USER GUIDE

Cisco UCS PowerTool, Release 0.9.11.0

1 Overview
2 Management Information Model
3 Installation
4 Examples
5 Samples
6 Related Cisco UCS Documentation
Obtaining Documentation and Submitting a Service Request

1 Overview

Cisco UCS PowerTool is a PowerShell module which helps automate all aspects of Cisco UCS Manager including server, network, storage and hypervisor management. PowerTool enables easy integration with existing IT management processes and tools.

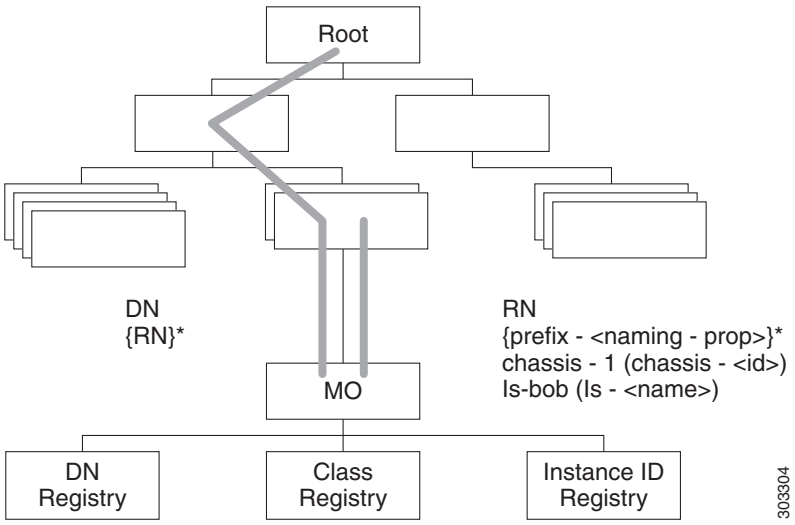
Bulk of the PowerTool cmdlets work on the UCS Manager's Management Information Tree (MIT), performing create, modify or delete actions on the Managed Objects (MO) in the tree. The next chapter provides an overview of the Cisco UCS Management Information Model (MIM) and relation of PowerTool cmdlets with it.

One of the easiest ways to learn UCS configuration through PowerTool is to generate PowerTool cmdlets, for configuration actions performed with the GUI, using the ConvertTo-UcsCmdlet described in "PowerTool Cmdlet Generation" section on page 12 section.

2 Management Information Model

All the physical and logical components that comprise a Cisco UCS domain are represented in a hierarchical Management Information Model (MIM), referred to as the Management Information Tree (MIT). Each node in the tree represents a Managed Object (MO), uniquely identified by its Distinguished Name (DN). Figure 1 illustrates the MIM.

Figure 1 Management Information Model



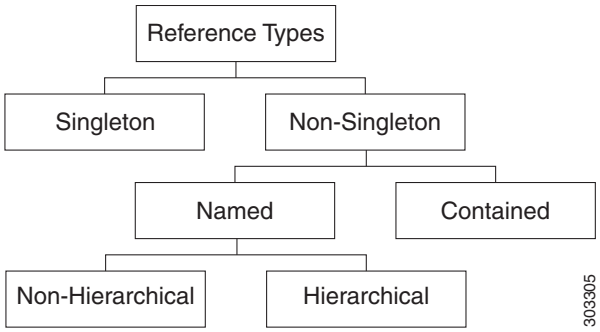
The following illustration shows a sample (partial) MIT for three chassis.

Tree (topRoot)	Distinguished Name
|— sys	sys
|— chassis-1	sys/chassis-1
|— chassis-2	sys/chassis-2
|— chassis-3	sys/chassis-3
|— blade-1	sys/chassis-3/blade-1
| |— adaptor-1	sys/chassis-3/blade-1/adaptor-1
|— blade-2	sys/chassis-3/blade-2
|— adaptor-1	sys/chassis-3/blade-2/adaptor-1
|— adaptor-2	sys/chassis-3/blade-2/adaptor-2

Managed Objects

Managed Objects (MO) are abstractions of Cisco UCS domain resources, such as fabric interconnects, chassis, blades, and rack-mounted servers. Managed Objects represent any physical or logical entity that is configured / managed in the Cisco UCS MIT. For example, physical entities such as Servers, Chassis, I/O cards, Processors and logical entities such as resource pools, user roles, service profiles, and policies are represented as managed objects.

Figure 2 Managed Objects



Every managed object is uniquely identified in the tree with its Distinguished Name (Dn) and can be uniquely identified within the context of its parent with its Relative Name (Rn). The Dn identifies the place of the MO in the MIT. A Dn is a concatenation of all the relative names starting from the root to the MO itself. Essentially, Dn = [Rn]/[Rn]/[Rn]/…/[Rn].

In the example below, Dn provides a fully qualified name for adaptor-1 in the model.

```
< dn = "sys/chassis-5/blade-2/adaptor-1" />
```

The above written Dn is composed of the following Rn:

```
topSystem MO: rn="sys" equipmentChassis MO: rn="chassis-<id>" computeBlade MO: rn ="blade-<slotId>" adaptorUnit
MO: rn="adaptor-<id>"
```

A Relative Name (Rn) may have the value of one or more of the MO's properties embedded in it. This allows in differentiating multiple MOs of the same type within the context of the parent. Any properties that form part of the Rn as described earlier are referred to as Naming properties.

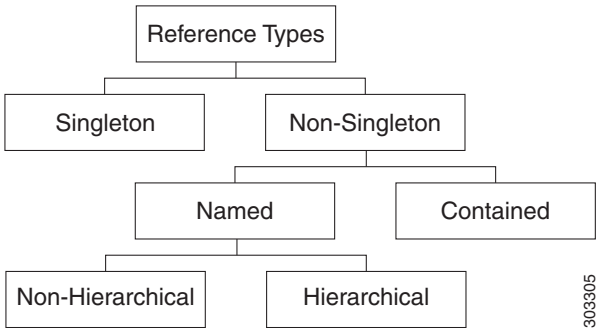
For instance, multiple blade MOs reside under a chassis MO. The blade MO contains the blade identifier as part of its Rn (blade-[Id]), thereby uniquely identifying each blade MO in the context of a chassis.

References to Managed Objects

The contents of the managed objects are referred to during the operation of Cisco UCS. Some of the MOs are referred to implicitly (PreLoginBanner during login) or as part of deployment of another MO (The Service Profile MO may refer to a template or a VNIC refers to a number of VLAN MOs).

The different types of references can be classified as shown below:

Figure 3 References to Managed Objects



A singleton MO type is found at most once in the entire MIT and is typically referred to implicitly.

Non-Singleton MO type may be instantiated one or more times in the MIT. In many cases, when an MO refers to another, the reference is made by name. Depending on the type of the referenced MO, the resolution may be hierarchical. For instance, a service profile template is defined under an org. Since an org may contain sub-orgs, a sub org may have a service profile template

defined with the same name. Now, when a service profile instance refers to a service profile template (by name), the name is looked up hierarchically from the org of the service profile instance up until the root org. The first match is used. If no match is found, the name "default" is looked up in the similar way and the first such match is used.

Reference Type	Example
Singleton	ChassisDiscoveryPolicy PreLoginBanner
Non-Singleton / Named / Non-Hierarchical	CallHomePolicy
Non-Singleton / Named / Hierarchical	BiosPolicy BootPolicy
Non-Singleton / Contained	BootDefinition under LsServer (ServiceProfile) VnicEtherIf under VnicEther

Properties of Managed Objects

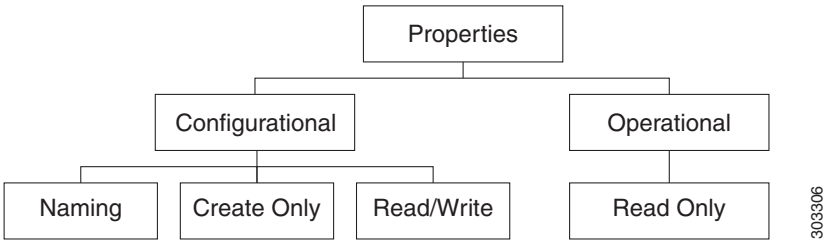
Properties of Managed Objects may be classified as Configuration or Operational.

Configuration properties may be classified as:

- Naming properties: Form part of the Rn. Needs to be specified only during MO creation and cannot be modified later.
- Create-Only properties: May be specified only during MO creation and cannot be modified later. If the property is not specified, a default value is assumed.
- Read / Write properties: May be specified during MO creation and can also be modified subsequently.

Operational properties indicate the current status of the MO / system and are hence read-only.

Figure 4 Properties of Managed Objects



The table below lists the examples of the various property types.

