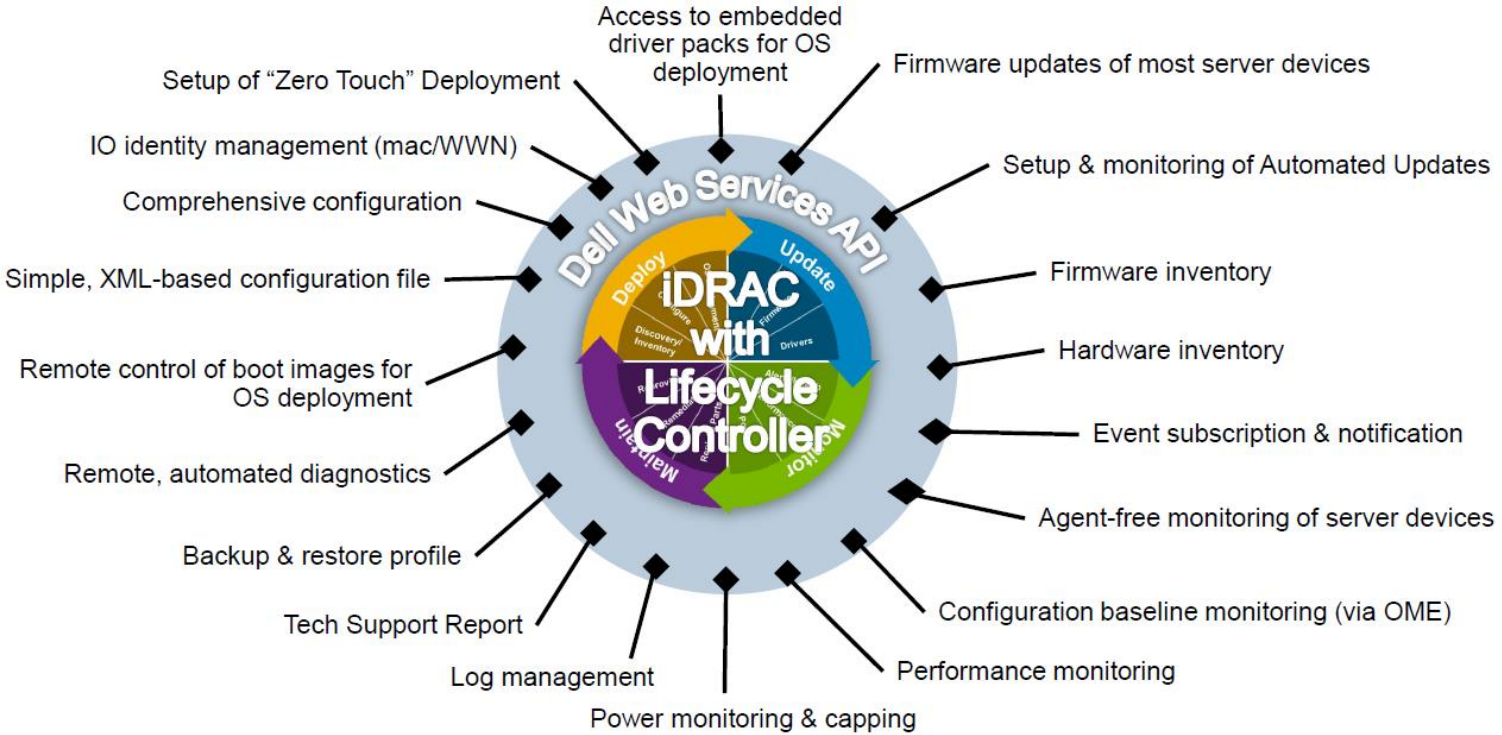
Automating idrac configuration

learned from santa clara lab

By: Mohamed ELMesseiry

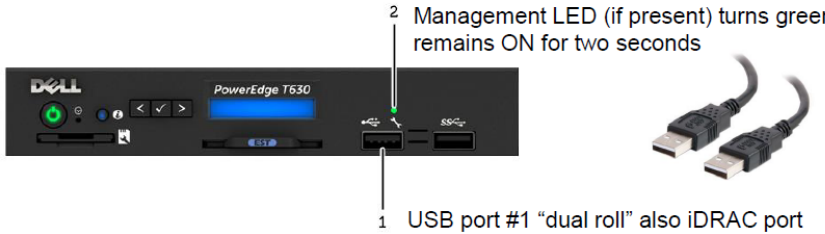
iDRAC Management API

Powerful API with capabilities spanning entire server lifecycle



iDRAC Direct (USB Connect)

