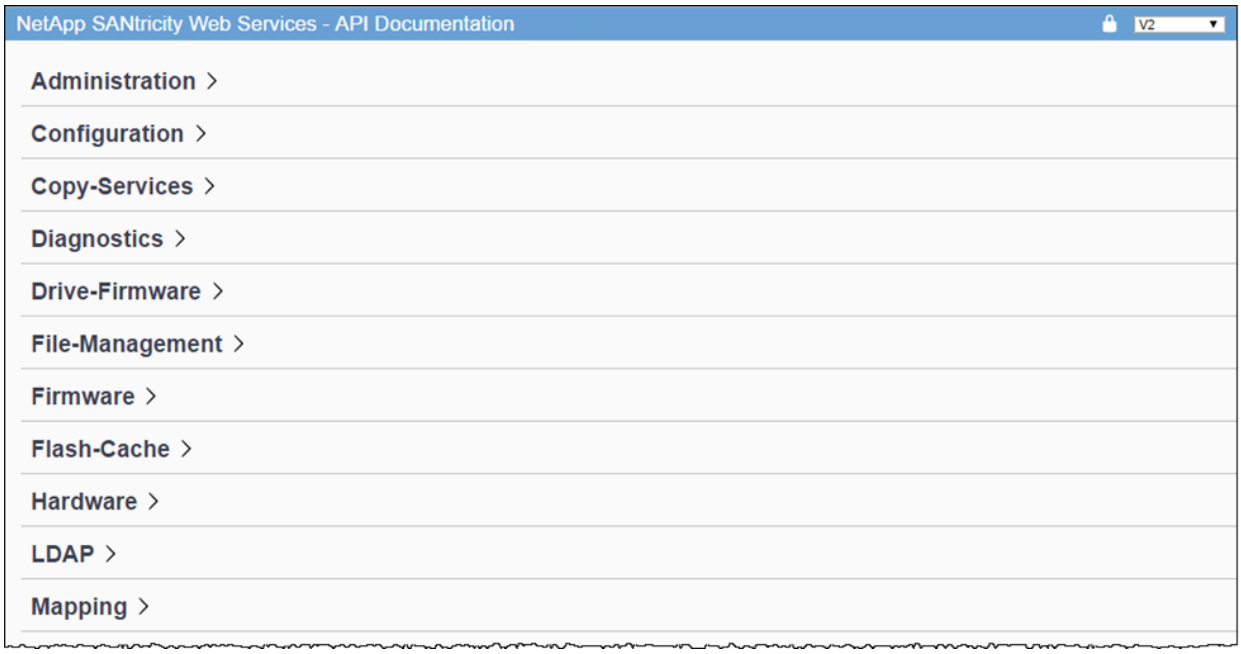
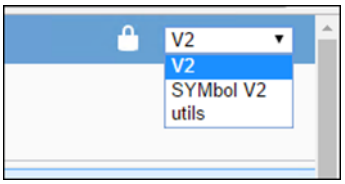
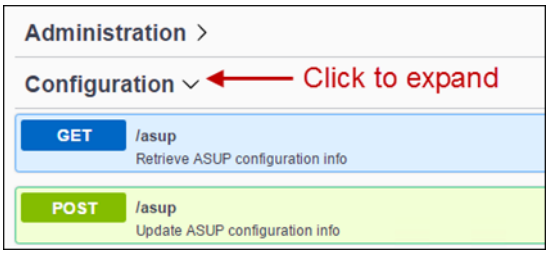
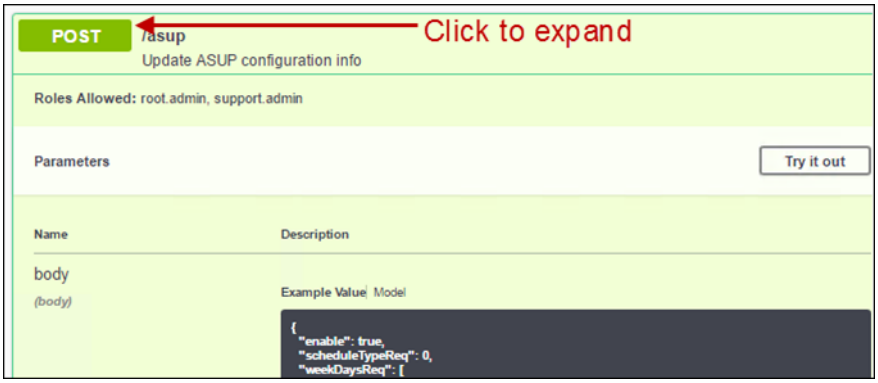
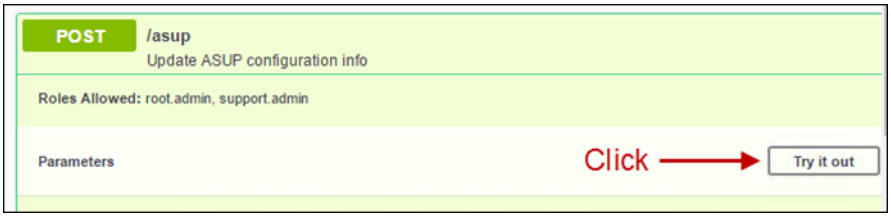
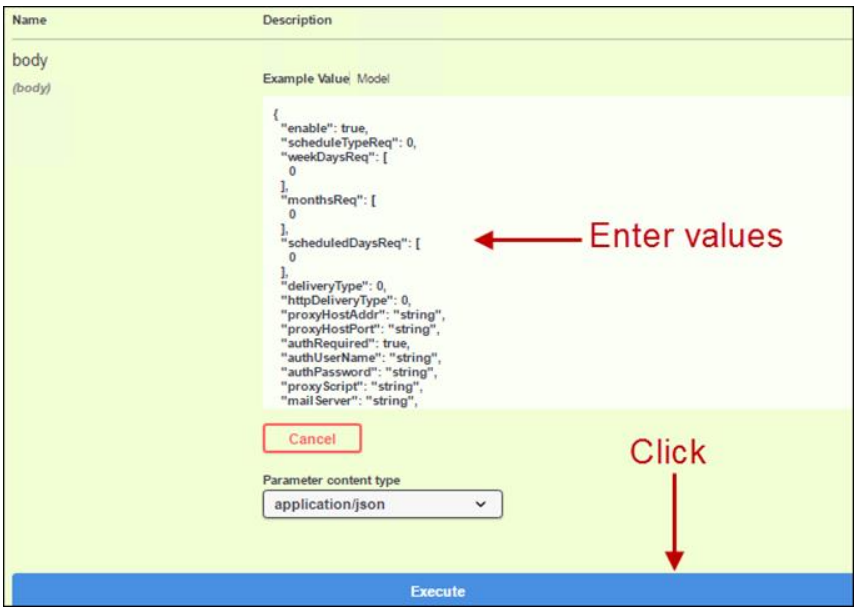


