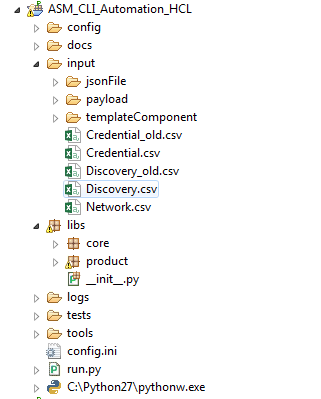
**CLI ASM Automation Framework**

# Key Documentation:

This documents the Active System Manager Automation Framework native to Python 2.7.

# Implementation Structure:



**Fig 1.1The diagram shows the folder structure of CLI ASM automation framework**

**Config floder:** The ‘config’ folder contains configuration and API's url related information and it contains three files.

**config.ini-** It contains configuration information like Appliance IP and Port

**Services.xml -** This file contains the relative internal URLS REST API for all ASM Operations.

**Services\_Public.xml -** This file contains the relative public URLS REST API for all ASM Operations.

**Input folder**: This folder contains all the input information needed at different stages of Automation. This folder contains the below folders/files:

**Payload -** The ‘payload’ folder contains the template for payload for operations like Creating Credential, Network, Discovery, and Login etc.

**jsonFile Folder -** The ‘jsonFile’ folder contains the JSON files of all the flows.

**templateComponent Folder -** The ‘templateComponent’ folder contains the different components of template.

**Credential.csv-** This file contains the credential related information for devices (Server, Storage, and Chassis, switch, scvmm and vcenter.)

NOTE**:** Change this file for any modification in credential related inputs.

**Network.csv-** This files Contains network related information for discovery of networks.

NOTE: change this file for any changes in network inputs

**Discovery.csv-** This files Contains the discovery of multiple resources (servers/switches/Storage) where IP Addresses are not in range (we can give multiple resources to discover in one go) Also resources can be discovered with different credentials.

**docs:** This folder contains all the documents related to the Automation Suite

**libs:** This folder contains two sub folders

**core –** This folder contains dellunit framework modules

**product –** This folder contains ASM related Framework modules

**requests** – This is python package used for making REST Calls

**templates** – This folder contains templates for reporting.

**BaseClass.py** – This module contains all the API for performing all ASM operations

**globalVars.py** – This module contains all the global variables which are used across automation suite.

**hooks.py** – This module contains Setup and Teardown methods to execute at automation suite level.

**utility.py** - This module contains all the methods related to the utilities

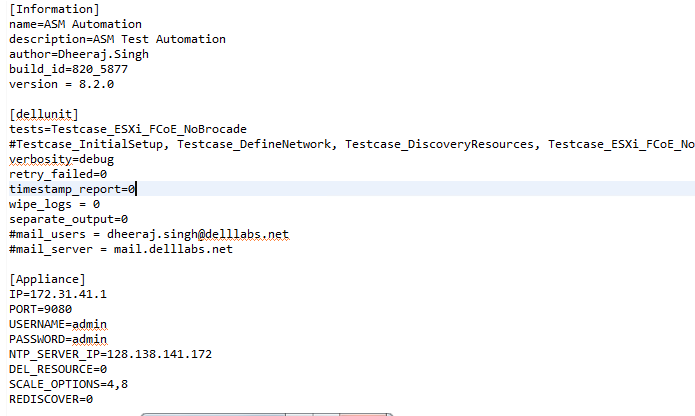
**logs:** This folder contains all the logs. A ‘report.html’ will be generated post execution of all test cases and the other logs generated while executing the test cases. The report will contain the list of test cases executed along with test result.

Hyperlinks “Steps” and “Execution Log” will be provided for each test case in the report page. The link “Steps” will display a list of all logical points of the test case and the status of that case. The link “Execution Log” will display a page containing all the operations that were carried out during the execution of the test case.

**tests:** The ‘tests’ folder contains all the test cases related to ASM. This folder must contain the target test files.

**tools:** The ‘tools’ folder contains tools like plink.exe and pscp.exe which are used to run remote commands on ASM appliance for enabling Debugging.