Full iDRAC GUI Session –No Nic Connection Required

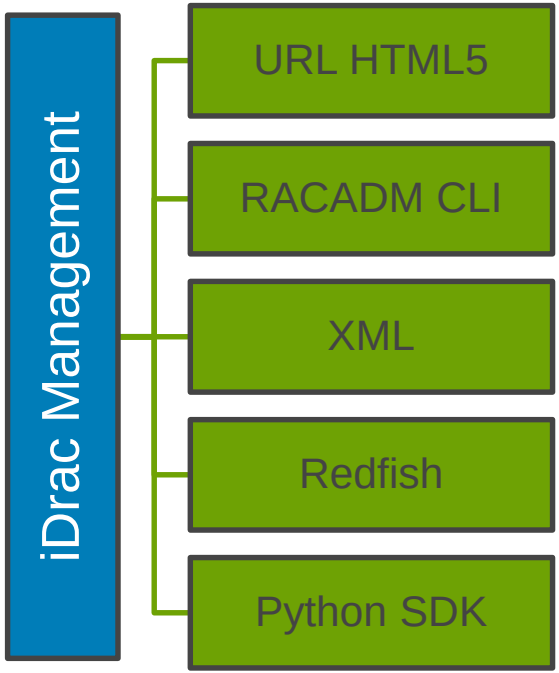


- Support by 13 gen servers with iDRAC 8 & IDRAC direct shared usb port
- Connect a Type A/A USB cable from the laptop to server USB port #1
- iDRAC USB access port / enabled by default (can be disabled)

Customer Benefit

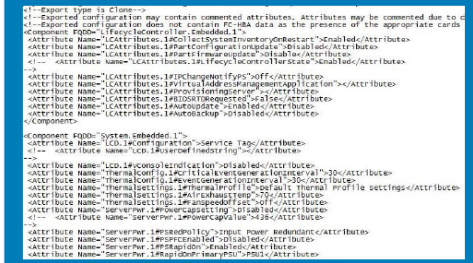
Launch browser or ssh session from connected laptop using 169.254.0.3 for full access to iDRAC gui/console, no network connection required

How I would manager 1000’s of servers?



#2 : Server Configuration Profile -XML Template

BIOS, iDRAC, RAID, NIC, FC-HBA & Lifecycle Controller configuration information



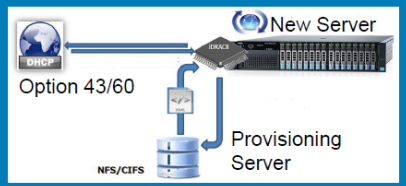
Export/import the XML profile Via :

UBS

OME deployment solution

Remote CLI/script

DHCP zero touch / bare metal



Customer Benefit

One file automaticity configure 1000's of servers fast & consistently

Racadm xml & Redfish API

```
# racadm get -f serverscp -t xml -u <CIFSuser> -p <CIFSpassword> -l
<CIFS Share path>

# racadm get -f serverscp -t xml -l <NFS Share path>
```

XML

For 14th generation PowerEdge servers, additional RACADM options include:

```
# racadm -get -f serverscp -t JSON -u <HTTP/S user> -p <HTTP/S password> -l
<HTTP/S Share path>
```

JSON

Redfish API, Show Embedded IDRAC Interface Details

Not secure | https://172.17.72.18/redfish/v1/Managers/iDRAC.Embedded.1/EthernetInterfaces/


```
{"@odata.context":"/redfish/v1/$metadata#EthernetInterfaceCollection.EthernetInterfaceCollection","@odata.id":"/redfish/v1/Managers/iDRAC.Embedded.1/EthernetI
nterfaces","@odata.type":"#EthernetInterfaceCollection.EthernetInterfaceCollection","Description":"Collection of EthernetInterfaces for this
Manager","Members":
[{"@odata.id":"/redfish/v1/Managers/iDRAC.Embedded.1/EthernetInterfaces/iDRAC.Embedded.1%23NIC.1"}],"Members@odata.count":1,"Name":"Ethernet Network Interface
Collection"}
```