Table 6 describes the functions available from the interactive documentation.

Table 6) Interactive API documentation.

Area	Description
Drop-down menu	At the upper right of the page, a drop-down menu provides options for switching between version 2 of the API documentation (V2), the SYMbol interface (SYMbol V2), and API utilities (utils) for logging in.  **Note:** Because version 1 of the API documentation was a prerelease and not generally available, V1 is not included in the drop-down menu.
Categories	The API documentation is organized by high-level categories for the supported endpoints. Click a category to see the endpoints. 

Area	Description
Endpoints	Select an endpoint to see its URL paths, required parameters, response bodies, and status codes that the URLs are likely to return. 
Try It Out	Interact with the endpoint directly by clicking Try It Out. This button is provided in each of the expanded views for endpoints.  When you click the button, fields appear for entering parameters (if applicable). You can then enter values and click Execute. The interactive documentation uses JavaScript to make the request directly to the API; it is not a test request. 

For an example of how to use endpoints to complete a task, see section 6, “Sample Workflow: Discover and Add a Storage System.”

5 Access and Login

The method for accessing the Web Services API depends on whether you plan to use the embedded or proxy implementation. Both require authentication before you can log in, as described in this section.

5.1 Authentication

Several methods of authentication are available:

- **Login URL**. The default method is to use POST from the utils page in the interactive API documentation.
- **Basic authentication**. To enable basic authentication, you must add `<env key="enable-basic-auth">true</env>` to the `wsconfig.xml` file in the Environmental Entries section. See the User Guide for details.

Note: In the latest version of Web Services (3.0 and later), you can implement authorization through an LDAP server. See the User Guide and TR-4712: NetApp SANtricity Management Security for additional details.

