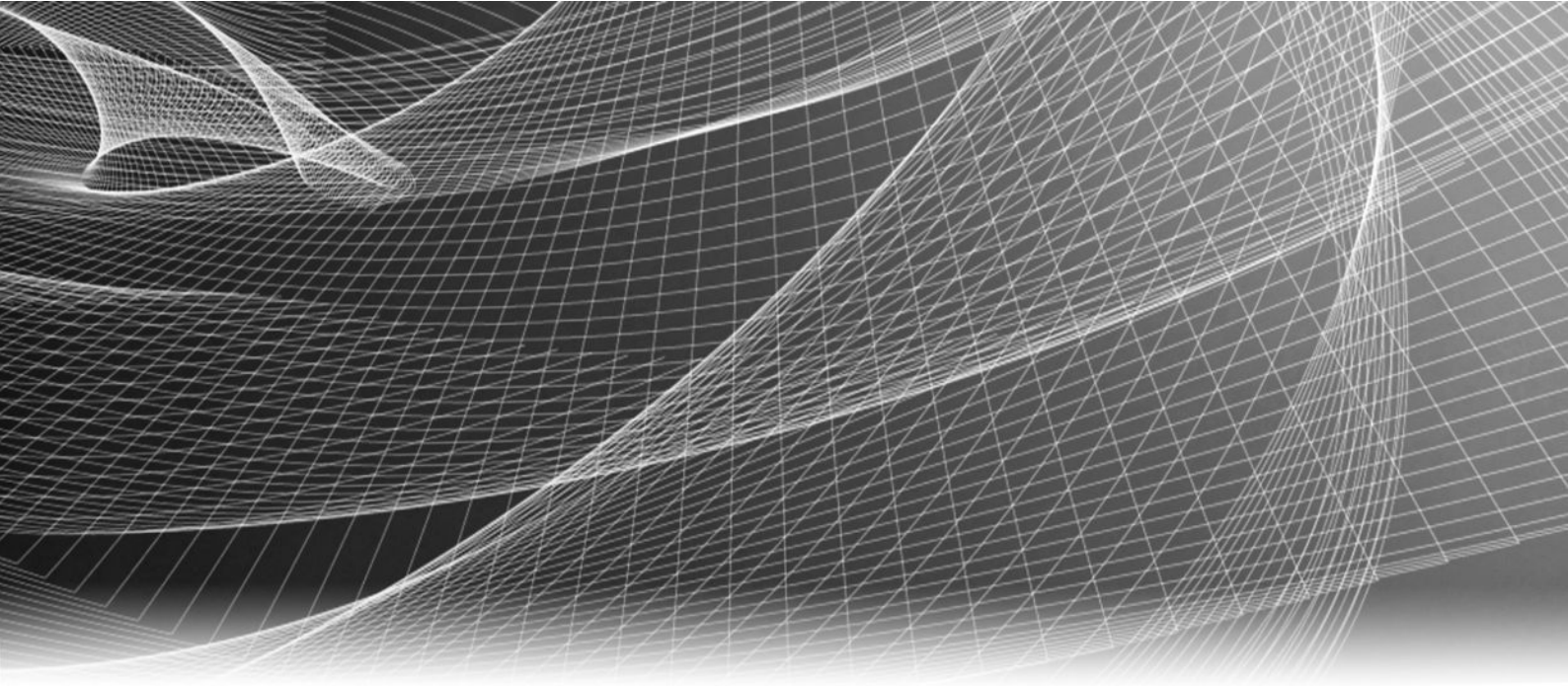

EMC® NetWorker®

Version 9.0.1

REST API Getting Started Guide

302-003-025

REV 02

EMC²

Published June 2016

EMC Corporation
Hopkinton, Massachusetts 01748-9103
1-508-435-1000 In North America 1-866-464-7381
www.EMC.com

CONTENTS

TABLES

Preface

As part of an effort to improve its product lines, EMC periodically releases revisions of its software and hardware. Therefore, some functions that are described in this document might not be supported by all versions of the software or hardware currently in use. The product release notes provide the most up-to-date information on product features.