Property Type	Example
Naming	Name in LsServer (Service Profile MO)
Create-Only	Type in LsServer
Read / Write	Description in LsServer
Read-Only	OperState in LsServer

Methods

Methods are Cisco UCS XML APIs used to manage and monitor the system. There are methods supported for:

- Authentication
- AaaLogin
- AaaRefresh

- AaaLogout
- Configuration
- ConfigConfMo(s)
- LsClone
- LsInstantiate*
- FaultAckFaults
- Query
- ConfigResolveDn(s)
- ConfigResolveClass(es)
- ConfigResolveChildren
- Event Monitor
- EventSubscribe

The class query methods (ConfigResolveClass(es), ConfigResolveChildren) allow a filter to be specified so that a specific set of MOs are matched and returned by the method.

The supported filters are:

- Property Filters:

Supported Filter	Definition
allbits	Match if all specified values present in a multi-valued property
anybit	Match if any of the specified values present in a multi-valued property
bw	Match if the property's value lies between the two values specified
eq	Match if property's value is the same as the specified value
ge	Match if property's value is greater than or equal to the specified value
gt	Match if property's value is greater than the specified value
le	Match if property's value is lesser than or equal to the specified value
lt	Match if property's value is lesser than the specified value
ne	Match if property's value is not equal to the specified value
wcard	Match if property's value matches the pattern specified

- Composite Filters (Acts on sub-filters)

Composite Filter	Definition
not	Negates result of sub-filter
and	True, if all the sub-filters return true
or	True, if any of the sub-filters return true

PowerTool Mapping

All but about 30 of the PowerTool cmdlets are generated from the MO specification. A convenient noun is used in place of the type (ServiceProfile instead of LsServer etc.). Get, Add, Set, Remove cmdlets or a subset is generated for the various MO types. All cmdlets support the Xml parameter, which dumps the Xml request and response on the screen.

Add Cmdlet—Uses the ConfigConfMo(s) method with MO status "created" along with the specified property values. If the ModifyPresent parameter is specified, status "created, modified" is specified instead. If the Force parameter is specified, there will be no prompt for confirmation.

Get Cmdlet—Use the ConfigResolveClass method to retrieve MOs. If any property parameters are specified, they are used to generate "eq" filters. If multiple property parameters are specified, the multiple "eq" filters are combined with a "and" filter.

Set Cmdlet—Uses the ConfigConfMo(s) method with MO status "modified" along with the specified property values. If the Force parameter is specified, there will be no prompt for confirmation.

Remove Cmdlet—Uses the ConfigConfMo(s) method with MO status "deleted". If the Force parameter is specified, there will be no prompt for confirmation.

The table below lists the properties that can be specified for a given Verb:

Property	Get	Add	Set
Naming	Yes (Positional)	Yes (Positional	No
Create-Only	Yes	Yes	No
Read-Write	Yes	Yes	Yes
Operational / Read-Only	Yes	No	No

The table below lists the types that can come down the pipeline for corresponding cmdlets:

Verb / Type	Pipeline Input
Get	Singleton – none Non-singleton – Parent Type
Add	Singleton – none Non-singleton – Parent Type
Set	MO has naming property – Same Type MO has no naming property – Same or Parent Type
Remove	Same Type

The table below lists the methods invoked to generate the required XML requests:

Cmdlet	Method
Add-Ucs[1] Set-Ucs1[1]	ConfigConfMos
Get-Ucs[1]	ConfigResolveClass with filters
Get-UcsManagedObject -ClassId	ConfigResolveClass
Get-UcsManagedObject –ClassId -Dnlist	ConfigFindDnsByClassId
Get-UcsManagedObject –Dn	ConfigResolveDns
Connect-Ucs	AaaLogin
Disconnect-Ucs	AaaLogout
Background[1]	AaaRefresh
Copy-UcsServiceProfile	LsClone
Add-UcsServiceProfileFromTemplate	LsInstantiateTemplate, LsInstantiateNTemplate, LsInstantiateNNamedTemplate
Get-UcsChild	ConfigResolveChildren
Acknowledge-UcsFault	FaultAckFaults
Start-UcsKvmSession	AaaGetNComputeAuthTokenByDn
Watch-Ucs	EventSubscribe (First Watcher)

Clear-UcsStatistics	StatsClearInterval
Get-UcsTransactionImpact	ConfigEstimateImpact

1. This is not a cmdlet. It is a background service.

Get-UcsCmdletMeta is useful cmdlet to explore the MO types, the corresponding nouns, supported Verbs, properties of the MOs, the details of properties including the type (Naming, Read/Write etc.) and the version of UCS Manager the property was introduced in etc.

3 Installation

Before You Begin

- Ensure you have PowerShell v2 installed in your system.
- Uninstall all versions of Cisco UCS Power Tool that are older than Cisco UCS PowerTool, Release 0.9.1.0.
- Close any instances of PowerShell running with the PowerTool module loaded.

Installation

Step 1 Download and launch the installer.

Step 2 (Optional) Select **Create Shortcut** to add a shortcut on the desktop.

Getting Started

Step 1 Launch **Cisco UCS PowerTool** from the desktop shortcut.

Step 2 View all cmdlets, functions, and aliases supported by the Cisco UCS PowerTool.

```
Get-Command -Module CiscoUcsPS
Get-Command -Module CiscoUcsPS | group CommandType
Get-Command -Module CiscoUcsPS | measure
```

Step 3 Connect to a Cisco UCS domain.

```
$handle = Connect-Ucs <ip or hostname> -NotDefault
```

Note After logging on, by default, the Cisco UCS handle is added to the default Cisco UCS domain list, unless the `-NotDefault` option is specified. Every cmdlet that operates on a Cisco UCS domain takes the –Ucs parameter, where the handle can be specified.

Step 4 Connect to a Cisco UCS domain using a proxy.

```
$proxy = New-Object System.Net.WebProxy
$proxy.Address = "http:\\<url>:<port>"
$proxy.UseDefaultCredentials =
$false
$proxy.Credentials = New-Object System.Net.NetworkCredential("<user name>", "<password>")
$handle = Connect-Ucs <ip or hostname> -Proxy
$proxy
```

Step 5 Use the following cmdlets:

a. Get Cisco UCS domains consolidated status information.

```
Get-UcsStatus -Ucs $handle
```

b. Get chassis objects.

```
Get-UcsChassis -Ucs $handle
```

c. Get list of service profile Instances.

```
Get-UcsServiceProfile -Ucs $handle -Type instance
```

d. Disconnect.

```
Disconnect-Ucs -Ucs $handle
```

Default UCS

If no handle or name is specified, the Cisco UCS domain handle is added to a DefaultUcs domain list unless the `-Ucs` parameter is specified, the first cmdlet in the pipeline operates on the default Ucs list.

```
Connect-Ucs <ip or hostname>
```

Get the Default Ucs list.

```
Get-UcsPSSession
```

Get UCS's consolidated status information.

```
Get-UcsStatus
```

Get the set of all chassis objects.

```
Get-UcsChassis
```

Get the object pertaining to chassis 1.

```
Get-UcsChassis -Id 1
```

Get the set of blades, pertaining to chassis 1.

```
Get-UcsChassis -Id 1 | Get-UcsBlade
```

Enable HTTP on the FI.

```
Get-UcsHttp | Set-UcsHttp -AdminState enabled
```

Disable HTTP on the FI.

```
Get-UcsHttp | Set-UcsHttp -AdminState disabled
```

Disconnect.

```
Disconnect-Ucs
```

Default UCS List with Multiple UCS

PowerTool cmdlets can work with multiple Cisco UCS domains by specifying multiple handles.