5.2 User Logins and Role-Based Access

Access to Web Services requires a user name and password. The default credentials vary depending on which implementation you are accessing (embedded or proxy).

Note: You can change passwords in either implementation. Currently, however, you can add only user accounts in the proxy.

For the embedded implementation of Web Services, the administrator login depends on the NetApp SANtricity version of the particular array, as shown in Table 7.

Table 7) Embedded implementation: user names and passwords.

SANtricity Version	User Credentials
11.30 and earlier	User name: `rw` Password: [`administrator password`]
11.40 and later	User name: `admin@local` Password: [`administrator password`]

The SANtricity Web Services Proxy supports role-based user access starting with the version 3.0 release. Predefined users are associated with specific roles that enforce access controls. For a complete list of users and roles, see the User Guide.

The administrator logins are shown in Table 8.

The Web Services Proxy includes a graphical interface that is available from the root server landing page. Upon the first login, you will be prompted to change the password from the admin user, but this may be performed entirely via the API if desired.

Table 8) Proxy implementation: default user names and passwords.

Version	User Credentials
2.12 and earlier	User name: `rw` Password: `rw`
3.0 and later	User name: `admin` Password: `<empty>`

Login (Default Method)

To log in to the API, complete the following steps:

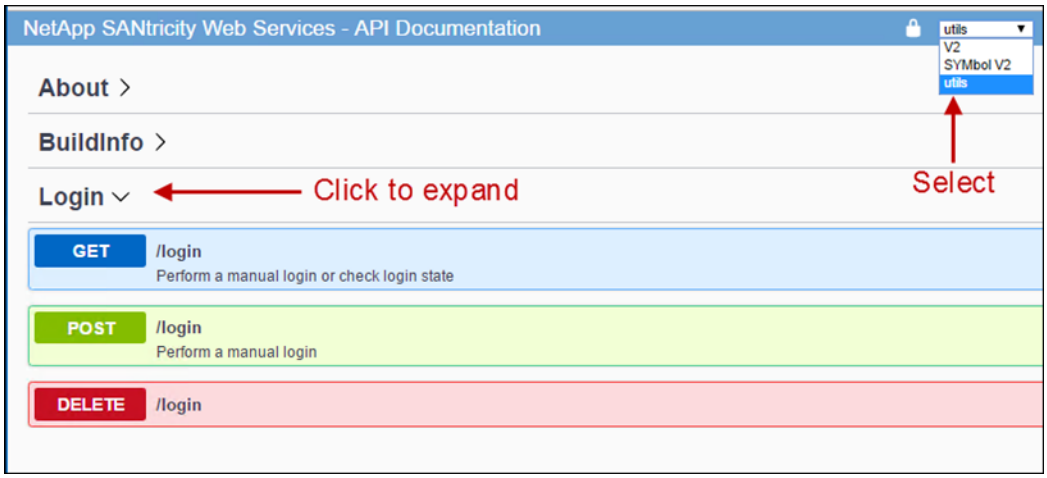
1. Enter the appropriate URL, depending on whether you want to access the embedded or proxy implementation:
   - **Embedded:** `https://<controller>:<port>/devmgr/docs/`

     In this URL, `<controller>` is the IP address or FQDN of the controller, and `<port>` is the management port number of the controller (defaults to 8443).
   - **Proxy:** `http[s]://<server>:<port>/devmgr/docs/`

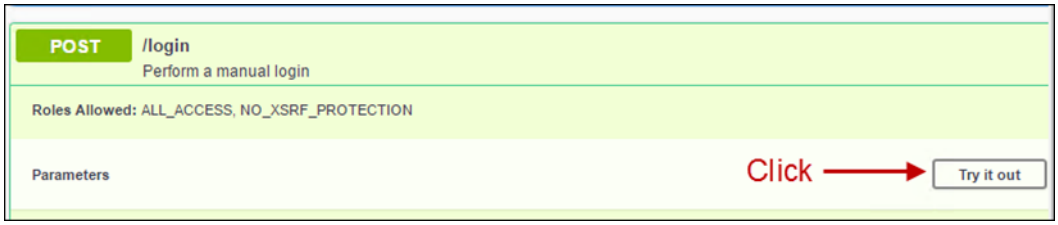
     In this URL, `<server>` is the IP address or FQDN of the server where the proxy is installed, and `<port>` is the listening port number (defaults to 8080 for HTTP or 8443 for HTTPS).