Contact your EMC technical support professional if a product does not function correctly or does not function as described in this document.

Note

This document was accurate at publication time. Go to EMC Online Support (https://support.emc.com) to ensure that you are using the latest version of this document.

Purpose

This document describes how to configure and use EMC NetWorker REST API.

Audience

This guide is part of the NetWorker documentation set, and is intended for use by developers who are creating programmatic interfaces to NetWorker systems.

Revision history

The following table presents the revision history of this document.

Table 1 Revision history

Revision	Date	Description
01	June 27, 2016	Initial release of this document.
02	September 19, 2016	Removed a repeated paragraph.

Related documentation

The NetWorker documentation set includes the following publications, available on EMC Online Support:

- *EMC NetWorker Online Software Compatibility Guide*
  Provides a list of client, server, and storage node operating systems supported by the EMC information protection software versions. You can access the guide at https://support.emc.com. From the Support by Product pages, search for NetWorker using "Find a Product", and then select the Install, License, and Configure link.
- *EMC NetWorker Administration Guide*
  Describes how to configure and maintain the NetWorker software.
- *EMC NetWorker Network Data Management Protocol (NDMP) User Guide*
  Describes how to use the NetWorker software to provide data protection for NDMP filers.
- *EMC NetWorker Cluster Integration Guide*
  Contains information related to configuring NetWorker software on cluster servers and clients.
- *EMC NetWorker Installation Guide*

Provides information on how to install, uninstall, and update the NetWorker software for clients, storage nodes, and servers on all supported operating systems.

- *EMC NetWorker Updating from a Previous Release Guide*
  Describes how to update the NetWorker software from a previously installed release.
- *EMC NetWorker Release Notes*
  Contains information on new features and changes, fixed problems, known limitations, environment and system requirements for the latest NetWorker software release.
- *EMC NetWorker Command Reference Guide*
  Provides reference information for NetWorker commands and options.
- *EMC NetWorker Data Domain Boost Integration Guide*
  Provides planning and configuration information on the use of Data Domain devices for data deduplication backup and storage in a NetWorker environment.
- *EMC NetWorker Performance Optimization Planning Guide*
  Contains basic performance tuning information for NetWorker.
- *EMC NetWorker Server Disaster Recovery and Availability Best Practices Guide*
  Describes how to design and plan for a NetWorker disaster recovery. However, it does not provide detailed disaster recovery instructions. The Disaster Recovery section of the NetWorker Procedure Generator (NPG) provides step-by-step disaster recovery instructions.
- *EMC NetWorker Snapshot Management Integration Guide*
  Describes the ability to catalog and manage snapshot copies of production data that are created by using mirror technologies on EMC storage arrays.
- *EMC NetWorker Snapshot Management for NAS Devices Integration Guide*
  Describes how to catalog and manage snapshot copies of production data that are created by using replication technologies on NAS devices.
- *EMC NetWorker Security Configuration Guide*
  Provides an overview of security configuration settings available in NetWorker, secure deployment, and physical security controls needed to ensure the secure operation of the product.
- *EMC NetWorker VMware Integration Guide*
  Provides planning and configuration information on the use of VMware in a NetWorker environment.
- *EMC NetWorker Error Message Guide*
  Provides information on common NetWorker error messages.
- *EMC NetWorker Licensing Guide*
  Provides information about licensing NetWorker products and features.
- *EMC NetWorker REST API Getting Started Guide*
  Describes how to configure and use the NetWorker REST API to create programmatic interfaces to the NetWorker server.
- *EMC NetWorker REST API Reference Guide*
  Provides the NetWorker REST API specification used to create programmatic interfaces to the NetWorker server.
- EMC NetWorker Management Console Online Help
  Describes the day-to-day administration tasks performed in the NetWorker Management Console and the NetWorker Administration window. To view the online help, click **Help** in the main menu.
- EMC NetWorker User Online Help

Describes how to use the NetWorker User program, which is the Windows client interface, to connect to a NetWorker server to back up, recover, archive, and retrieve files over a network.