Connect to a Cisco UCS domain.

```
$handle1 = Connect-Ucs <ip1> -NotDefault
$handle2 = Connect-Ucs <ip2> -NotDefault
Get-UcsStatus -Ucs $handle1,$handle2
Disconnect-Ucs -Ucs $handle1,$handle2
```

By default, multiple Cisco UCS handles are not allowed in DefaultUcs. This can be overridden using the `Set-UcsPowerToolConfiguration` cmdlet.

```
Get-UcsPowerToolConfiguration
```

```
Set-UcsPowerToolConfiguration -SupportMultipleDefaultUcs $true

Connect-Ucs <ip1>
Connect-Ucs <ip2>
Get-UcsStatus
Disconnect-Ucs
```

Connect to multiple Cisco UCS domains using the same login credentials.

```
$user = "<username>"
$password = "<password>" |
 ConvertTo-SecureString -AsPlainText -Force
$cred = New-Object System.Management.Automation.PSCredential($user, $password)
$servers = @("<ucs1>", "<ucs2>", "<ucs3>")
Connect-Ucs $servers -Credential $cred
```

Credentials to/from File

```
Connect-Ucs <ip1>
Connect-Ucs <ip2>
```

Credentials can be stored to a file. The stored credentials are encrypted with a specified Key.

```
Export-UcsPSSession -LiteralPath C:\work\labs.xml
Disconnect-Ucs
```

Login can be initiated from credentials stored in a file.

```
Connect-Ucs -LiteralPath C:\work\labs.xml
```

Specify proxy while logging in with credentials stored in a file.

```
$proxy = New-Object System.Net.WebProxy
$proxy.Address = "http:\\<url>:<port>"
$proxy.UseDefaultCredentials = $false
$proxy.Credentials = New-Object System.Net.NetworkCredential("<user name>", "<password>")
Connect-Ucs -LiteralPath C:\work\lab.xml -Proxy $proxy
```

Login to an additional system and add the credentials to the file.

```
Connect-Ucs <ip3>
Export-UcsPSSession -Path C:\work\lab?.xml -Merge
```

SSL Handling

When a user connects to a Cisco UCS server and the server cannot recognize any valid certificates; connection establishment depends on InvalidCertificateAction.InvalidCertificateAction is set to Ignore by default. By default PowerTool is configured to establish the connection without taking into account if the certificate is invalid.

You can override this using the Set-UcsPowerToolConfiguration cmdlet.

```
Get-UcsPowerToolConfiguration
Set-UcsPowerToolConfiguration -InvalidCertificateAction Fail
```

	Description
Fail	The cmdlet will not establish connection if the certificate is not valid.
Ignore	The cmdlet will establish the connection without taking into account that the certificate is invalid.
Default	(Windows default) The cmdlet will establish connection if the certificate is valid.

Register/Unregister Cisco UCS Central

If you want to have Cisco UCS Central manage a Cisco UCS domain, you need to register that domain. When you register, you need to choose the types of policies and other configurations, such as backups and firmware, that will be managed by Cisco UCS Central and which will be managed by Cisco UCS Manager.

Before you register a Cisco UCS domain with Cisco UCS Central, do the following:

Step 1 Configure an NTP server and the correct time zone in both Cisco UCS Manager and Cisco UCS Central to ensure that they are in sync. If the time and date in the Cisco UCS domain and Cisco UCS Central are out of sync, the registration may fail.

Step 2 Obtain the hostname or IP address of Cisco UCS Central

Step 3 Obtain the shared secret that you configured when you deployed Cisco UCS Central

```
$password = "SharedSecret" | ConvertTo-SecureString -AsPlainText -Force
Register-UcsCentral -Name 10.10.10.10 -SharedSecret $password
```

Step 4 Unregister from UCS Central

```
Get-UcsCentral | Unregister-UcsCentral
```

Aliases

Some aliases have been defined for convenience.

```
gal | ? {$_.Name -like "*-Ucs*" } | select Name
Name
----
Acknowledge-UcsBlade
Acknowledge-UcsChassis
Acknowledge-UcsFault
Acknowledge-UcsFex
Acknowledge-UcsRackUnit
Acknowledge-UcsSlot
Associate-UcsServiceProfile
Decommission-UcsBlade
Decommission-UcsChassis
Decommission-UcsFex
Decommission-UcsRackUnit
Disassociate-UcsServiceProfile
Get-UcsMo
Recommission-UcsRackUnit
Recommission-UcsBlade
Recommission-UcsChassis
Recommission-UcsFex
Remove-UcsBlade
Remove-UcsChassis
Remove-UcsFex
Remove-UcsMo
Remove-UcsRackUnit
Sync-UcsMo
```

4 Examples

The examples in this section show how to execute the cmdlets. This following examples are included in this section:

- PowerTool Cmdlet Generation
- Org
- Faults
- Get Cmdlet -Hierarchy Flag

- Get Cmdlet –LimitScope Flag
- Transaction Support
- VLANs (Creation & Deletion)
- MAC Pools & Blocks
- Server Pools
- UUID Suffix Pools & Blocks
- WWNN Pools & Blocks
- WWPN Pools & Blocks
- IQN Suffix Pools & Blocks
- Port Roles
- Port Channel
- VLANs (Assignment)
- Blade Power & Temperature Statistics
- Configuration Backup
- Import Configuration
- Managed Object Synchronization
- Monitoring UCS Managed Object Transitions
- Tech Support
- Service Profile
- Service Profile Components
- Service Profile Association
- Filters
- iSCSI Boot
- vNIC Template
- vHBA Template
- Boot Policy
- Adapter Policy
- BIOS Policy
- Host Firmware Package
- IPMI Access Profile
- Management Firmware Package
- Power Control Policy
- Server Pool Policy QualificationsDynamic vNIC connection Policy
- Network Control Policy
- Privileges
- User Roles
- Locales
- User Accounts
- Remote Authentication – RADIUS
- Remote Authentication – TACACS
- Remote Authentication – LDAP
- RADIUS Provider
- TACACS Provider

- LDAP Provider
- Authentication Domains
- Communication Services
- Communication Services - Telnet
- Communication Services - CIM XML
- Communication Services - SNMP
- Communication Services – HTTP
- Communication Services - HTTPS
- Generic Managed Object Queries
- Generic Managed Object Cmdlets
- Generic Cmdlet –XmlTag
- XtraProperty in Get/Add/Set cmdlets
- CCO Integration
- Upload Firmware
- Export to XML
- Import from XML
- KVM
- UCS Statistics
- Transaction Impact
- Cmdlet Meta Information
- Compare-UcsManagedObject - Dn Translation
- Compare-UcsManagedObject – GetPropertyDiff()
- Add Cmdlet –ModifyPresent Flag

PowerTool Cmdlet Generation

Generate cmdlets for the specified actions in UCS GUI.

```
ConvertTo-UcsCmdlet
```

Get xml request along with generated cmdlets.

```
ConvertTo-UcsCmdlet -Verbose
```

Generate cmdlets for action in the specified GUI log.

```
ConvertTo-UcsCmdlet -GuiLog -LiteralPath 'C:\Work\centrale_7128.log.1'
ConvertTo-UcsCmdlet -GuiLog -Path 'C:\Work\centrale_71*.log.?'
```

Generate cmdlets for the specified xml request.

```
ConvertTo-UcsCmdlet -Xml -Request '<lsClone dn="org-root/ls-sp1" inTargetOrg="org-root" inServerName="sp2"
inHierarchical="false"></lsClone>'
```

Generate cmdlets for the specified xml requests in file.

```
ConvertTo-UcsCmdlet -Xml -LiteralPath 'C:\Work\config.xml'
```

The Cisco UCS Manager GUI considers a few XML snippets as secure and does not log them. So, the ConvertTo- UcsCmdlet does not find the logs to do the translation.

The workaround is as follows:

Step 1 Save the launch link as a file, for example: https://<ip_or_hostname>/ucsm/ucsm.jnlp.

Step 2 Right click the file and open the file with **Notepad**.

Step 3 Add the following line after the other property definitions:

```
<property name="log.show.encrypted" value="true"/>
```

Step 4 Save and close the file.

Step 5 Right click the file and open the file with Java™ Web Start Launcher.

Step 6 After the UI launches, from a PowerShell window with the CiscoUCS module loaded, run `ConvertTo-UcsCmdlet`.

Org

Get a list of orgs across Cisco UCS domains, in the Default UCS list.

```
Get-UcsOrg | select Ucs, Name, Dn
```

Get a handle to the root level Org.

```
Get-UcsOrg -Level root
```

Add 5 orgs to UCS.

```
1..5 | % { Add-UcsOrg -Ucs <handle or name> qwerty$_ }
```

Faults

Retrieve faults, group them by severity.

```
Get-UcsFault | group Severity
```

Retrieve critical faults.

```
Get-UcsFault -Severity critical | select Ucs, Dn, Cause
```

Acknowledge all unacknowledged faults.

```
Get-UcsFault | ? {$_.Ack -eq "no" } | Acknowledge-UcsFault
```

Get Cmdlet -Hierarchy Flag

Get Managed Object including all children.

```
Get-UcsServiceProfile –Name sp_name –Hierarchy
```

Get Cmdlet –LimitScope Flag

Get service profile at the root level without descending into org root children.

```
Get-UcsServiceProfile -Name sp_name –LimitScope
```

Get service profile from org Finance without descending into org Finance children.

```
Get-UcsOrg –Name Finance | Get-UcsServiceProfile -Name sp_name –LimitScope
```

Get VLAN from LanCloud.

```
Get-UcsLanCloud | Get-UcsVlan –LimitScope
```

Transaction Support

Start a transaction.

```
Start-UcsTransaction
```

Perform an operation.

```
...
```

End a transaction.

```
Complete-UcsTransaction
```

Undo a transaction.

```
Undo-UcsTransaction
```

VLANs (Creation & Deletion)

VLANs in Cisco UCS domains are referred to by name and VLAN definitions can be created under four nodes in the MIT.