   Note: If the listening port is already in use, the proxy installer detects the conflict and prompts you to choose a different listening port.

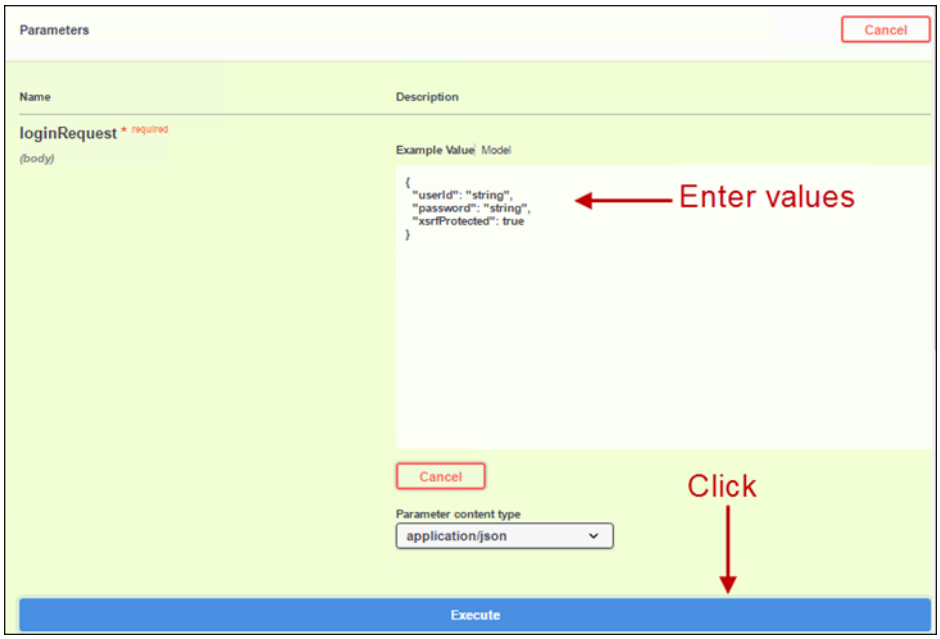
   After you enter the URL, the interactive API documentation opens.
2. From the drop-down menu at the upper right of the page, select utils.
3. Click the Login category to see the available endpoints.



4. Click the POST: /login endpoint, and then click Try It Out.



5. Enter the login parameters. For first-time setup, enter credentials for the admin user as described earlier in “User Logins and Role-Based Access.” The admin user is equivalent to a “super” user with access to all functions.



6. Click Execute.
7. Make sure that the code response is 200, indicating that the login is successful.

6 Sample Workflow: Discover and Add a Storage System

This section provides a sample workflow that uses the NetApp SANtricity Web Services Proxy to discover and add a storage system to the managed list. By discovering and adding a storage system, you create a management connection between the storage system and the API.

An overview of the required steps is provided in Table 9.