**config.ini:**



Config.ini has various sections like

**Information**: Project level Information has to be provided here

**dellunit**: Provide which tests to execute in ‘tests’ attribute.

**Appliance**: Provide Appliance Information under Appliance Section

**ADInfo**: Active Directory Information has to be provided here

**Execution**:

Launch console and change directory to the automation folder and run the following command after providing all the required inputs.

**“python run.py config.ini”**

**Possible Value of Network in Json file**:

**"Networks":"ESXI\_EQL\_CONVERGED\_2\_REDUNDANCY\_PARTITION",**

**"Networks":"ESXI\_EQL\_CONVERGED\_2\_REDUNDANCY",**

**"Network”: “ESXI\_EQL\_CONVERGED\_2\_PARTITION",**

**"Networks":"ESXI\_EQL\_CONVERGED\_2",**

**"Networks":"ESXI\_FCOE\_CONVERGED\_2",**

**"Networks":"ESXI\_FCOE\_CONVERGED\_2\_PARTITION",**

**"Networks":"ESXI\_CPL\_ISCSI\_CONVERGED\_2",**

**jsonFile Folder**

1. **Exsi\_EQL\_Conv\_Div.json**

{

"**Template**":

{

"**Name":"OnlyStorage\_Esxi\_Template",**

**"Description":"Deploy storage and scale up an storage",**

**"Flow**":"esxi"

},

"**Server**":

{

"**Instances":1, // possible values are 0, 1, 2, 3**

**"target\_boot\_device\_value":"HD**", **// possible values are HD, SD, iSCSI, FC**

**"OSImage\_Value": ["esxi-5.1"], // possible values are esxi-5.1, esxi-5.5, esxi-6.0, (WindowsR2,WindowsNonR2) for hyperV**

**"iSCSI\_Initiator":"hardware", // possible values are hardware, software**

**"Networks": "** **ESXI\_EQL\_CONVERGED\_2\_REDUNDANCY\_PARTITION",**

**"serverPoolName":"ESXI", // We can give any name**

**"HostNameTemplate":"server${num}", // possible values are**

**server${num}, server** **${service\_tag}, server${model}, server** **${vendor}, server** **${dns}**

**"ServersIPForPool**": ["172.31.61.80", "172.31.61.82", "172.31.32.126", "172.31.32.156"]

},

"**Storage**":

{

**"Instances":1,**

**"Type":"Equallogic",** **// possible values are Equallogic, Compellent and NetApp**

**"Size":["200GB"], // This is size of storage for multiple values we should give like ["200GB", "50GB"]**

**"IPAdress":"172.31.32.108",** **// IP Address of Equallogic or Compellent**

**"Auth\_Type":"chap", // Authentication method for storage volume possible value should be chap** and **iqnip**

**"Initiator\_IQN\_IP\_Addresses":"",** **// Initiator IQN or IP Addresses for Equallogic**

**"Operating\_SystemName\_Compellent":"VMware ESX 5.5",** **// Operating System Name in case of Compellent**

**"PortType\_Compellent":"FibreChannel"** **// Port Type in case of**

**Compellent**

**"Aggregate\_Name\_NetApp":"Aggregate1DP", possible value should be aggr0, Aggregate1DP, Aggregate2RAID4** and **Greatest\_aggregate**