Special notice conventions that are used in this document

EMC uses the following conventions for special notices:

NOTICE

Identifies content that warns of potential business or data loss.

Note

Contains information that is incidental, but not essential, to the topic.

Typographical conventions

EMC uses the following type style conventions in this document:

Table 2 Style conventions

Bold	Used for names of interface elements, such as names of buttons, fields, tab names, and menu paths (what the user specifically selects or clicks)
Italic	Used for full titles of publications that are referenced in text
`Monospace`	Used for: • System code • System output, such as an error message or script • Pathnames, file names, prompts, and syntax • Commands and options
Monospace italic	Used for variables
`Monospace bold`	Used for user input
[ ]	Square brackets enclose optional values
|	Vertical bar indicates alternate selections - the bar means “or”
{ }	Braces enclose content that the user must specify, such as x or y or z
...	Ellipses indicate non-essential information that is omitted from the example

Where to get help

EMC support, product, and licensing information can be obtained as follows:

Product information

For documentation, release notes, software updates, or information about EMC products, go to EMC Online Support at https://support.emc.com.

Technical support

Go to EMC Online Support and click Service Center. Several options for contacting EMC Technical Support appear on the site. Note that to open a service request, you must have a valid support agreement. Contact your EMC sales representative for details about obtaining a valid support agreement or with questions about your account.

Online communities

Go to EMC Community Network at https://community.emc.com for peer contacts, conversations, and content on product support and solutions. Interactively engage online with customers, partners, and certified professionals for all EMC products.

Your comments

Your suggestions help to improve the accuracy, organization, and overall quality of the user publications. Send your opinions of this document to DPAD.Doc.Feedback@emc.com.

CHAPTER 1

Introduction

This chapter includes the following topics:

Overview of the NetWorker REST API

The NetWorker REST API is an interface that provides programmatic access to the NetWorker data protection service. By using this REST API, customers can build client applications to automate NetWorker operations. This document describes how to access NetWorker data protection Service using NetWorker REST API and provides example tutorials for common tasks.

NetWorker REST API service

The NetWorker REST API uses a service to facilitate operations. The NetWorker REST API service is deployed in the same Apache Tomcat container as NetWorker authentication service. The NetWorker REST API uses the same set of tomcat processes to deliver its service.

Installation

The NetWorker REST API is installed as part of NetWorker installation.

As part of the installation process, an Apache Tomcat instance is installed and a non-root -user, nsrtomcat, is created. If your system has special user security requirements, ensure that proper operational permissions are granted to the nsrtomcat users.

The *EMC NetWorker Installation Guide* contains further information.

Getting started with the NetWorker REST API

This document provides example tutorials about how to perform some of the common tasks that programmers working with the NetWorker software will want to accomplish. After working through these tutorials, the use of the API should be familiar enough to be able to develop interfaces that make use of the full range of available functionality.

NetWorker REST API specification

The complete specification for the NetWorker REST API is available in PDF format in the *EMC NetWorker REST API Reference Guide*.

Using the example tutorials in this guide

The example tutorials provided in this guide follow a similar pattern, first presenting an operation that achieves a particular objective (for example, creating a client) in the form of a NetWorker command line operation that can be run locally. Next, the same objective is presented in the form of a NetWorker REST API call. Users familiar with the NetWorker command line will recognize the command line form of these operations, and for this reason it is assumed that users of these example tutorials are familiar with working with the Users familiar with the NetWorker server and command line.

This document uses an example NetWorker server installed on a server named mars, and NetWorker REST API is exposed in the following URL: https://mars:9090/nwrestapi/.