Not secure | https://172.17.72.18/redfish/v1/Managers/iDRAC.Embedded.1/EthernetInterfaces/iDRAC.Embedded.1%23NIC.1


```
{"@odata.context":"/redfish/v1/$metadata#EthernetInterface.EthernetInterface","@odata.id":"/redfish/v1/Managers/iDRAC.Embedded.1/EthernetInterfaces/iDRAC.Embe
dded.1%23NIC.1","@odata.type":"#EthernetInterface.v1_0_2.EthernetInterface","Description":"Management Network Interface","FQDN":"","HostName":"idrac-
6B79HK2","IPv4Addresses":
[{"Address":"172.17.72.18","AddressOrigin":null,"Gateway":"172.17.72.254","SubnetMask":"255.255.255.0"}],"IPv4Addresses@odata.count":1,"IPv6AddressPolicyTable
":[],"IPv6AddressPolicyTable@odata.count":0,"IPv6Addresses":[{"Address":"::","AddressOrigin":"Static","AddressState":"Preferred","PrefixLength":64},
{"Address":"::","AddressOrigin":null,"AddressState":"Failed","PrefixLength":64},{"Address":"::","AddressOrigin":null,"AddressState":null,"PrefixLength":64},
{"Address":"::","AddressOrigin":null,"AddressState":null,"PrefixLength":64},{"Address":"::","AddressOrigin":null,"AddressState":null,"PrefixLength":64},
{"Address":"::","AddressOrigin":null,"AddressState":null,"PrefixLength":64},{"Address":"::","AddressOrigin":null,"AddressState":null,"PrefixLength":64},
{"Address":"::","AddressOrigin":null,"AddressState":null,"PrefixLength":64},{"Address":"::","AddressOrigin":null,"AddressState":null,"PrefixLength":64},
{"Address":"::","AddressOrigin":null,"AddressState":null,"PrefixLength":64},{"Address":"::","AddressOrigin":null,"AddressState":null,"PrefixLength":64},
{"Address":"::","AddressOrigin":null,"AddressState":null,"PrefixLength":64},{"Address":"::","AddressOrigin":null,"AddressState":null,"PrefixLength":64},
{"Address":"::","AddressOrigin":null,"AddressState":null,"PrefixLength":64},
{"Address":"::","AddressOrigin":null,"AddressState":null,"PrefixLength":64}],"IPv6Addresses@odata.count":15,"IPv6DefaultGateway":"::","IPv6StaticAddresses":
[{"Address":"::","PrefixLength":64}],"IPv6StaticAddresses@odata.count":1,"Id":"iDRAC.Embedded.1#NIC.1","InterfaceEnabled":true,"MACAddress":"E0:D8:48:19:23:43
","MTUSize":1500,"MaxIPv6StaticAddresses":16,"Name":"Manager Ethernet Interface","NameServers":
["8.8.8.8","8.8.4.4","8.8.8.8","8.8.4.4","::","::","::","::"],"NameServers@odata.count":8,"PermanentMACAddress":"E0:D8:48:19:23:43","Status":
{"Health":"Ok","State":"Enabled"},"VLAN":{"VLANEnable":false,"VLANId":1}}
```

RACADM CLI

Local or remote cli

```
sudo echo 'deb http://linux.dell.com/repo/community/ubuntu xenial openmanage' | sudo tee -a /etc/apt/sources.list.d/linux.dell.com.sources.list
sudo gpg --keyserver pool.sks-keyservers.net --recv-key 1285491434D8786F
gpg -a --export 1285491434D8786F | sudo apt-key add -
sudo apt-get update
sudo apt-get install libssl-dev
sudo apt-get install srvadmin-all
racadm -r 172.17.72.18 -u roor -p calvin getsysinfo
```

```
racadm -r 172.17.72.18 -u developer -p changeme sshpkauth -v -i 2 -k all
```

```
/home/developer/.ssh/id_rsa.pub
```

Location of public key of automation/DevOps machine

```
racadm -r 172.17.72.18 -u developer -p changeme  sshpkauth -i 3 -k 2 -f /home/developer/.ssh/id_rsa.pub
racadm -r 172.17.72.18 -u root -p calvin  sshpkauth -i 2 -k 1 -f /home/developer/.ssh/id_rsa.pub
```