**"Space\_Reservation\_Mode\_NetApp":"file" possible value should be file, none,** and **volume**

},

**"Cluster":**

{

**"Instances":0,**

**"Type":"VMWareCluster",**

**"IPAdress":"172.31.32.195",** **// IP Address of VCenter**

**"ExistingClusterName":"",** **// Existing Cluster Name if we want to use New Cluster leave it balnk**

**"HA\_Config":"false", // possible values are true, false**

**"DRS\_Config":"false", // possible values are true, false**

**"VDS\_Enabled":"false"** **// possible values are true, false**

},

**"VM":**

{

**"Instances":0,**

**"Type":"VMWareVM",**

**"AutogenerateHostName":"true",**

**"OS\_Image":"red7",** **// possible values are esxi-5.1, esxi-5.5, esxi-6.0, red6 and red7**

**"Os\_Image\_Version":"",** **// required if we give OS Image Type as**

**windows2008, windows2012, hyperv**

**"OS\_Image\_Type”: "redhat7",** **// possible values are**

**windows2008, windows2012, hyperv, redhat, redhat7**

**"Product\_Key":""** **// required if we give OS Image Type as**

**windows2008, windows2012, hyperv**

**"CPU\_Count":"2",**

**"Memory\_MB":"1024",**

**"hostname\_HyperVCloneVM":"Clone003",**

**"description\_HyperVCloneVM":"Clone003",**

**"vm\_template\_name":"HCLCentos7",**

**"block\_dynamic\_optimization" : "false",**

**"highly\_available":"true",**

**"start\_action":"always\_auto\_turn\_on\_vm", // possible values are always\_auto\_turn\_on\_vm, never\_auto\_turn\_on\_vm, turn\_on\_vm\_if\_running\_when\_vs\_stopped**

**"stop\_action":"turn\_off\_vm", // possible values are save\_vm, shutdown\_guest\_os, turn\_off\_vm**

**"path":"C:\\ClusterStorage\\Volume1"**

},

**"Application":**

{

**"Instances":0,**

**"Type":"LinuxPostInstall"**, **// possible values are LinuxPostInstall, WindowPostInstall**

**"yum\_proxy":"",** **// Value of Yum Proxy for LinuxPostInstall**

**"install\_packages":"",** **// Value of install packages for LinuxPostInstall**

**"upload\_share":"",** **// Value of upload share for LinuxPostInstall**

**"upload\_file":"",** **// Value of upload file for LinuxPostInstall or WindowPostInstall**

**"execute\_file\_command":"sh",** **// Value of execute file command for LinuxPostInstall or WindowPostInstall**

**"windows\_postinstall\_share":"",** **// Value of share for WindowPostInstall**

**"windows\_install\_command":""** **// Value of install command for WindowPostInstall**

**},**

**"Scaleup":**

**{**

**"Server":**

**{**

**"Sequence":1, // This is order of scalup in case of**

**Multiple scalupe "Sequence":1, will be scale up first after that “Sequence":2 and so on.**

**"Instances":0, // possible values are 0, 1, If we want to scaleup a server then this value will be 1 otherwise 0**

**"target\_boot\_device\_value":"SD",**

**"iSCSI\_Initiator":"hardware",**

**"HostNameTemplate":"server${num}",**

**"OSImage\_Value": ["esxi-5.1"]**

**},**

**"Storage":**

**{**

**"Sequence":2,**

**"Instances":1,**

**"Type":"Equallogic",**

**"Size":["200GB"],**

**"Initiator\_IQN\_IP\_Addresses":"",**

**"Operating\_SystemName\_Compellent":"VMware ESX 5.5",**

**"PortType\_Compellent":"FibreChannel"**

**},**

**"Cluster":**

**{**

**"Sequence":3,**

**"Instances":0,**

**"Type":"VMWareCluster",**

**"HA\_Config":"true",**

**"DRS\_Config":"false",**

**"VDS\_Enabled":"false"**

**},**

**"VM":**

**{**

**"Sequence":4,**

**"Instances":0,**

**"Type":"VMWareVM",**

**"AutogenerateHostName":"true",**

**"OS\_Image":"red7",**

**"Os\_Image\_Version":"",**

**"OS\_Image\_Type" : "redhat7",**

**"Product\_Key":""**

**},**

**"Application":**

**{**

**"Sequence":5,**

**"Instances":0,**

**"Type":"LinuxPostInstall",**

**"yum\_proxy":"",**

**"install\_packages":"",**

**"upload\_share":"",**

**"upload\_file":"",**

**"execute\_file\_command":"sh",**

**"windows\_postinstall\_share":"",**

**"windows\_install\_command":""**

**}**

**}**

**}**