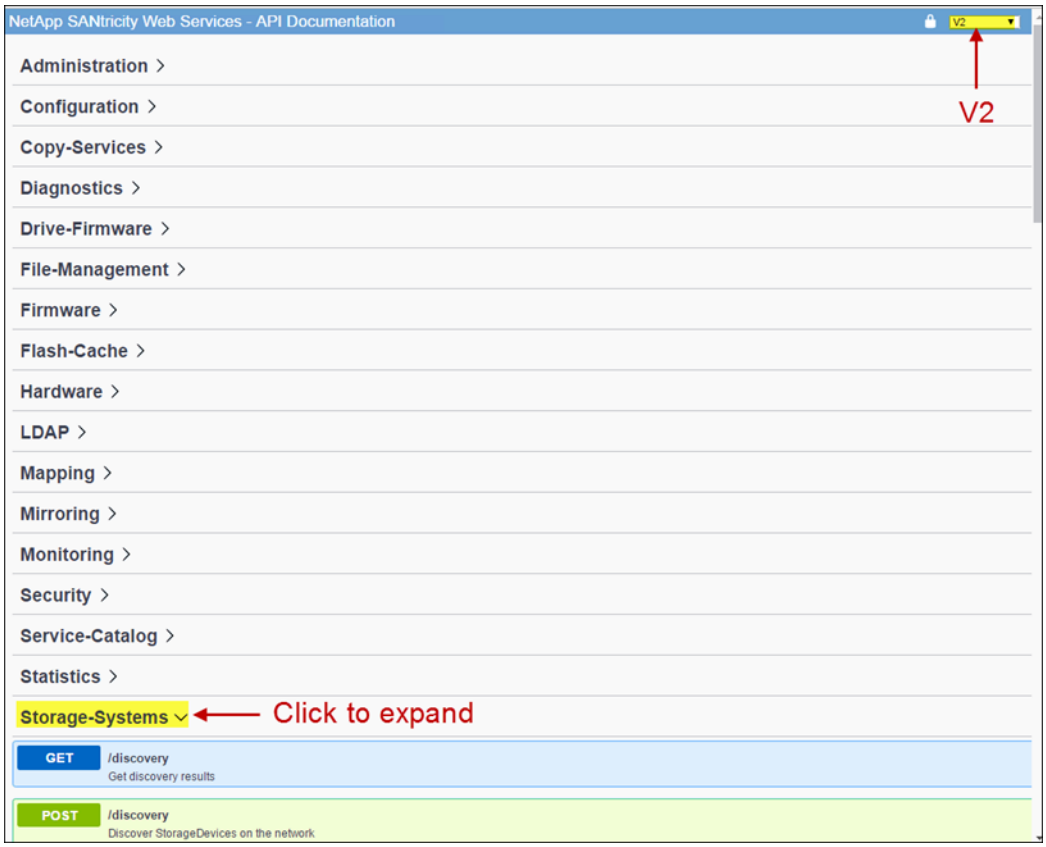
Table 9) Workflow for discovering and adding a storage system.

Step	Endpoint	Description
1. Discover storage system	POST: /discovery	Locate a storage system by using the IP address of the controller or by entering a range of addresses for multiple systems.
2. View discovery results	GET: /discovery	Display the storage system information by using the request ID from POST: /discovery.
3. Add storage system	POST: /storage-systems	Add the discovered storage system by using the WWN, label, and IP addresses from GET: /discovery.
4. Confirm list addition	GET: /storage-systems	Make sure that the discovered storage system has been added to the list of managed systems.

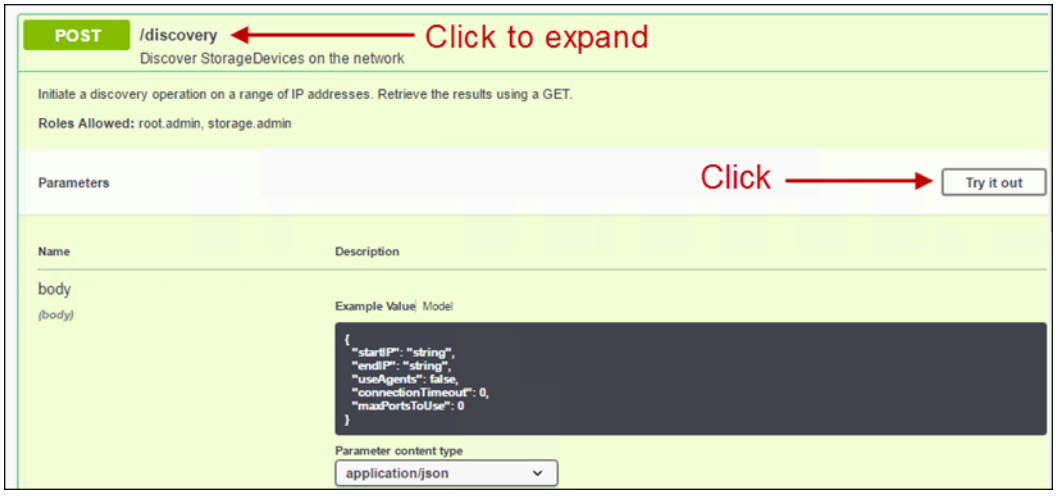
Note: For storage systems with SANtricity versions 11.30 and later, make sure that the legacy management interface for SYMbol is enabled in SANtricity System Manager. Otherwise, the discovery endpoints fail. This setting is located in Settings > System > Additional Settings > Change Management Interface.

Step 1: Discover the storage system

1. Log in to the API and access the interactive documentation, as described in section 5, “Access and Login.”
2. Make sure that V2 is selected in the drop-down, and then expand the Storage-Systems category.