Create users using Racadm CLI

```
racadm -r 172.17.72.18 -u root -p calvin config -g cfgUserAdmin -o cfgUserAdminUserName -i 3 developer
racadm -r 172.17.72.18 -u root -p calvin config -g cfgUserAdmin -o cfgUserAdminPassword -i 3 changeme

racadm -r 172.17.72.18 -u root -p calvin config -g cfgUserAdmin -i 3 -o cfgUserAdminPrivilege 0x00000001
racadm -r 172.17.72.18 -u root -p calvin config -g cfgUserAdmin -i 3 -o cfgUserAdminIpmiLanPrivilege 2
racadm -r 172.17.72.18 -u root -p calvin config -g cfgUserAdmin -i 3 -o cfgUserAdminIpmiSerialPrivilege 2
racadm -r 172.17.72.18 -u root -p calvin config -g cfgUserAdmin -i 3 -o cfgUserAdminSolEnable 1
racadm -r 172.17.72.18 -u root -p calvin config -g cfgUserAdmin -i 3 -o cfgUserAdminEnable 1

racadm -r 172.17.72.18 -u root -p calvin getconfig -u developer
```

```
racadm -r 172.17.72.18 -u root -p calvin --nocertwarn set IDRAC.Users.4.UserName messei
racadm -r 172.17.72.18 -u root -p calvin --nocertwarn set IDRAC.Users.4.Password changeme
racadm -r 172.17.72.18 -u root -p calvin --nocertwarn set IDRAC.Users.4.Privilege 0x000001ff
racadm -r 172.17.72.18 -u root -p calvin --nocertwarn set IDRAC.Users.4.IpmiLanPrivilege 4
racadm -r 172.17.72.18 -u root -p calvin --nocertwarn set IDRAC.Users.4.SolEnable 1
racadm -r 172.17.72.18 -u root -p calvin --nocertwarn set IDRAC.Users.4.Enable 1
#racadm -r 172.17.72.18 -u root -p calvin set IDRAC.Users.4.IpmiSerialPrivilege 4
```

Admin user creation script

```
#!/bin/sh

adminUser="developer"
adminPass="changeme"
idracDefaultUser="root"
idracDefaultPass="calvin"
publicKeyFile="/home/developer/.ssh/id_rsa.pub"
userIndex="4"


for server in `cat server.list`;
do
    echo "Configuring idrac for server: $server ..."
    echo "Creating user: $adminUser user with password: $adminPass ...."

    racadm -r $server -u $idracDefaultUser -p $idracDefaultPass --nocertwarn set IDRAC.Users.$userIndex.UserName $adminUser
    racadm -r $server -u $idracDefaultUser -p $idracDefaultPass --nocertwarn set IDRAC.Users.$userIndex.Password $adminPass
    racadm -r $server -u $idracDefaultUser -p $idracDefaultPass --nocertwarn set IDRAC.Users.$userIndex.Privilege 0x000001ff
    racadm -r $server -u $idracDefaultUser -p $idracDefaultPass --nocertwarn set IDRAC.Users.$userIndex.IpmiLanPrivilege 4
    racadm -r $server -u $idracDefaultUser -p $idracDefaultPass --nocertwarn set IDRAC.Users.$userIndex.SolEnable 1
    racadm -r $server -u $idracDefaultUser -p $idracDefaultPass --nocertwarn set IDRAC.Users.$userIndex.Enable 1

    echo "done creating user"

    echo "checking public key generation on localhost"
    if [ -f "$publicKeyFile" ]
    then
        echo "$publicKeyFile was found"
        echo "Uploading public key from localhost:"
        racadm -r $server -u $idracDefaultUser -p $idracDefaultPass --nocertwarn sshpkauth -i $userIndex -k 1 -f /home/developer/.ssh/id_rsa.pub

    else
        echo "$publicKeyFile was not found, attempting to create file"
        ssh-keygen -t rsa -N "" -f $publicKeyFile
    fi

done
```

Admin user creation script

```
#!/bin/sh

adminUser="developer"
adminPass="changeme"
idracDefaultUser="root"
idracDefaultPass="calvin"
publicKeyFile="/home/developer/.ssh/id_rsa.pub"
userIndex="4"
list="deletevDisk.list"

for server in `cat $list`;
do
    echo "Configuring idrac for server: $server ..."
    echo "getting the existing vdisks"

    vDiskCmdOutput=`racadm -r $server -u $adminUser -p $adminPass --nocertwarn raid get vdisks`
    echo "\n\n"
    echo "--- $server -----------------------------"
    echo "$vDiskCmdOutput"
    echo "-------------------------------------------"
    echo ">> Raid 0 is configured with 2 disks ..."
    echo ">> deleting virtual disks Disk.Virtual.0:RAID.Integrated.1-1 ...."

    racadm -r $server -u $adminUser -p $adminPass --nocertwarn storage deletevd:Disk.Virtual.0:RAID.Integrated.1-1
    racadm -r $server -u $adminUser -p $adminPass --nocertwarn jobqueue  create RAID.Integrated.1-1 -s TIME_NOW --realtime
    racadm -r $server -u $adminUser -p $adminPass --nocertwarn jobqueue view



    echo "\n\n"
done
```