	Description
LanCloud	This is a global VLAN and is applicable to both FIs.
FabricLanCloud	This is a Fabric Specific VLAN and is applicable to either Fabric A or Fabric B.
ApplianceCloud	This is a global VLAN and is applicable to both FIs.
FabricApplianceCloud	This is a fabric specific VLAN applicable to either Fabric A or Fabric B, used during configuration of Appliance Ports.

Create a VLAN under the Global LAN Cloud.

```
Get-UcsLanCloud | Add-UcsVlan -Name lan_cloud_vlan -Id 500
```

Create a VLAN under Fabric A LAN Cloud.

```
Get-UcsFiLanCloud -Id A | Add-UcsVlan -Name fi_a_vlan -Id 500
```

Create a VLAN under Fabric B LAN Cloud.

```
Get-UcsFiLanCloud -Id B | Add-UcsVlan -Name fi_b_vlan -Id 500
```

Create a VLAN under the Global Appliance Cloud.

```
Get-UcsApplianceCloud | Add-UcsVlan -Name appliance_vlan -Id 500
```

Create a VLAN under the Fabric A Appliance Cloud.

```
Get-UcsFabricApplianceCloud -Id A | Add-UcsVlan -Name fi_a_appliance_vlan -Id 500
```

Create a VLAN under the Fabric B Appliance Cloud.

```
Get-UcsFabricApplianceCloud -Id B | Add-UcsVlan -Name fi_b_appliance_vlan -Id 500
```

Import a list of VLANs from a csv file and create the VLANs under the LAN cloud. (This example creates the csv file as well.)

Create VLANs on 1 device

```
$("Name,Id";foreach ($vlan in 501..550) { "vlan${vlan},${vlan}" }) > C:\work\Demo\vlans.csv
$lc=(Get-UcsLanCloud)
$lc | Get-UcsVlan | select Ucs, Name, Id
Start-UcsTransaction
import-csv C:\work\Demo\vlans.csv | % {$lc | Add-UcsVlan -Name $_.Name -Id $_.Id } Complete-UcsTransaction
$lc | Get-UcsVlan | select Ucs, Name, Id
```

Remove the added VLANs.

```
$lc | Get-UcsVlan | ? {$_.Id -ge 501 -and $_.Id -le 550 }  | Remove-UcsVlan -Force
```

MAC Pools & Blocks

Add a MAC Block to the default MAC Pool.

Get-UcsMacPool -Ucs <handle or name> default | Add-UcsMacMemberBlock 00:25:B5:00:A0:00 00:25:B5:00:A0:08

Check for any clashes in MAC Pool assignments across all Cisco UCS domains in default list.

Get-UcsMacPoolAddr | group Id | where {$_.Count -ne 1 } | select -ExpandProperty Group | select Ucs, Id, Assigned, AssignedToDn

Server Pools

Create a server pool.

$server_pool = Add-UcsServerPool -Name server_pool

Add Blade 1/4 to the server pool.

$server_pool | Add-UcsComputePooledSlot -ChassisId 1 -SlotId 4

Add Rack 1 to the server pool.

$server_pool | Add-UcsComputePooledRackUnit -Id 1

Remove Server pool.

$server_pool | Remove-UcsServerPool

UUID Suffix Pools & Blocks

Create a UUID Suffix pool.

$uuid_pool = Add-UcsUuidSuffixPool -Name uuid_pool -Prefix 3864EB9A-89A2-11DF

Add a block of UUID Suffixes to the suffix pool.

$uuid_pool | Add-UcsUuidSuffixBlock -From 0000-000000000001 -To 0000-00000000002C

Remove a UUID Suffix pool.

$uuid_pool | Remove-UcsUuidSuffixPool

WWNN Pools & Blocks

Get all WWNN pool in UCS.

Get-UcsWwnPool -Purpose node-wwn-assignment

Create a WWNN pool.

$wwnn_pool = Add-UcsWwnPool -Name wwnn_pool -Purpose node-wwn-assignment

Add a WWN block to the WWNN pool.

$wwnn_pool | Add-UcsWwnMemberBlock -From 20:00:00:24:B5:00:00:00 -To 20:00:00:24:B5:00:00:09

Add a WWNN initiator to the WWNN pool.

$wwnn_pool | Add-UcsWwnInitiator -Id 20:00:00:25:B5:00:00:2C -Name wwnn_initiator

Remove a WWNN initiator.

$wwnn_pool | Get-UcsWwnInitiator -Id 20:00:00:25:B5:00:00:2C | Remove-UcsWwnInitiator

Remove a WWNN pool.

$wwnn_pool | Remove-UcsWwnPool

WWPN Pools & Blocks

Get all WWPN pool in UCS. Get-UcsWwnPool -Purpose port-wwn-assignment

Create a WWPN pool.

$wwpn_pool = Add-UcsWwnPool -Name wwpn_pool -Purpose port-wwn-assignment

Add a WWN block to the WWPN pool.

$wwpn_pool | Add-UcsWwnMemberBlock -From 20:00:00:24:B5:00:00:00 -To 20:00:00:24:B5:00:00:09

Add a WWPN initiator to the WWPN pool.

$wwpn_pool | Add-UcsWwnInitiator -Id 20:00:00:25:B5:00:00:2D -Name wwpn_initiator

Set descr for WWPN initiator.

$wwpn_pool | Get-UcsWwnInitiator -Id 20:00:00:25:B5:00:00:2D | Set-UcsWwnInitiator -Descr “WWPN initiator modified”

Remove a WWPN pool.

$wwpn_pool | Remove-UcsWwnPool

IQN Suffix Pools & Blocks

Get IQN pool in UCS.

Get-UcsIqnPoolPool -Name iqnSuffixPool

Create IQN pool.

$iqn_pool = Get-UcsOrg -Level root | Add-UcsIqnPoolPool -Name iqn_pool -Prefix I

Create IQN pool block.

$iqn_pool_block =

$iqn_pool | Add-UcsIqnPoolBlock -Suffix B -From 0 -To 10

Remove IQN pool block.

$iqn_pool_block | Remove-UcsIqnPoolBlock

Remove IQN pool.

$iqn_pool | Remove-UcsIqnPoolPool

Port Roles

Make Fabric A's Slot 1 (Fixed Ports Slot) Port 19 a server port.

Get-UcsFabricServerCloud -Id A | Add-UcsServerPort -PortId 19 -SlotId 1

Unconfigure Fabric A's Slot 1 (Fixed Ports Slot) Port 19 from being a server port.

Get-UcsFabricServerCloud -Id A | Get-UcsServerPort -PortId 19 -SlotId 1 | Remove-UcsServerPort -Force

Make Fabric A's Slot 1 (Fixed Ports Slot) Port 15 an appliance port.

Get-UcsFabricApplianceCloud -Id A | Add-UcsAppliancePort -PortId 15 -SlotId 1

Unconfigure Fabric A's Slot 1 (Fixed Ports Slot) Port 15 from being an appliance port.

Get-UcsFabricApplianceCloud -Id A | Get-UcsAppliancePort -PortId 15 -SlotId 1 | Remove-UcsAppliancePort –Force

Make Fabric A's Slot 1 (Fixed Ports Slot) Port 16 an uplink port.

Get-UcsFiLanCloud -Id A | Add-UcsUplinkPort -PortId 16 -SlotId 1

Unconfigure Fabric A's Slot 1 (Fixed Ports Slot) Port 16 from being an uplink port.