3. Click the POST: /discovery endpoint, and then click Try It Out.



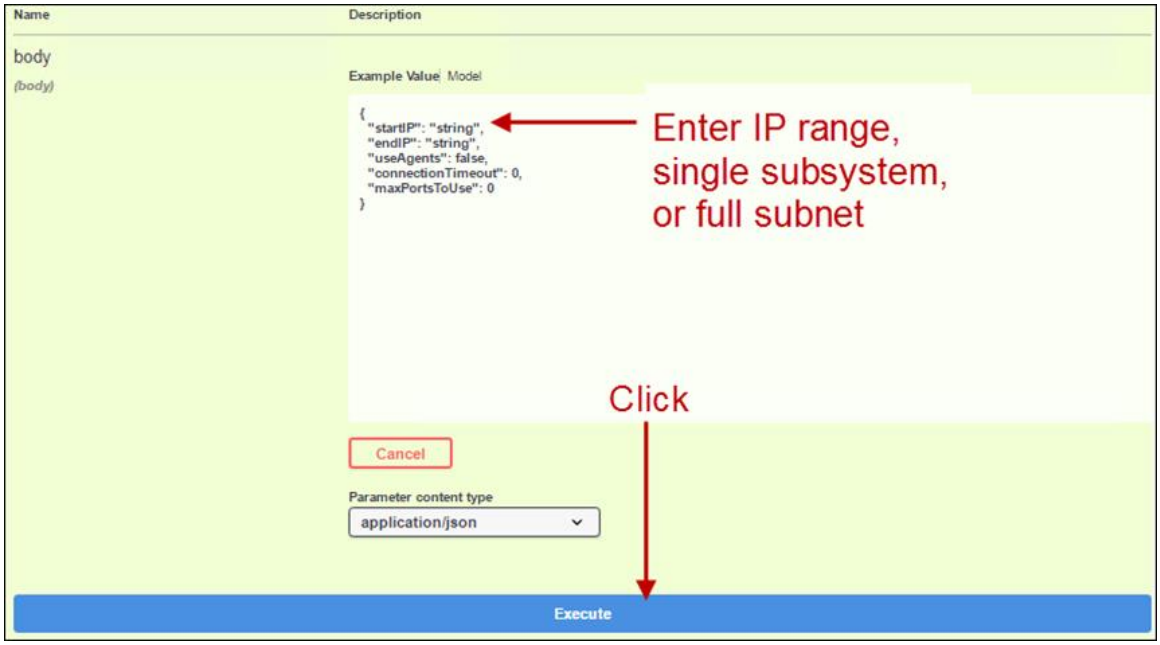
4. Enter the parameters as described in Table 10.

Table 10) POST: /discovery parameters.

Field	Instructions
startIP endIP	Replace `string` with the starting and ending IP address range for one or more storage systems in the network.
useAgents	Set this value to either: `true` = Use in-band agents for the network scan. `false` = Do not use in-band agents for the network scan.
connectionTimeout	Enter the seconds allowed for the scan before the connection times out.
maxPortsToUse	Enter a maximum number of ports used for the network scan.

5. Click Execute.

   Note: API actions execute without user prompts.