Ansible Automation

```
ansible-playbook -k getsysinfo.yml
```

```yaml
---
- hosts: openstack_blades
  tasks:
  - name: "get the status of the storage"
    raw: racadm storage get status
    register: result
```

https://github.com/dell/redfish-ansible-module

https://github.com/dell/Dell-EMC-Ansible-Modules-for-iDRAC

Ansible Role Examples

Create user, delete raid, enable snmp

```
---
- hosts: hosts
  connection: local
  name: Configure the iDRAC services attributes
  gather_facts: False

  tasks:
  - name: Configure the iDRAC services attributes
    dellemc_configure_idrac_services:
       idrac_ip:   "x.x.x.x"
       idrac_user: "user"
       idrac_pwd:  "pwd"
       share_name: "x.x.x.x:/NFSSahre"
       share_pwd: "share_pwd"
       share_user: "share_user"
       share_mnt:  "/mnt/mntpoint"
       snmp_enable: "Enabled"
       snmp protocol: "All"
```

```
-
hosts: hosts
connection: local
name: Removve RAID
gather_facts: False

tasks:
- name: Removve RAID
  dellemc_configure_raid:
     idrac_ip:   "x.x.x.x"
     idrac_user: "user"
     idrac_pwd:  "pwd"
     share_name: "x.x.x.x:/NFSShare"
     share_pwd: "share_pwd"
     share_user: "share_user"
     share_mnt:  "/mnt/mntpoint"
     state: "absent"
     vd_name: "VD0"
     controller_fqdd: "RAID.Integrated.1-1"
```

```
---
- hosts: hosts
  connection: local
  name: Configure the iDRAC users attributes
  gather_facts: False

  tasks:
  - name: Configure the iDRAC users attributes
    dellemc_configure_idrac_users:
       idrac_ip:   "x.x.x.x"
       idrac_user: "user"
       idrac_pwd:  "pwd"
       share_name: "x.x.x.x:/NFSShare"
       share_pwd: "Share_pwd"
       share_user: "Share_user"
       share_mnt:  "/mnt/mntpoint"
       action: "create"
       user_name: "user_name"
       user_password: "user_pwd"
       privilege_users: 'Administrator'
       ipmilanprivilege_users: 'Operator'
       ipmiserialprivilege_users: 'Administrator'
       enable_users: 'Enabled'
       solenable_users: 'Enabled'
       protocolenable_users: 'Enabled'
       privacyprotocol_users: 'DES'
       authenticationprotocol_users: 'MD5'
```

Open manage python SDK

Branch: master ▾ Dell-EMC-Ansible-Modules-for-iDRAC / library / dellemc_configure_idrac_users.py

rajeevarakkal Committing local streaming

1 contributor

362 lines (326 sloc) | 14.8 KB | Raw

```
#!/usr/bin/python
# _*_ coding: utf-8 _*_

#
# Dell EMC OpenManage Ansible Modules
# Version 1.0
# Copyright (C) 2018 Dell Inc.

# GNU General Public License v3.0+ (see COPYING or https://www.gnu.org/licenses/gpl-3.0.txt)
# All rights reserved. Dell, EMC, and other trademarks are trademarks of Dell Inc. or its subsidiaries.
# Other trademarks may be trademarks of their respective owners.
#

from __future__ import (absolute_import, division,
                        print_function, unicode_literals)
from builtins import *
from ansible.module_utils.dellemc_idrac import *
from ansible.module_utils.basic import AnsibleModule
from omdrivers.enums.iDRAC.iDRAC import *
# from omsdk.sdkfile import FileOnShare
# import logging.config

ANSIBLE_METADATA = {'metadata_version': '1.1',
                    'status': ['preview'],
                    'supported_by': 'community'}

DOCUMENTATION = """
---
module: dellemc_configure_idrac_users
short_description: Configures the iDRAC users attributes.
version_added: "2.3"
description:
    - This module is responsible for configuring the iDRAC users attributes.
options:
    idrac_ip:
        required: True
        description: iDRAC IP Address
        default: None
    idrac_user:
        required: True
        description: iDRAC username
        default: None
    idrac_pwd:
        required: True
        description: iDRAC user password
        default: None
    idrac_port:
```

Demo