When following these tutorials, you can use a curl client, or any other HTTP client. Typical curl syntax to call the API will look as follows:

```
curl -X POST -H "Content-Type: application/json" -H "Authorization:
Basic YWRtaW5pc3RyYXRvcjohNFUyYnVpbGQ=" -d '{ "hostname" : "saturn"
}' "https://mars:9090/nwrestapi/v1/global/clients" -k -1
```

The following table examines the parameters being passed into the curl command in the above example.

Parameter	Value
-X POST	This specifies HTTP method that is being used. Generally: • The GET method is used to retrieve data. • The PUT method is used to update data. • The POST method is used to create. • The DELETE method is used to remove.
-H "Content-Type: application/json"	Specifies the type of payload we being sent to the API. The NetWorker REST API uses json exclusively, so this never changes.
-H "Authorization: Basic YWRtaW5pc3RyYXRvcjohNFUyYnVpbGQ="	Specifies an authorization header that contains Base64 encoded username and password.
-d '{ "hostname" : "saturn" }'	This parameter represents to actual payload being sent to the API.
https://mars:9090/nwrestapi/v1/global/clients	The URL of the API endpoint to which the request is being sent.
-k -1	This option is environment specific, and specifies version of the SSL protocol to be used for communication with the API.

To make the examples more readable, examples in this document simplify the API request into tool independent form, where only the data that actually changes between requests is specified (generally the HTTP method, endpoint URL, and the actual payload). For the endpoint URL, the hostname in the URL will also be ignored (for example, https://mars:9090).

So the example will in fact be denoted as:

```
POST /global/clients{ "hostname" : "saturn" }
```

Note that the payload is only required for PUT and POST requests.

CHAPTER 2

NetWorker REST API example usage

This chapter contains the following topics:

Creating a client

This section demonstrates how to create a client by using the NetWorker REST API.

Command line example

The following example command uses the `nsradmin` program to create a client:

```
nsradmin> create type: NSR client; name: saturn; save set: /etc/hosts
                          type: NSR client;
                          name: saturn;
                      save set: /etc/hosts;
Create? y created resource id
46.0.236.47.0.0.0.0.71.53.50.87.10.5.167.140(1)
```

API request

The same client is created in the following example by using the NetWorker REST API:

```
POST /nwrestapi/v1/global/clients
{
    "hostname" : "saturn",
    "saveSets" : ["/etc/hosts"]
}
```

API response

The following response is received from the API:

```
201 Created
```

Viewing all clients

This section demonstrates how to view all clients configured on a NetWorker server using the NetWorker REST API.

Command line example

The following example command uses the `nsradmin` program to list all client configured on the NetWorker server, and displays the returned data (in this case for one client, with the hostname "saturn"):

```
nsradmin> p type: NSR client
...
                          type: NSR client;
                          name: saturn;
                       comment: ;
                     directive: ;
       protection group list: ;
                     save set: /etc/hosts;
          Checkpoint enabled: Disabled;
Parallel save streams per save set: Disabled;
                 remote access: ;
                   remote user: ;
                      password: ;
         NAS management user: ;
     NAS management password: ;
        NAS file access user: ;
    NAS file access password: ;
        index backup content: No;
              backup command: ;
                 Pre command: ;
                Post command: ;
```

```
     application information: ;
     ndmp vendor information: ;
                        ndmp: No;
  ndmp multi-streams enabled: No;
                Disable IPv6: No;
                  NAS device: No;
             NDMP array name: ;
  NAS device management name: ;
storage replication policy name: ;
         Probe resource name: ;
          Block based backup: No;
             executable path: ;
                     aliases: saturn, saturn.lss.emc.com;
                 parallelism: 12;
                 backup type: ;
                         tag: ;
        restricted data zone: ;
```

API request

The following command uses the NetWorker REST API to list all configured clients:

```
GET nwrestapi/v1/global/clients
```