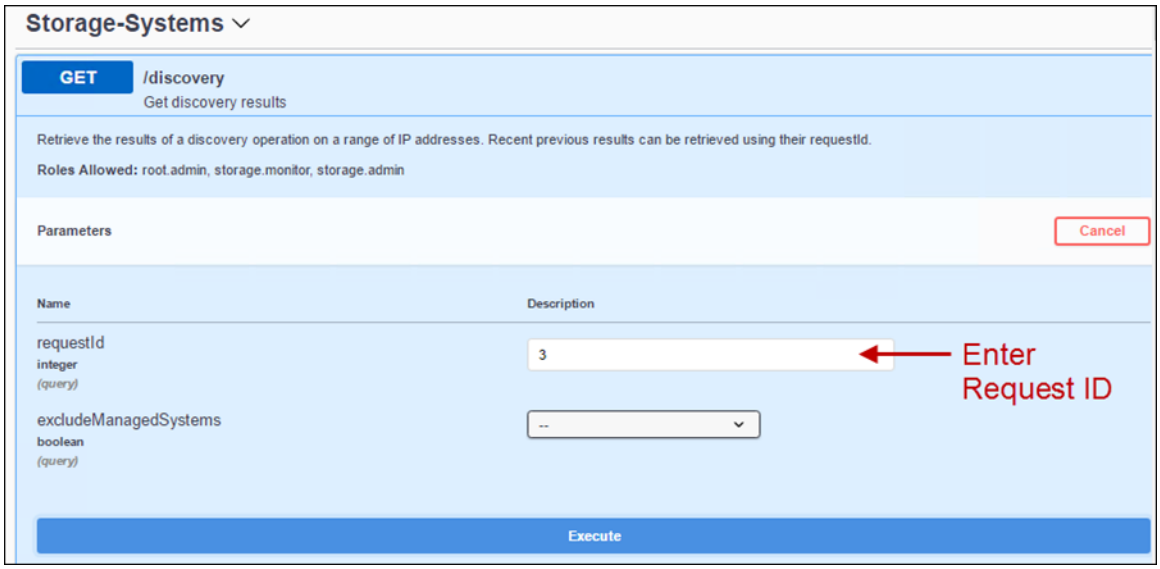


   The discovery process runs in the background.

6. Make sure that the code response is 202.
7. In the response body, locate the value returned for `requestId`. You need the request ID to view the results in the next step.

Step 2: View discovery results

1. Click the GET: /discovery endpoint, and then click Try It Out.
2. Enter the request ID from the previous step. If you leave the request ID blank, the endpoint defaults to the last request ID executed.



3. Click Execute.
4. Make sure that the code response is 200.
5. In the response body, locate your request ID and the strings for `storageSystems`. The strings look similar to the following example:

```
"storageSystems": [
    {
      "serialNumber": "123456789",
      "wwn": "000A011000AF0000000000001A0C000E",
      "label": "EF570_Array",
      "firmware": "08.41.10.01",
      "nvsram": "N5700-841834-001",
      "ipAddresses": [
        "10.xxx.xx.213",
        "10.xxx.xx.214"
      ],
```

You need the following values for the next endpoint:

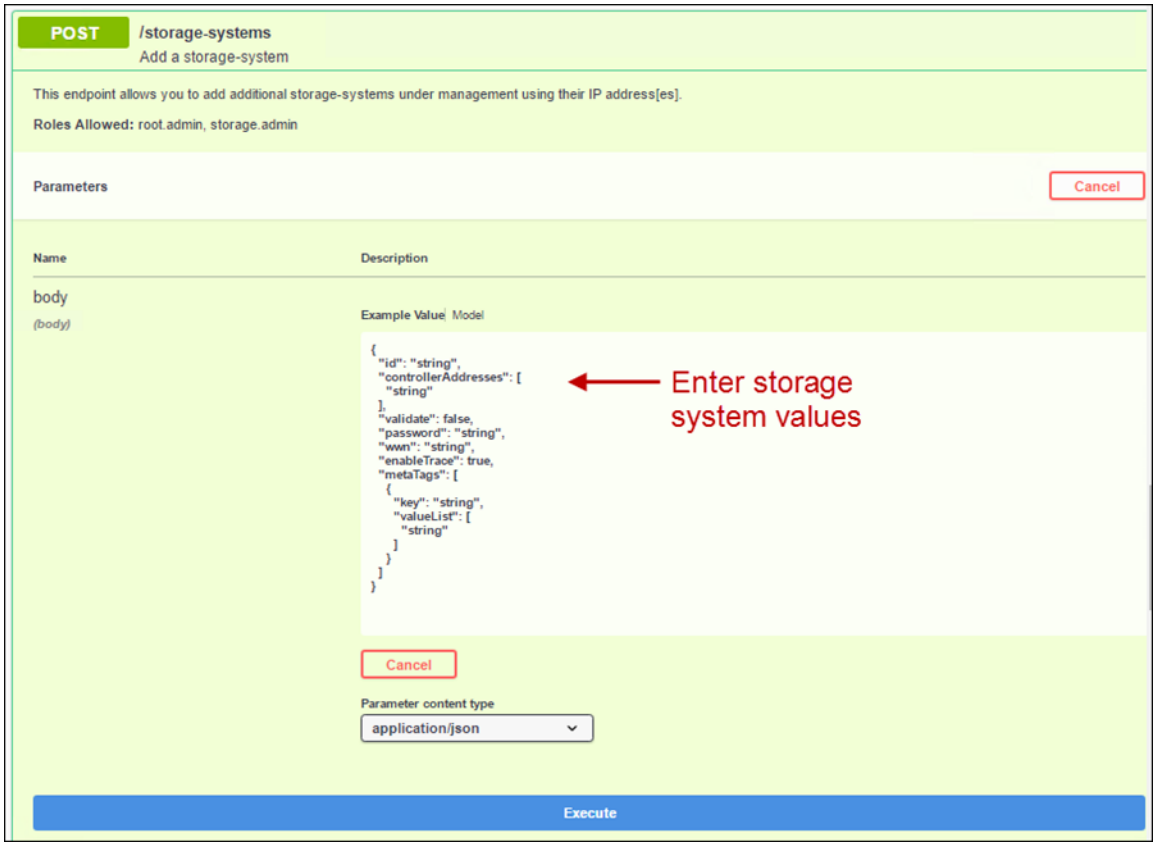
- `wwn` – The worldwide name, which is unique to each storage system.
- `label` – The name of the storage system that was entered during the array installation and base configuration process.
- `ipAddresses` – The IP addresses of the controller's management ports.