Get-UcsFiLanCloud -Id A | Get-UcsUplinkPort -PortId 16 -SlotId 1 | Remove-UcsUplinkPort –Force

Port Channel

Create an Appliance Port Channel on Fabric A with ports 19 & 20.

```
$ap_pc = Get-UcsFabricApplianceCloud -Id A | Add-UcsAppliancePortChannel -PortId 55
$ap_pc | Add-UcsAppliancePortChannelMember -SlotId 1 -PortId 19
$ap_pc | Add-UcsAppliancePortChannelMember -SlotId 1 -PortId 20
```

Add Port Channel to the Appliance VLAN.

```
Get-UcsApplianceCloud | Get-UcsVlan -Name ApplianceVlan | Add-UcsVlanMemberPortChannel -SwitchId A -PortId
$ap_pc.PortId
```

Remove Port Channel from Appliance VLAN.

```
Get-UcsApplianceCloud | Get-UcsVlan -Name ApplianceVlan | Get-UcsVlanMemberPortChannel -SwitchId A -PortId 55 |
Remove-UcsVlanMemberPortChannel -Force
```

Remove Appliance Port Channel.

```
Get-UcsFabricApplianceCloud -id A | Get-UcsAppliancePortChannel -PortId 55 | Remove-UcsAppliancePortChannel
-Force
```

VLANs (Assignment)

Add Appliance port A/1/15 to an Appliance VLAN.

```
Get-UcsApplianceCloud | Get-UcsVlan -name ApplianceVlan | Add-UcsVlanMemberPort -SwitchId A -SlotId 1 -PortId 15
```

Remove Appliance port A/1/15 from Appliance VLAN.

```
Get-UcsApplianceCloud | Get-UcsVlan -name ApplianceVlan | Get-UcsVlanMemberPort -SwitchId A -SlotId 1 -PortId 15
| Remove-UcsVlanMemberPort -Force
```

Blade Power & Temperature Statistics

View Power Statistics of all blades.

```
Get-UcsBlade | Get-UcsComputeBoard | Get-UcsComputeMbPowerStats | Out-GridView
```

View Temperature Statistics of all blades.

```
Get-UcsBlade | Get-UcsComputeBoard | Get-UcsComputeMbTempStats| Out-GridView
```

Configuration Backup

Remove any previously stored backups in UCS.

```
Get-UcsMgmtBackup | Remove-UcsMgmtBackup
```

The PathPattern can be auto-filled, allowing the cmdlet to be used with multiple Cisco UCS domains. Create and download full-state system backup of UCS. This creates a binary file that includes a snapshot of the entire system. You can use the file generated from this backup to restore the system during disaster recovery. This file can restore or rebuild the configuration on the original fabric interconnect, or recreate the configuration on a different fabric interconnect. You cannot use this file for an import.

```
Backup-Ucs -Type full-state -PathPattern 'C:\Backups\${ucs}-${yyyy}${MM}${dd}-${HH}${mm}-full-state.tar.gz'
```

Create and download logical backup of UCS. This creates an XML file that includes all logical configuration settings such as service profiles, VLANs, VSANs, pools, and policies. You can use the file generated from this backup to import these configuration settings to the original fabric interconnect or to a different fabric interconnect. You cannot use this file for a system restore.

```
Backup-Ucs -Type config-logical -PathPattern 'C:\Backups\${ucs}-${yyyy}${MM}${dd}-${HH}${mm}-config-logical.xml'
```

Create and download system backup of UCS. This creates an XML file that includes all system configuration settings such as usernames, roles, and locales. You can use the file generated from this backup to import these configuration settings to the original fabric interconnect or to a different fabric interconnect. You cannot use this file for a system restore.

```
Backup-Ucs -Type config-system -PathPattern 'C:\Backups\${ucs}-${yyyy}${MM}${dd}-${HH}${mm}-config-system.xml'
```

Create and download config-all backup of UCS. This creates an XML file that includes all system and logical configuration settings. You can use the file generated from this backup to import these configuration settings to the original fabric interconnect or to a different fabric interconnect. You cannot use this file for a system restore. This file does not include passwords for locally authenticated users.

```
Backup-Ucs -Type config-all -PathPattern 'C:\Backups\${ucs}-${yyyy}${MM}${dd}-${HH}${mm}-config-all.xml'
```

Import Configuration

The import function is available for all configuration, system configuration, and logical configuration files. You can perform an import while the system is up and running.

Import all configuration xml (An XML file that includes all system and logical configuration settings. The current configuration information is replaced with the information in the imported configuration file one object at a time.

```
Import-UcsBackup -LiteralPath 'C:\Backups\config-all.xml'
```

Import all configuration xml. The information in the imported configuration file is #compared with the existing configuration information. If there are conflicts, the import operation overwrites the information on the Cisco UCS domain with the information in the import configuration file.

```
Import-UcsBackup -LiteralPath 'C:\Backups\config-all.xml' -Merge
```

Managed Object Synchronization

Sync a set of MOs from SYSA to SYSB.

```
Sync-UcsManagedobject -Ucs SYSB (Compare-UcsManagedObject (Get-UcsOrg -ucs SYSB) (Get-UcsOrg -ucs SYSA)) -whatif
Sync-UcsManagedobject -Ucs SYSB (Compare-UcsManagedObject (Get-UcsOrg -ucs SYSB) (Get-UcsOrg -ucs SYSA)) -Force
```

Sync a set of MOs from SYSA to all systems in the default list.

```
Get-UcsPSSession | % {Sync-UcsManagedobject -Ucs $_ (Compare-UcsManagedObject (Get-UcsOrg -ucs $_) (Get-UcsOrg
-ucs SYSA)) -Force}
```

Monitoring UCS Managed Object Transitions

Watch Cisco UCS domains for all events for 60 seconds.

```
Watch-Ucs -TimeoutSec 60
```

Watch Cisco UCS domains for any changes in faults for 60 seconds.

```
Watch-Ucs -TimeoutSec 60 -ClassId FaultInst
```

Watch UCS for a particular field in MO to attain a success value.

```
Send-UcsFirmware -LiteralPath C:\work\Images\ucs-k9-bundle-b-series.1.4.2b.B.bin | Watch-Ucs -Property
TransferState -SuccessValue downloaded -PollSec 30 -TimeoutSec 600
```

Tech Support

Technical support data for the entire UCSM instance will be created and downloaded to the specified file.

```
Get-UcsTechSupport -PathPattern 'C:\${ucs}-techsupp-ucsm.tar' -UcsManager -RemoveFromUcs -TimeoutSec 600
```

Technical support data for the UCSM management services(excluding fabric interconnects) will be created and downloaded to the specified file.

```
Get-UcsTechSupport -PathPattern 'C:\${ucs}-techsupp-ucsmmgmt.tar' -UcsMgmt   -RemoveFromUcs -TimeoutSec 600
```

Technical support data for Chassis id 1 and Cimc id 1 will be created and downloaded to specified file.

```
Get-UcsTechSupport -PathPattern 'C:\${ucs}-techsupp-chassis.tar' -RemoveFromUcs -TimeoutSec 600 -ChassisId 1
-CimcId 1
```

Technical support data for Chassis id 1 and Iom id 1 will be created and downloaded to specified file.

```
Get-UcsTechSupport -PathPattern 'C:\${ucs}-techsupp-iom.tar' -RemoveFromUcs -TimeoutSec 600 -ChassisId 1 -IomId
1
```

Technical support data for RackServer id 1 and RackAdapter id 1 will be created and downloaded to specified file.

```
Get-UcsTechSupport -PathPattern 'C:\${ucs}-techsupp-rack.tar' -RemoveFromUcs -TimeoutSec 600 -RackServerId 1
-RackAdapterId 1
```

Technical support data for FEX id 1 will be created and downloaded to specified file.

```
Get-UcsTechSupport -PathPattern 'C:\${ucs}-techsupp-fex.tar' -RemoveFromUcs -TimeoutSec 600 -FexId 1
```

Service Profile

Get all service profile instances in UCS.

```
Get-UcsServiceProfile –Type instance
```

Get all service profile updating templates in UCS.

```
Get-UcsServiceProfile -Type updating-template
```

Get all service profile initial templates in UCS.

```
Get-UcsServiceProfile -Type initial-template
```

Add a new service profile sp_name from service profile template sp_template

```
Add-UcsServiceProfile -SrcTemplName sp_template –Name sp_name
```

Add a service profile.

```
Add-UcsServiceProfile -Name sp_name -BootPolicyName boot_policy -BiosProfileName bios_policy  -HostFwPolicyName
1.4-3i -MgmtFwPolicyName 1.4-3i -MaintPolicyName maint_policy -MgmtAccessPolicyName ipmi_policy -PowerPolicyName
power_policy -ScrubPolicyName scrub_policy -SolPolicyName sol_policy -StatsPolicyName stats_policy
-AgentPolicyName agent_policy -DynamicConPolicyName vNIC_policy -ExtIPState static -IdentPoolName UUID_pool
-LocalDiskPolicyName  disk_policy -Uuid "00000000-0000-0000-0000-000000000008" -UsrLbl "serviceprofile"
```

Get power state of a service profile.

```
Get-UcsServiceProfile -Name sp_name | Get-UcsServerPower
```

Bind service profile to a template.

```
Get-UcsServiceProfile -Name sp_name | Set-UcsServiceProfile -SrcTemplName sp_template
```

Remove a service profile.

```
Get-UcsServiceProfile -Name sp_name | Remove-UcsServiceProfile
```

Service Profile Components

Create a service profile.

```
$sp = Add-UcsServiceProfile -Name sp_name
```

Create a vNIC with reference to QoS Policy.

```
$eth0 = $sp | Add-UcsVnic -Name eth0 -QosPolicyName qos_policy
```

