Dell EMC DSS 9000 RackManager

User Guide

DELL EMC

Notes, cautions, and warnings

NOTE: A NOTE indicates important information that helps you make better use of your product.

CAUTION: A CAUTION indicates either potential damage to hardware or loss of data and tells you how to avoid the problem.

WARNING: A WARNING indicates a potential for property damage, personal injury, or death.

2017 - 09

Rev. A00

Contents

1

Documentation Resources

For more information about RackManager's toolkit utilities, services and other functions, go to RackManager github Published Documentation Repository at https://github.com/DellESI/RackManager-Docs-Published.

2

Overview

The Rack Level Management is the unique feature of DSS 9000, which is designed to meet the expectations of end users for simple and secure management of modern scalable platform hardware. The DSS 9000 RackManager is an embedded CentOS server in the DSS9000 rack that provides enhanced rack-level management functions.

Topics:

Using RackManager

RackManager interfaces with other controllers in the DSS9000 rack using the internal rack ManagementNetwork. It has an enhanced GbE internal ManagementNetwork that interconnects RackManager directly with the sled BMCs and other infrastructure controllers. For e.g. the MCs, IMs, BCs

RackManager uses the ManagementNetwork to:

- Communicate with the internal rack infrastructure management controllers -- primarily the managed MC.
- Communicate directly with node BMCs. This allows RackManager software to use any network-based API supported by the BMC. For e.g. ipmitool, WSManagement, racadm, redfish, etc. over the high-speed GbE internal management network.

Interfaces for RackManager

RackManager uses the following interfaces.

- **External RJ45 Ethernet Interface**
  - DSS9000 integrated with RackManager- These ports are labeled Mgmt1 and Mgmt2 on the IM module in the first PowerBay.
- **Serial Console**
  - DSS9000 integrated with RackManager - A USB-Serial interface on the Infrastructure Module can be used to connect to either the RackManager serial port or the RackManager mini-BMC.

3

Connecting to the RackManager

1 Physical Network Connection
   a Plug in a RJ45 cable which is connected to your DHCP server to the port labeled "Mgmt1" which is located on the DSS 9000's IM module.
   b This will assign an IP Address from your DHCP server pool from which you can access the RackManager
2 Logging into the RackManager:
   a `ssh` into the RackManager using the assigned DHCP IP address of Mgmt1 using username 'root' and password 'calvin'

```
foouser@serveruno:~> ssh root@<Mgmt1_DHCP_IP>
root@192.168.16.191's password:
Last login: Thu Sep 21 21:04:00 2017 from XXX.XXX.XXX
[root@RackManagerMini1 ~]#
```

4

Example Utilities

1 'RMversion' – Show the RackManager toolkit version
2 RMg5cli -- a utility to connect to the DSS9000 MC and execute legacy G5 CLI commands
3 RMredfishtool -- a version of redfishtool CLI that is optimized for the RackManager toolkit on DSS9000

Topics:

'RMversion' – Show the RackManager toolkit version

```
[root@RackManagerMini1 ~]# RMversion
RackManager Toolkit 1.0.0
```

RMg5cli -- a utility to connect to the DSS9000 MC and execute legacy G5 CLI commands

Example1: start interactive legacy G5 CLI on MMC1

```
[root@RackManagerMini1 ~]# RMg5cli
=== Dell DCS Domain Controller CLI, 03.40 ===

/DEVICEMANAGER-> show rack1
/DeviceManager/Rack1
    Properties
        AssetTag(W) = Rack_AssetTag
        ChassisPowerOffDelay = 10 s
        ConfigName(W) =
        Grouping(W) = DISABLE
        Height(W) = Rack_Height
        Infrastructure = G5.5
        InternalMgtNetworkConfig = G5IDprotocolV2
        InternalMgtNetworkG5IdNetmask(W) = 255.255.255.0
   InternalMgtNetworkG5IdNetwork(W) = 10.253.0.0
        LastPowerChangeStatus = NA
        LastUpgradeStatus = UPDATE_SUCCESSFUL
        Location(W) = Rack_Location
        PowerState = ON
        PowerThrottling(W) = DISABLE
        RackName(W) = Rack_Name
        RackNumber(W) = 1
        RackType(W) = Rack_Type
        SerialNumber = MINI_SerialNumber
        ServiceTag = Rack_ServiceTag
        Status = NORMAL
        TotalPower = 429 W
        Type = Rack
    Targets
```

```
        Block1
        Block2
        Block3
        PowerBay1
        Summary

/DEVICEMANAGER->
```