API response

The following response is received from the API:

```
200 OK
{
  "clients": [
   ...
    {
      "aliases": [
        "saturn",
        "saturn.lss.emc.com"
      ],
      "applicationInformation": [],
      "blockBasedBackup": false,
      "checkpointEnabled": false,
      "clientId":
"5c088865-00000004-57339887-57339886-00010c00-80c64a29",
      "hostname": "saturn",
      "indexBackupContent": false,
      "links": [
        {
          "href": "https://mars:9090/nwrestapi/v1/global/clients/
48.0.254.77.0.0.0.0.149.148.51.87.128.222.109.201",
          "rel": "item"
        }
      ],
      "nasDevice": false,
      "ndmp": false,
      "ndmpMultiStreamsEnabled": false,
      "ndmpVendorInformation": [],
      "parallelSaveStreamsPerSaveSet": false,
      "parallelism": 4,
      "protectionGroups": [],
      "remoteAccessUsers": [],
      "resourceId": {
        "id": "48.0.254.77.0.0.0.0.149.148.51.87.128.222.109.201",
        "sequence": 2
      },
      "saveSets": [
        "/etc/hosts"
      ],
      "scheduledBackup": true,
```

```
      "tags": []
    }
  ],
  "count": 2
}
```

Viewing only selected fields for all clients

Because the example above listed all fields for all clients, the returned JSON objects will sometimes be very large. You can limit the number of fields that are returned by the NetWorker REST API, as follows.

Command line example

The following example command uses the `nsradmin` program to view all clients configured on a NetWorker, but restricting the returned data to only the client name and aliases (two clients are found in this example, with the hostnames "saturn" and "jupiter"):

```
nsradmin> show aliases; name
nsradmin> p type: NSR client
                        name: saturn;
                     aliases: saturn, saturn.lss.emc.com;

                        name: jupiter;
                     aliases: jupiter;
```

API request

The following command performs the same functionality using the NetWorker REST API:

```
GET nwrestapi/v1/global/clients?fl=aliases,hostname
```

API response

The following response is received from the API:

```
200 OK
{
  "clients": [
    {
      "aliases": [
        "jupiter"
      ],
      "hostname": "jupiter"
    },
    {
      "aliases": [
        "saturn",
        "saturn.lss.emc.com"
      ],
      "hostname": "saturn"
    }
  ],
  "count": 2
}
```

Searching for clients

You can also restrict the number of clients that are found by using search criteria. In this case example, only one client with the hostname of "saturn" will be returned.

Command line example

The following example command uses the `nsradmin` program to search for a client:

```
nsradmin> . type: NSR client; name: saturn
Current query set
nsradmin>            name: saturn;
                     aliases: saturn, saturn.lss.emc.com;
```

API request

The same search is performed in the following example by using the NetWorker REST API:

```
GET /nwrestapi/v1/global/clients?fl=aliases,hostname&q=hostname:saturn
```

API response

The following response is received from the API:

```
200 OK
{
  "clients": [
    {
      "aliases": [
        "saturn",
        "saturn.lss.emc.com"
      ],
      "hostname": "saturn"
    }
  ],
  "count": 1
}
```

Creating an Advanced File Type Device

This section demonstrates how to create an Advanced File Type Device (AFTD) using the NetWorker REST API.

Command line example

The following example command uses the `nsradmin` program to create an AFTD:

```
nsradmin> create type: NSR device; name: /space/storage; device
access information: /space/storage; media type: adv_file
                          type: NSR device;
                          name: /space/storage;
    device access information: /space/storage;
                       media type: adv_file;
Create? y created resource id
47.0.236.47.0.0.0.0.71.53.50.87.10.5.167.140(1) nsradmin> q
```

API request

The same device is created in the following example by using the NetWorker REST API:

```
POST /nwrestapi/v1/global/devices
{
    "name": "/space/storage",
```

```
    "deviceAccessInfo": "/space/storage",
    "mediaType": "adv_file"
}
```

API response
The following response is received from the API:

```
201 Created
```

Labeling a device

This section demonstrates how to label a device using the NetWorker REST API.