Add a VLAN for vNIC, make it Native VLAN.

```
$eth0 | Add-UcsVnicInterface -Name fi_a_vlan -DefaultNet true
```

Add a VLAN for vNIC.

```
$eth0 | Add-UcsVnicInterface -Name fi_b_vlan
```

Create a vNIC and instantiate from template.

```
$sp | Add-UcsVnic -Name eth1 -NwTemplName vnic_template
```

Create a vHBA.

```
$fc0 = $sp | Add-UcsVhba -Name fc0 -IdentPoolName wwpn_pool
```

Add a VSAN for vHBA.

```
$fc0 | Set-UcsVhbaInterface -Name fi_b_vsan
```

Add a WWNN pool.

```
$sp | Add-UcsVnicFcNode -IdentPoolName node_default
```

Service Profile Association

Associate a service profile to a blade.

```
Get-UcsServiceProfile sp_name -LimitScope | Associate-UcsServiceProfile -Blade (Get-UcsBlade -Chassis 1 -SlotId 1)
```

Associate service profile to a rack.

```
Get-UcsServiceProfile sp_name -LimitScope | Associate-UcsServiceProfile -RackUnit (Get-UcsRackUnit -ServerId 1)
```

Associate service profile to a server pool.

```
Get-UcsServiceProfile sp_name -LimitScope | Associate-UcsServiceProfile -ServerPoolName FileServerPool
```

Associate a service profile to a server pool along with server pool qualification policy.

```
Get-UcsServiceProfile sp_name -LimitScope | Associate-UcsServiceProfile -ServerPoolName file_server_pool
-ServerPoolQualificationPolicyName file_server_pool
```

Disassociate a service profile.

```
Get-UcsServiceProfile sp_name -LimitScope | Disassociate-UcsServiceProfile
```

Create a copy of a service profile.

```
Get-UcsServiceProfile -Name sp_name -LimitScope | Copy-UcsServiceProfile -NewName copy_sp_name
```

Rename a service profile.

```
Get-UcsServiceProfile -Name sp_name | Rename-UcsServiceProfile -NewName rename_sp_name
```

Filters

#Get all Service Profile Templates.

```
Get-UcsServiceProfile -Filter 'Type -clike *-template' | select Ucs,Dn,Name
```

Get all Service Profiles with Name containing string 'SJC'.

```
Get-UcsServiceProfile -Filter 'Name -cmatch SJC' | select Ucs, Dn, Name
```

Get all Service Profiles with Name beginning with string 'SJC'.

```
Get-UcsServiceProfile -Filter 'Name -clike SJC' | select Ucs, Dn, Name
```

Get all VLANs with Id between 8 and 50.

```
Get-UcsVlan -Filter 'Id -cbw 8,50' | select Ucs,Dn, Name, Id
```

Get all Roles that have the fault privilege.

```
Get-UcsRole -Filter 'Priv -ccontains fault' | select Ucs, Dn, Name
```

Get all Roles that have the fault or operations privilege.

```
Get-UcsRole -Filter 'Priv -canybit fault,operations' | select Ucs, Dn, Name
```

Get all Roles that have the fault and operations privilege.

```
Get-UcsRole -Filter 'Priv -callbits fault,operations' | select Ucs, Dn, Name
```

Get a list of blades/rack units with temperature greater than 45.

```
Get-UcsProcessorEnvStats -Filter 'Temperature -cgt 45'| Get-UcsParent | Get-UcsParent | Get-UcsParent | select Ucs, Dn
```

Get a list of faults generated between 4/18/2012 9:00 and 4/19/2012 9:30.

```
Get-UcsFault -Filter 'Created -cbw "4/18/2012 9:00","4/19/2012 9:30"' | select Ucs, Cause, Dn, Created
```

Get Service Profiles with Name equals 'SJC'.

```
Get-UcsServiceProfile -Filter 'Name -ceq SJC' | select Ucs, Dn, Name
```

Get all Service Profiles with Name equals 'SJC/sjc/SjC' etc.

```
Get-UcsServiceProfile -Filter 'Name -ieq sjc' | select Ucs, Dn, Name
```

Get all Service Profiles with Name beginning with string 'SJC/sjc/SjC' etc.

```
Get-UcsServiceProfile -Filter 'Name -ilike SJC*' | select Ucs, Dn, Name
```

Get all Service Profiles with Name except 'SJC/sjc/SjC' etc.

```
Get-UcsServiceProfile -Filter 'Name -ine SJC' | select Ucs, Dn, Name
```

iSCSI Boot

Start Ucs Transaction.

```
Start-UcsTransaction
```

Create a service profile.

```
$sp = Add-UcsServiceProfile -Type instance -Name iscsiboot
```

Add a static IP þ(not related to iSCSI boot).

```
$staticIp = Add-UcsVnicIpV4StaticAddr -ServiceProfile $sp -Addr 10.65.224.161 -DefGw 10.65.224.1 -Subnet
255.255.255.0
```

Create the required vNIC and add VLAN.

```
$vnic = Add-UcsVnic -ServiceProfile $sp -Name enic1 -SwitchId A -Addr 00:25:B5:07:80:00 $vlan605 =
Add-UcsVnicInterface -Vnic $vnic -Name vlan605 -DefaultNet yes
```

Create iSCSI vNIC and map it to the vNIC created above.

Add iSCSI Initiator Parameters – VLAN and IP address.

```
$enic = Add-UcsVnicIScsi -ServiceProfile $sp -Name iscsienic1 -InitiatorName iqn.1995-05.com.broadcom.iscsiboot
-VnicName enic1 $vlan = Add-UcsVnicVlan -VlanName vlan605 -VnicIScsi $enic $ipv4if = Add-UcsVnicIPv4If -VnicVlan
$vlan $ipv4iscsi = Add-UcsVnicIPv4IscsiAddrþ -VnicIPv4If $ipv4if -Addr 10.65.224.157
```

Add target parameters.

```
$primaryTarget = Add-UcsVnicIScsiStaticTargetIf -VnicVlanþ $vlan -IpAddress 10.65.224.16 -Name
iqn.1992-08.com.netapp:sn.135037408 -Priority 1 $primaryLun = Add-UcsVnicLunþ -VnicIScsiStaticTargetIf
$primaryTarget -Id 2
```

Create a specific boot policy.

```
$bootPolicy = Add-UcsBootDefinition -ServiceProfile $sp
```

If installation is required, create a LsbootVirtualMedia.

```
$vmedia = Add-UcsLsbootVirtualMedia -BootDefinition $bootPolicy -Access read-only -Order 1
```

Add iSCSI enic in the boot path.

```
$iscsiBoot = Add-UcsLsbootIScsi -BootDefinition $bootPolicy -Order 2 $iscsiBootImagePath =
Add-UcsLsbootIScsiImagePath -LsbootIscsi $iscsiBoot -Type primary -ISCSIVnicName iscsienic1
```

Complete Ucs Transaction.

```
Complete-UcsTransaction | Out-null
```

vNIC Template

Create an Initial vNIC template.

```
$vnic_init_temp = Add-UcsVnicTemplate -Name vnic_init_temp -TemplType initial-template -SwitchId A
```

Create an Updating vNIC template.

```
$vnic_update_temp = Add-UcsVnicTemplate -Name vnic_update_temp -TemplType updating-template -SwitchId B -Target
adaptor
```

Add a VLAN to an Initial vNIC template.

```
$vnic_init_temp | Add-UcsVnicInterface -Name fi_a_vlan
```

Add a VLAN to an Initial vNIC template and make it Native VLAN.

```
$vnic_init_temp | Add-UcsVnicInterface -Name lan_cloud_vlan -DefaultNet true
```

Set MAC Pool, Network Control policy and QoS policy for Initial vNIC template.

```
$vnic_init_temp | Set-UcsVnicTemplate -IdentPoolName mac_pool -NwCtrlPolicyName network_policy -QosPolicyName
qos_policy
```

Remove an Initial vNIC template.

```
$vnic_init_temp | Remove-UcsVnicTemplate
```

vHBA Template

Create an Initial vHBA template.

```
$vhba_init_temp = Add-UcsVhbaTemplate -Name vhba_init_temp -TemplType initial-template -SwitchId A
```

Create an Updating vHBA template.

```
$vhba_update_temp = Add-UcsVhbaTemplate -Name vhba_update_temp -TemplType updating-template -SwitchId B
```

Add a VSAN to Updating vHBA template.

```
$vhba_update_temp | Add-UcsVhbaInterface -Name fi_b_vsan
```

Set WWN Pool, QoS policy, Pin Group and Stats policy for Updating vHBA template.

```
$vhba_update_temp | Set-UcsVhbaTemplate -IdentPoolName wwpn_pool -QosPolicyName qos_policy -PinToGroupName
san_pin_group -StatsPolicyName threshold_policy
```