Step 3: Add the storage system

1. Click the POST: /storage-system endpoint, and then click Try It Out.
2. Enter the parameters as described in Table 11.

Table 11) POST: /storage-system parameters.

Field	Instructions
Id	Enter a unique name for this storage system. You can enter the label (displayed in the response for GET: /discovery), but the name can be any string you choose. If you do not provide a value for this field, Web Services automatically assigns a unique identifier.
controllerAddresses	Enter the IP addresses displayed in the response for GET: /discovery. For dual controllers, separate the IP addresses with a comma. For example: `"IP address 1","IP address 2"`
validate	Enter `true`, so you can receive confirmation that Web Services can connect to the storage system.
password	Enter the administrative password for the storage system.
wwn	Enter the WWN of the storage system (displayed in the response for GET: /discovery).



3. Remove all strings after `"enableTrace": true`, so that the entire string set includes the following:

```
{
  "id": "EF570_Array",
  "controllerAddresses": [
    "Controller-A-Mgmt-IP","Controller-B-Mgmt_IP"
  ],
  "validate":true,
```

```
  "password": "array-admin-password",
  "wwn": "000A011000AF0000000000001A0C000E",
  "enableTrace": true
}
```

4. Click Execute.
5. Make sure that the code response is 201, which indicates that the endpoint executed successfully.

   The Post: /storage-systems endpoint is queued. You can view the results by using the GET: /storage-systems endpoint in the next step.

Step 4: Confirm the list addition

1. Click the GET: /storage-system endpoint.
   No parameters are required.
2. Click Execute.
3. Make sure that the code response is 200, which indicates that the endpoint executed successfully.
4. In the response body, look for the storage system details. The returned values indicate that it was successfully added to the list of managed arrays, similar to the following example:

```
[
  {
    "id": "EF570_Array",
    "name": "EF570_Array",
    "wwn": "000A011000AF0000000000001A0C000E",
    "passwordStatus": "valid",
    "passwordSet": true,
    "status": "optimal",
    "ip1": "10.xxx.xx.213",
    "ip2": "10.xxx.xx.214",
    "managementPaths": [
      "10.xxx.xx.213",
      "10.xxx.xx.214"
  ]
  }
]
```

Where to Find Additional Information

The Web Services API provides customers with our full suite of management capabilities. This technical report is only a subset of our documentation; therefore, you should review additional sources of information before you get started with development.

To learn more about the information described in this technical report, see the following:

- NetApp SANtricity Web Services Proxy 4.2
- NetApp SANtricity Web Services Proxy 4.2 Documentation

Version History

Version	Date	Document Version History
Version 1.0	December 2017	Initial release
Version 2.0	December 2018	Updates for Web Services 3.0 release
Version 3.0	March 2019	Added system requirements for Web Services Proxy
Version 4.0	August 2020	Minor version updates

Refer to the Interoperability Matrix Tool (IMT) on the NetApp Support site to validate that the exact product and feature versions described in this document are supported for your specific environment. The NetApp IMT defines the product components and versions that can be used to construct configurations that are supported by NetApp. Specific results depend on each customer's installation in accordance with published specifications.

Copyright Information

Trademark Information

TR-4736-0820