Example2: Execute command to show detail of the first node in the first block

```
[root@RackManagerMini1 ~]# RMg5cli SHOW /DEVICEMANAGER/RACK1/Block1/Sled1
/DeviceManager/Rack1/Block1/Sled1
    Properties
        BlockNumber = 1
        Children = BLOCK1:SLED2
        ConfigName(W) =
        CurrentReading = 11.416 A
        CurrentState = NORMAL
        GroupState = PARENT
        Health = NORMAL
        IdLedState(W) = OFF
   InletTempReading = 27 C
        InletTempState = NORMAL
        Name = DCS_G5
        ParentSled(W) = NA
        PowerConsumptionReading = 137 W
        PowerConsumptionState = NORMAL
        PowerState = ON
        PresenceState = PRESENT
        RackNumber = 1
        SledInterfaceType = 0x40 (GHOST/ROVER)
   SledNumber = 1
        SledType = Compute
        Type = Sled
        VoltageReading = 12 V
        VoltageState = NORMAL
    Targets
        Node1
```

Example3: Getting RMg5cli Help

```
[root@RackManagerMini1 ~]# RMg5cli -h
    Usage:  RMg5cli  [-v][-V][-h][-m <mmc>] [<sub-command-and-args>]
```

OPTIONS:

-V	-- display version and exit
-h	-- display usage help
-m <mmc>	-- target the specified Managed-MC. `default=MC1_1`
-v	-- verbose output. can repeat for additional level of verbosity

-v	-- gives the RM user group that this user is a member of
-vv	-- adds the ssh command sent to the MC
-vvv	-- progname, version, & verboseLvl

RMredfishtool -- a version of redfishtool CLI that is optimized for RackManager toolkit on DSS9000

Example1: Execute “Systems sub-command to show detail about the /Systems collection

```
[root@RackManagerMini1 ~]# RMredfishtool -r localhost -u root -p calvin Systems
{
    "Name": "Computer System Collection",
    "@odata.type": "#ComputerSystemCollection.ComputerSystemCollection",
    "@odata.context": "/redfish/
v1/$metadata#ComputerSystemCollection.ComputerSystemCollection", Members":
        [
            {
                "@odata.id": "/redfish/v1/Systems/Rack1-Block2-Sled4-Node1"
            }
            {
                "@odata.id": "/redfish/v1/Systems/Rack1-Block1-Sled1-Node1"
            }
            {
                "@odata.id": "/redfish/v1/Systems/Rack1-Block3-Sled4-Node1"
            }
        ],
        "Members@odata.count": 3
        "@odata.id": "/redfish/v1/Systems"
```

Example2: Execute raw command to show detail of the RackManager Redfish Service root

```
[root@RackManagerMini1 ~]# RMredfishtool -r localhost -u root -p calvin  raw GET /redfish/v1
{
    "Id": "RootService",
    "Managers": {
        "@odata.id": "/redfish/v1/Managers"
             },
        "Oem": {},
        "Links": {
            "Sessions": {
            "@odata.id": "/redfish/v1/SessionService/Sessions"
            }
    }
    "Systems": {
        "@odata.id": "/redfish/v1/Systems"
    },
    "RedfishVersion": "1.0.0",
    "UUID": "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
    "Chassis": {
        "@odata.id": "/redfish/v1/Chassis"
    },
    "@odata.type": "#ServiceRoot.1.0.0.ServiceRoot",
    "JsonSchemas": {
        "@odata.id": "/redfish/v1/JsonSchemas"
     },
    "Name": "RackManager Root Service",
    "@odata.context": "/redfish/v1/$metadata#ServiceRoot.ServiceRoot",
    v"@odata.id": "/redfish/v1/",
    "@odata.id": "/redfish/v1/",
  "Registries": {
```

```
        "@odata.id": "/redfish/v1/Registries"
    },
    "SessionService": {
        "@odata.id": "/redfish/v1/SessionService"
    },
    "AccountService": {
        "@odata.id": "/redfish/v1/AccountService"
    }
}
```

Example3: Getting RMredfishtool help

```
[root@RackManagerMini1 ~]# RMredfishtool -h
    Usage:
    RMredfishtool:    {} [OPTIONS]  <SubCommand> <operation> [<args>]...
    RMredfishtool:    {} [OPTIONS]  hmraw  <method> <hmUrl> [<data>]
```

Common OPTIONS:

-V, --version -- show RMredfishtool version, and exit

-h, --help -- show Usage, Options, and list of subCommands, and exit

-v, --verbose -- verbose level, can repeat up to 5 times for more verbose output

```
-v(header,)
-vv(+addl info),
-vvv(Requesttrace),
-vvvv(+subCmd dbg),
-vvvvv(max dbg)
```