Remove an Updating vHBA template.

```
$vhba_update_temp | Remove-UcsVhbatemplate
```

Boot Policy

Create a Boot policy and enable Reboot on boot order change and enforce vNIC/vHBA/iSCSI name.

```
$boot_policy = Add-UcsBootPolicy -Name boot_policy -EnforceVnicName yes -RebootOnUpdate yes
```

Add a Floppy.

```
$boot_policy | Add-UcsLsbootVirtualMedia -Order 3 -Access read-write
```

Add a CD-ROM.

```
$boot_policy | Add-UcsLsbootVirtualMedia -Order 2 -Access read-only
```

Add a Local Disk.

```
$boot_storage = $boot_policy | Add-UcsLsbootStorage -Order 1 $boot_storage | Add-UcsLsbootLocalStorage
```

Add a SAN boot.

```
$boot_storage | Add-UcsLsbootSanImage -VnicName fc0 -Type primary | Add-UcsLsbootSanImagePath -Type primary -Lun
1 -Wwn 20:00:00:25:B5:00:00:00
```

Add a LAN boot.

```
$boot_policy | Add-UcsLsbootLan -Order 4 | Add-UcsLsbootLanImagePath -VnicName eth0 -Type primary
```

Remove a Boot policy.

```
$boot_policy | Remove-UcsBootPolicy
```

Adapter Policy

Add a custom Eth Adapter policy, that disables receive checksum offload.

```
$eth_adap_policy = Add-UcsEthAdapterPolicy -Name eth_adap_policy -Descr "Custom Adapter Policy” $eth_adap_policy
| Set-UcsEthAdapterOffloadProfile -TcpRxChecksum disabled
```

Add a FC Adapter policy.

```
$fc_adap_policy = Add-UcsFcAdapterPolicy -Name fc_adap_policy -Descr "Fibre Channel Adapter Policy"
```

Enable FCP error recovery for Fibre Channel Adapter policy.

```
$fc_adap_policy | Set-UcsAdaptorFcErrorRecoveryProfile -FcpErrorRecovery enabled
```

Add an iSCSI Adapter policy.

```
$iscsi_adap_policy = Add-UcsIScsiAdapterPolicy -Name iscsi_policy
```

Enable TCP timestamp, HBA mode and Boot to target for iSCSI Adapter policy.

```
$iscsi_adap_policy | Set-UcsIScsiAdapterPolicyProperties -BootToTarget yes -TcpTimeStamp yes -HbaMode yes
```

BIOS Policy

Create a Bios policy and enable reboot on Bios setting change.

```
$bios_policy = Add-UcsBiosPolicy -Name bios_policy -RebootOnUpdate yes
```

Modify USB system idle power optimizing setting to high-performance.

```
$bios_policy | Set-UcsBiosVfUSBSystemIdlePowerOptimizingSetting -VpUSBIdlePowerOptimizing high-performance
```

Enable Virtulization technology.

```
$bios_policy | Set-UcsBiosVfIntelVirtualizationTechnology -VpIntelVirtualizationTechnology enabled
```

Enable quite boot for Bios policy.

```
$bios_policy | Set-UcsBiosVfQuietBoot -VpQuietBoot enabled
```

Resume Ac on power loss to last-state.

```
$bios_policy | Set-UcsBiosVfResumeOnACPowerLoss -VpResumeOnACPowerLoss last-state
```

Disable boot option retry.

```
$bios_policy | Set-UcsBiosVfBootOptionRetry -VpBootOptionRetry disabled
```

Disable console redirection.

```
$bios_policy | Set-UcsBiosVfConsoleRedirection -VpConsoleRedirection disabled
```

Remove a Bios policy.

```
$bios_policy | Remove-UcsBiosPolicy
```

Host Firmware Package

Create a Host Firmware package and set IgnoreCompCheck to No.

```
$host_firm_pack = Add-UcsFirmwareComputeHostPack -Name host_firm_pack -IgnoreCompCheck no
```

Add a Host Firmware pack item.

```
$host_firm_pack | Add-UcsFirmwarePackItem -Type adaptor -HwModel N20-AC0002 -HwVendor "Cisco Systems Inc"
-Version '1.4(1i)'
```

Set version of Host Firmware pack item.

```
$host_firm_pack | Get-UcsFirmwarePackItem -HwModel N20-AC0002 | Set-UcsFirmwarePackItem -Version '2.0(1t)'
```

Remove Host Firmware package.

```
$host_firm_pack | Remove-UcsFirmwareComputeHostPack
```

IPMI Access Profile

Create an IPMI Access profile.

```
$ipmi_profile = Add-UcsIpmiAccessProfile -Name ipmi_profile
```

Add an IPMI user with Administrator's role.

```
$ipmi_profile | Add-UcsAaaEpUser -Name ipmiUser -Priv admin
```

Modify role for IPMI user.

```
$ipmi_profile | Get-UcsAaaEpUser -Name ipmiUser | Set-UcsAaaEpUser -Priv readonly
```

Remove an IPMI Access profile.

```
$ipmi_profile | Remove-UcsIpmiAccessProfile
```

Management Firmware Package

Create a Management Firmware package and set IgnoreCompCheck to no.

```
$mgmt_firm_pack = Add-UcsFirmwareComputeMgmtPack -Name mgmt_firm_pack -IgnoreCompCheck no
```

Add a Management Firmware pack item.

```
$mgmt_firm_pack | Add-UcsFirmwarePackItem -Type blade-controller -HwModel "N20-B6620-1" -HwVendor "Cisco Systems
Inc" -Version '1.4(1i)'
```

Set version of Management Firmware pack item.

```
$mgmt_firm_pack | Get-UcsFirmwarePackItem -HwModel N20-B6620-1 | Set-UcsFirmwarePackItem -Version '2.0(1t)'
```

Remove Management Firmware package.

```
$mgmt_firm_pack | Remove-UcsFirmwareComputeMgmtPack
```

Power Control Policy

Create a Power Control policy. Priority is ranked on a scale of 1-10, where 1 indicates the highest priority and 10 indicates lowest priority. The default priority is 5.

```
$power_policy = Add-UcsPowerPolicy -Name power_policy -Prio 6
```

Create a Power Control policy with power capping 'no-cap'.

```
$power_nocap = Add-UcsPowerPolicy -Name power_nocap -Prio no-cap
```

Remove Power Control policy.

```
$power_policy | Remove-UcsPowerPolicy
```

Server Pool Policy Qualifications

Create a Server Pool policy qualification.

```
$server_pool_qual = Add-UcsServerPoolQualification -Name server_pool_qual
```

Create an Adaptor qualification.

```
$server_pool_qual | Add-UcsAdaptorQualification
```

Create a Memory qualification policy with memory clock speed of 1067Mhz and 16 memory units.

```
$server_pool_qual | Add-UcsMemoryQualification -Clock 1067 -Units 16
```

Create a CPU/Cores qualification policy with Pentium_4 processor architecture

```
$server_pool_qual | Add-UcsCpuQualification -Arch Pentium_4
```

Create a Diskless Storage qualification policy for servers without a local disk (SAN only configuration).

```
$server_pool_qual | Add-UcsStorageQualification -Diskless yes
```

Create a Rack qualification.

```
$server_pool_qual | Add-UcsRackQualification -MaxId 1 -MinId 1[1]
```

Remove a Server Pool policy qualification.

```
$server_pool_qual | Remove-UcsServerPoolQualification
```

Dynamic vNIC connection Policy

Create a Dynamic vNIC connection policy dy_vnic_conn with 54 dynamic vNICs and protection enabled for failover mode.

```
$dy_vnic_conn = Add-UcsDynamicVnicConnPolicy -Name dy_vnic_conn -AdaptorProfileName Linux -DynamicEth 54
-Protection protected
```

Remove a Dynamic vNIC connection policy.

```
$dy_vnic_conn | Remove-UcsDynamicVnicConnPolicy
```

Network Control Policy

Create a Network Control policy network_policy with CDP enabled and VIF configured to change the operational state of a vNIC to down when uplink connectivity is lost on the fabric interconnect.

```
$network_policy = Get-UcsOrg -Level root | Add-UcsNetworkControlPolicy -Name network_policy -Cdp enabled
-UplinkFailAction link-down
```

Set Mac security for Network Control policy to allow forged MAC addresses.

```
$network_policy | Set-UcsPortSecurityConfig -Forge allow
```

Set Mac security for Network Control policy so that after the first packet has been sent to the fabric interconnect, all other packets must use the same MAC address or they will be silently rejected by the fabric interconnect. This enables port security for the associated vNIC.

```
$network_policy | Set-UcsPortSecurityConfig -Forge deny
```