Command line example
The following example command uses the `nsrmm` command to label a device:

```
# nsrmm -l -m -y -b Default -f /space/storage
Using volume name `mars.001' for pool `Default'
```

API request
The device is labeled in the same mannger in the following example by using the NetWorker REST API:

```
POST /nwrestapi/v1/global/devices
{
    "name": "/space/storage",
    "deviceAccessInfo": "/space/storage",
    "mediaType": "adv_file"
}
```

API response
The following response is received from the API:

```
202 Accepted
```

Creating a protection group

This section demonstrates how to create a protection group by using the NetWorker REST API.

Command line example
The following example command uses the `nsrpolicy` command to create a protection group :

```
# nsrpolicy group create client -g EngineeringWorkstations -C saturn
nsrpolicy: Group 'EngineeringWorkstations' was successfully created
```

API request
The same protection group is created in the following example by using the NetWorker REST API:

```
POST /nwrestapi/v1/global/protectiongroups
{
    "workItemType" : "Client",
    "name" : "EngineeringWorkstations",
    "workItems" :
```

```
["48.0.254.77.0.0.0.0.149.148.51.87.128.222.109.201"]
}
```

Note

In the command line example, the name of the client is used. However when using the REST API, the resourceId of the client must be used instead, as show in this example in the `workItems` element. This can be retrieved from the resourceId element in the client resource, as shown in the API response section of Viewing all clients on page 16.

API response
The following response is received from the API:

```
201 Created
```

Creating a protection policy

This section demonstrates how to create a protection policy by using the NetWorker REST API.

Command line example
The following example command uses the `nsrpolicy` command to create a protection policy:

```
# nsrpolicy policy create -p Engineering
nsrpolicy: Policy 'Engineering' was successfully created
```

API request
The same protection policy is created in the following example by using the NetWorker REST API:

```
POST /nwrestapi/v1/global/protectionpolicies
{
    "name": "Engineering"
}
```

API response
The following response is received from the API:

```
201 Created
```

Creating a workflow

This section demonstrates how to create a workflow by using the NetWorker REST API.

Command line example
The following example command uses the `nsrpolicy` command to create a workflow:

```
# nsrpolicy workflow create -p Engineering -w ClientProtection -g
EngineeringWorkstations -S 21:00
nsrpolicy: workflow 'ClientProtection' was successfully created
```

API request

The same workflow is created in the following example by using the NetWorker REST API:

```
POST /nwrestapi/v1/global/protectionpolicies/Engineering/workflows
{
    "name" : "ClientProtection",
    "protectionGroups" : ["EngineeringWorkstations"],
    "startTime" : "21:00"
}
```

API response

The following response is received from the API:

```
201 Created
```

Creating actions for a traditional backup

This section demonstrates how to create actions for a traditional backup by using the NetWorker REST API.

Command line example

The following example command uses the `nsrpolicy` command to create a actions for a traditional backup:

```
# nsrpolicy action create backup traditional -p Engineering -w
ClientProtection -A Backup
nsrpolicy: Assigned default schedule period of 'week' and schedule
activity of 'full, incr, incr, incr, incr, incr, incr'.
nsrpolicy: Action 'Backup' was given '1 Months' retention period
nsrpolicy: Action 'Backup' was given 'nsrserverhost' storage node
nsrpolicy: Action 'Backup' was given 'Default' destination pool
nsrpolicy: Action 'Backup' was successfully created
```

API request

The same actions are created in the following example by using the NetWorker REST API:

```
PUT /nwrestapi/v1/global/protectionpolicies/Engineering/workflows/
ClientProtection
{
  "actions":[
  {
    "name":"Backup",
    "actionSpecificData":{
      "backup":{
         "backupSpecificData":{
            "traditional":{}
         }
      }
    }
 }
 ]
}
```

API response

The following response is received from the API:

```
204 No Content
```

Performing an on-demand backup

This section demonstrates how to perform an on-demand backup of a client by using the NetWorker REST API.