-s, --status -- status level, can repeat up to 5 times for more status output

```
-s(http_status),
-ss(+r.url, +r.elapsed executionTime),
-sss(+request hdrs,data,authType, +response status_code, +response
executionTime +login auth token/sessId/sessUri)
-ssss(+response headers),
-sssss(+response data)
```

-u <user>, --user=<usernm> -- username used for remote redfish authentication

-p <passwd>, --password=<passwd> -- password used for remote redfish authentication

-r <rhost>, --rhost=<rhost> -- remote redfish service hostname or IP:port

-t <token>, --token=<token> -- redfish auth session token-for sessions across multiple calls

-q, --quiet -- quiet mode--suppress error, warning, and diagnostic messages

-c <cfgFile>, --config=<cfgFile> -- read options (including credentials) from file `<cfgFile>`

-T <timeout>, --Timeout=<timeout> -- timeout in seconds for each http request. `Default=10`

-P <property>, --Prop=<property> -- return only the specified property. Applies only to all "`get`" operations

Options used by "raw" subcommand:

-d <data>, --data=<data> -- the http request "data" to send on PATCH,POST,or PUT requests

Options to specify top-level collection members: eg: Systems -I <sysId>

-I <Id>,--Id=<Id> -- Use`<Id>` to specify the collection member

-M <prop>:<val> --Match=<prop>:<val> -- Use `<prop>=<val>` search to find the collection member

-F, --First -- Use the 1st link returned in the collection or 1st "matching" link if used with -M

-1, --One -- Use the single link returned in the collection. Return error if more than one member exists

-a, --all -- Returns all members if the operation is a Get on a top-level collection like Systems

-L <Link>, --Link=<Link> -- Use `<Link> (eg /redfish/v1/Systems/1)` to reference the collection member.
-- If `<Link>` is not one of the links in the collection, and error is returned.

Options to specify 2nd-level collection members: eg: Systems -I<sysId> Processors -i<procId>

-i <id>, --id=<id> -- use <id> to specify the 2nd-level collection member

-m <prop>:<val> --match=<prop>:val> --use <prop>=<val> search of 2nd-level collection to specify member

-l <link> --link=<link> -- Use <link> (eg /redfish/v1/SYstems/1/Processors/1) to reference a 2nd level resource
-- A -I|M|F|1|L option is still required to specify the link to the top-lvl collection

-a, --all -- Returns all members of the 2nd level collection if the operation is a Get on the
-- 2nd level collection (eg Processors). -I|M|F|1|L still specifies the top-lvl collection.

Additional OPTIONS:

-W <num>:<connTimeout>	-- Send up to <num> {GET /redfish} requests with <connTimeout> TCP connection timeout --Wait=<num>:<ConnTimeout> --before sending subcommand to rhost. Default is -W 1:3
-A <Authn>, --Auth <Authn>	-- Authentication type to use: Authn={None|Basic|Session} Default is Basic
-S <Secure>, --Secure=<Secure>	-- When to use https: (Note: doesn't stop rhost from redirect http to https) Secure>={Always | IfSendingCredentials | IfLoginOrAuthenticatedApi(default) }
-R <ver>, --RedfishVersion=<ver>	-- The Major Redfish Protocol version to use: ver={v1(dflt), v<n>, Latest}
-C, -CheckRedfishVersion	-- tells Redfishtool to execute GET /redfish to verify that the rhost supports the specified redfish protocol version before executing a sub-command. The -C flag is auto-set if the -R Latest or -W ... options are selected
-H <hdrs>, --Headers=<hdrs>	-- Specify the request header list--overrides defaults. Format "{ A:B, C:D...}"
-D <flag>, --Debug=<flag>	-- Flag for dev debug. <flag> is a 32-bit uint: 0x<hex> or <dec> format

Subcommands:

hello	-- redfishtool hello world subcommand for dev testing
about	-- display version and other information about this version of RMredfishtool
versions	-- get redfishProtocol versions supported by rhost: GET ^/redfish
root| serviceRoot	-- get serviceRoot resouce: GET ^/redfish/v1/
Systems	-- operations on Computer Systems in the /Systems collection
Chassis	-- operations on Chassis in the /Chassis collection
Managers	-- operations on Managers in the /Managers collection
AccountService	-- operations on AccountService including user administration
SessionService	-- operations on SessionService including Session login/logout
odata	-- get the Odata Service document: GET ^/redfish/v1/odata
metadata	-- get the CSDL metadata document: GET ^/redfish/v1/$metadata
raw	-- subcommand to execute raw http methods(GET,PATCH,POST...) and URIs

For Subcommand usage, options, operations, help: RMredfishtool <SubCommand> -h -- usage and options for specific subcommand