Command line example

The following example commands uses the `nsrpolicy` command or the `nsrworkflow` command to perform an on-demand backup of a client:

```
# nsrpolicy start -p Engineering -w ClientProtection -c saturn:/etc/
hosts
144091:nsrpolicy: Workflow 'Engineering/ClientProtection' started and
has job id 32023
```

or

```
# nsrworkflow -p Engineering -w ClientProtection -c saturn:/etc/hosts
133550:nsrworkflow: Starting Protection Policy 'Engineering' workflow
'ClientProtection'.
123316:nsrworkflow: Starting action 'Engineering/ClientProtection/
Backup' with command: 'savegrp -Z backup:traditional -v'.
123321:nsrworkflow: Action 'Engineering/ClientProtection/Backup's log
will be in '/nsr/logs/policy/Engineering/ClientProtection/
Backup_032028.raw'.
123325:nsrworkflow: Action 'Engineering/ClientProtection/Backup'
succeeded.
133553:nsrworkflow: Workflow 'Engineering/ClientProtection' succeeded.
```

API request

The same on-dmand backup is performed in the following example by using the NetWorker REST API:

```
API request:
POST /nwrestapi/v1/global/protectionpolicies/Engineering/workflows/
ClientProtection/op/backup
{
    "clients": ["saturn:/etc/hosts"]
}
```

API response

The following response is received from the API:

```
201 Created
```

Retriving information about the backup job

The location header in the response from the above API will contain a URL to the job that was created. Use the following GET request to that URL to retrieve information about the job:

```
GET /nwrestapi/v1/global/jobs/64081
```

The following information is returned:

```
{
   "adhocJob": false,
   "command": "/usr/sbin/nsrworkflow -p Engineering -w
ClientProtection -L -c saturn:/etc/hosts",
   "dependentJobIds": [
     0
```

```
   ],
   "id": 64081,
   "itemIdLong": 64081,
   "links": [
     {
       "href": "https://mars:9090/nwrestapi/v1/global/jobs/64081/op/
cancel",
       "title": "Cancel job"
     }
   ],
   "logFile": "/nsr/logs/policy/Engineering/
workflow_ClientProtection_064081.raw",
   "name": "Engineering",
   "ndmp": false,
   "parentJobId": 0,
   "previousJobId": 0,
   "rootParentJobId": 0,
   "runOnHost": "mars",
   "siblingJobIds": [],
   "startTime": "2016-05-17T12:34:07-04:00",
   "state": "Active",
   "stopped": false,
   "tenant": "",
   "type": "workflow job"
}
```

Viewing all backups for a client

This section demonstrates how to view all backups for a client by using the NetWorker REST API.

Command line example

The following example command uses the `mminfo` command to view the backups for a client:

```
# mminfo -avot -q "client=saturn"
 volume    type   client    date     time      size ssid      fl
lvl name
mars.001   adv_file mars     05/17/16 11:41:38 6 KB 4030413746 cb
manual /etc/hosts
...
```

API request

The same request is created in the following example by using the NetWorker REST API:

```
GET /nwrestapi/v1/global/clients/
82.0.31.22.0.0.0.0.33.12.41.87.128.222.109.201/backups
```

Note

In the command line example, the name of the client is used. However when using the REST API, the resourceId of the client must be used instead, as show in this example URL. This can be retrieved from the resourceId element in the client resource, as shown in the API response section of Viewing all clients on page 16.

API response

The following response is received from the API:

```
{
  "backups": [
```

```
    {
      "attributes": [
        {
          "key": "*backup start time",
          "values": [
            "1463499698"
          ]
        },
        {
          "key": "*ss clone retention",
          "values": [
            "          1463499698:          1463499698:        -2"
          ]
        },
        {
          "key": "saveset features",
          "values": [
            "CLIENT_SAVETIME"
          ]
        }
      ],
      "browseTime": "2038-01-18T22:14:06-05:00",
      "clientHostname": "mars",
      "clientId":
"b7d73be0-00000004-57290c2f-57290c2e-00010c00-80c64a29",
      "completionTime": "2016-05-17T11:41:38-04:00",
      "creationTime": "2016-05-17T11:41:38-04:00",
      "fileCount": 3,
      "id": "67f3852c-00000006-f03b3bb2-573b3bb2-00120c00-80c64a29",
      "instances": [
        {
          "clone": false,
          "id": "1463499698",
          "status": "Browsable",
          "volumeIds": [
            "4264957582"
          ]
        }
      ],
      "level": "Manual",
      "links": [
        {
          "href": "https:\/\/mars:9090\/nwrestapi\/v1\/global\/clients
\/82.0.31.22.0.0.0.0.33.12.41.87.128.222.109.201\/backups\/
67f3852c-00000006-f03b3bb2-573b3bb2-00120c00-80c64a29",
          "rel": "item"
        }
      ],
      "name": "\/etc\/hosts",
      "retentionTime": "2038-01-18T22:14:06-05:00",
      "saveTime": "2016-05-17T11:41:38-04:00",
      "shortId": "4030413746",
      "size": {
        "unit": "Byte",
        "value": 6736
      },
      "type": "File"
    }
   ...
  ]
}
```

APPENDIX A

Troubleshooting

This appendix includes the following topics:

NetWorker REST API log files

The following table provides a summary of the log files available for the NetWorker REST API.

Component	File name and default location	Description
Installation log Tomcat Access log Apache Catalina log	Refer to the NetWorker Authentication Service logs for these log files.	The NetWorker REST API service is deployed in the same Apache Tomcat container as the NetWorker authentication service. The NetWorker REST API uses the same installation log, Tomcat Access log and Apache Catalina log with NetWorker Authentication service.
NetWorker REST API log	• Linux: `/nsr/logs/restapi/restapi.log` • Windows: C:\Program `Files\EMC NetWorker\nsr\logs\restapi\restapi.log`	Main NetWorker REST API log file.

NetWorker REST API log file management

The NetWorker REST API uses the `Logback` API to manage log files.

To modify how NetWorker REST API manages the `restapi.log` log file, edit the `logback.xml` file, which is found in the following locations:

- Linux: `/nsr/authc/webapps/nwrestapi/WEB-INF/classes`
- Windows: `C:\Program Files\EMC NetWorker\nsr\authc-server\tomcat\webapps\nwrestapi\WEB-INF\classes`

This section describes how to modify the commonly used log attributes in the logback.xml file. Logback project documentation provides more detailed information about each attribute in the logback.xml file.

Modifying the logging level

In the `logger` configuration element, the value of the level attribute defines the level of logging that the NetWorker REST API writes to the log files. By default, the NetWorker REST API sets the logging level to info and messages appear in the log files. There are five standard log levels: trace, debug, info, warn, error.

To change the logging level to error, modify the level attribute to appear as follows:

```
<logger name="com.emc.nw.webapi" level="error"/>
```

Modifying the rollover period

In the `rollingPolicy` configuration element, the `fileNamePattern` element can be used to define the rollover period of the `restapi.log` file. When the log file reaches the end of a rolling period, the NetWorker REST API renames the log file for archival purposes and creates new log file. The rollover period is inferred from the value of

`fileNamePattern`. By default, the NetWorker REST API sets the rollover period to `monthly`.

To change the rollover period to daily, modify the `fileNamePattern` element to appear as follows:

```
<fileNamePattern>${logdir}/restapi_%d.log</fileNamePattern>
```

Please refer to Logback project document on how to modify the rollover period.

Modifying the number of rollover log files

In the `rollingPolicy` configuration element, the `maxHistory` element defines the number of restapi.log rollover log files that the NetWorker REST API maintains. It controls the maximum number of archive files to keep, deleting older files. By default, the NetWorker REST API maintains six rollover log files.