Remove a Network Control policy.

```
$network_policy | Remove-UcsNetworkControlPolicy
```

Privileges

List out all privileges on the UCS.

```
Get-UcsPrivilege
```

User Roles

Add a user role “test_role” with admin privileges.

```
Add-UcsRole -Name user_role -Priv admin
```

Change privileges for a user role to allow read-and-write access to fabric interconnect infrastructure, network security operations and read access to the rest of the system.

```
Get-UcsRole -Name user_role | Set-UcsRole -Priv ls-network
```

Set multiple privileges using Set-UcsRole.

```
Get-UcsRole -Name multi_priv_role | Set-UcsRole -Priv "ls-network", "ls-qos"
```

Get all user roles in UCS.

```
Get-UcsRole
```

Get a user role by name.

```
Get-UcsRole -Name multi_priv_role
```

Remove a user role.

```
Get-UcsRole -Name multi_priv_role | Remove-UcsRole
```

Locales

Add a Locale.

```
Add-UcsLocale -Name asia_pacific -Descr "Locale for Asia Pacific users"
```

Get all locales.

```
Get-UcsLocale
```

Add an org to a locale.

```
Get-UcsLocale -Name asia_pacific | Add-UcsAaaOrg -Name org_asia_pacific -OrgDn org-root/org-Finance
Remove a locale.
Get-UcsLocale -Name asia_pacific | Remove-UcsLocale
```

User Accounts

Add a local user.

```
$user = Add-UcsLocalUser -Name jdoe -Pwd Passw0rdJdoe
```

Edit a local user.

```
$user | Set-UcsLocalUser -FirstName John -Lastname Doe
```

Add to a user.

```
$user | Add-UcsUserRole -Name user_role
```

Remove a local user.

```
Get-UcsLocalUser -Name jdoe | Remove-UcsLocalUser
```

Remote Authentication – RADIUS

Set global configuration for RADIUS authentication.

```
Set-UcsRadiusGlobalConfig -Descr "RADIUS authentication configuration" -Timeout 20 -Retries 3 -Force
```

Create a RADIUS server instance with server key “test1234” and maximum 2 retries.

```
Add-UcsRadiusProvider -Name "192.168.23.84" -Key test1234 -Retries 2
```

Set RADIUS as default authentication.

```
Set-UcsDefaultAuth -Realm radius
```

Remote Authentication – TACACS

Set global configuration for TACACS authentication.

```
Set-UcsTacacsGlobalConfig -Descr "TACACS authentication configuration" -Timeout 20 -Retries 3
```

Add a TACACS Provider.

```
Add-UcsTacacsProvider -Name "192.168.23.84" -Key test1234
```

Set TACACS as default authentication.

```
Get-UcsNativeAuth | Set-UcsNativeAuth -DefLogin tacacs
```

Remote Authentication – LDAP

Set global configuration for LDAP authentication.

```
Set-UcsLdapGlobalConfig -Descr 'LDAP authentication configuration' -Timeout 20 -Retries 3 -Force
```

Add a LDAP Provider.

```
add-UcsLdapProvider -Attribute 'CiscoAVPair' -Basedn 'CN=Users,DC=qasamlab,DC=com' -FilterValue 'cn=$userid'
-Key 'Bbv03515' -Name '10.193.23.84' -Rootdn 'CN=Administrator,CN=Users,DC=qasamlab,DC=com'
```

Set LDAP as default authentication.

```
Get-UcsNativeAuth | Set-UcsNativeAuth -DefLogin ldap
```

RADIUS Provider

Create a RADIUS server instance with server key “test1234” and maximum 2 retries.

```
Add-UcsRadiusProvider -Name "192.168.23.84" -Key test1234 -Retries 2
```

Add a RADIUS provider group and set it as the default remote authentication.

```
Get-UcsRadiusGlobalConfig | Add-UcsProviderGroup -Name radiusprovidergroup1 Get-UcsProviderGroup -Name
radiusprovidergroup1 | Add-UcsProviderReference -Name "192.168.23.84"
Get-UcsNativeAuth | Set-UcsNativeAuth -DefLogin radius
Get-UcsDefaultAuth | Set-UcsDefaultAuth -ProviderGroup radiusprovidergroup1
```

TACACS Provider

Add a TACACS provider.

```
Add-UcsTacacsProvider -Name "192.168.23.84" -Key test1234
```

Add a TACACS provider group and set it as the default remote authentication.

```
Get-UcsTacacsGlobalConfig | Add-UcsProviderGroup -Name tacacsprovidergroup1
Get-UcsProviderGroup -Name tacacsprovidergroup1 | Add-UcsProviderReference -Name "192.168.23.84"
Get-UcsNativeAuth | Set-UcsNativeAuth -DefLogin tacacs
Get-UcsDefaultAuth | Set-UcsDefaultAuth -ProviderGroup tacacsprovidergroup1
```

LDAP Provider

Add an LDAP provider.

```
add-UcsLdapProvider -Attribute 'CiscoAVPair' -Basedn
'CN=Users,DC=qasamlab,DC=com' -FilterValue 'cn=$userid' -Key 'Bbv03515' -Name '192.168.23.84' -Rootdn
'CN=Administrator,CN=Users,DC=qasamlab,DC=com'
```

Add an LDAP provider group and set it as the default remote authentication.

```
Get-UcsLdapGlobalConfig | Add-UcsProviderGroup -Name ldapprovidergroup1
Get-UcsProviderGroup -Name ldapprovidergroup1  | Add-UcsProviderReference -Name "192.168.23.84"
Get-UcsNativeAuth | Set-UcsNativeAuth -DefLogin ldap
Get-UcsDefaultAuth | Set-UcsDefaultAuth -ProviderGroup ldapprovidergroup1
```

Authentication Domains

Authentication Domains configure simultaneous support for different authentication methods (local, TACACS+, RADIUS, and LDAP/Active Directory) and provider groups.

Configure a TACAS Provider Group with a TACACS Provider.

```
$tp = Add-UcsTacacsProvider -Name "192.168.23.84" -Key test1234
$tpg = Get-UcsTacacsGlobalConfig | Add-UcsProviderGroup -Name tacacs_pg $tpg | Add-UcsProviderReference -Name
$tp.Name
```

Create an Authentication Domain and add a reference to the TACACS Provider group.

```
$ad = Add-UcsAuthDomain -Name adtacacs
$ad | Get-UcsAuthDomainDefaultAuth | þSet-UcsAuthDomainDefaultAuth -Realm
tacacs-ProviderGroup tacacs_pg
```

Now if a user logs in from the console, GUI or XML API with the user name being "ucs-adtacacs\user" the TACACS configuration created above will be used for authentication.

Communication Services

Get UCS Web Session Limits, which define the maximum number of concurrent web sessions (both GUI and xml) permitted access to the system at any one time.

```
Get-UcsWebSessionLimit
```

Set Web Session Limit for the user to 30 and an overall session limit of 255.

```
Set-UcsWebSessionLimit -SessionsPerUser 30 -TotalSessions 255
```

Communication Services - Telnet

Get UCS telnet configuration.

```
Get-UcsTelnet
```

Allow telnet connections.

```
Set-UcsTelnet -AdminState enabled -Descr "Telnet configuration for UCS"
```

Communication Services - CIM XML

Get UCS CIM XML configuration.

```
Get-UcsCimXml
```

Enable the CIM XML service.

```
Set-UcsCimXml -AdminState enabled
```

Communication Services - SNMP

Get UCS SNMP configuration.

```
Get-UcsSnmp
```

Enable SNMP with community string being "public", system contact person being "CiscoSystems" and location of the host being "Bangalore".

```
Set-UcsSnmp -Descr "SNMP config for UCS" -AdminState enabled -SysContact CiscoSystems -SysLocation Bangalore
-Community public
```

Get a UCS SNMP user.

```
Get-UcsSnmpUser
```

Add a UCS SNMP user.

```
Add-UcsSnmpUser -Name joe -Auth md5 -Privpwd Joe@Cisco -Pwd Joe@Cisco -UseAes true
```

Set a UCS SNMP user.

```
Get-UcsSnmpUser -Name joe | Set-UcsSnmpUser -Auth sha -UseAes false
```

Remove a UCS SNMP user.

```
Get-UcsSnmpUser -Name joe | Remove-UcsSnmpUser
```

Get a UCS SNMP trap.

```
Get-UcsSnmpTrap
```

Add a UCS SNMP trap.

```
Add-UcsSnmpTrap -Hostname 168.65.120.32 -Community public -NotificationType traps -Port 162 -V3Privilege noauth
-Version v3
```

Set UCS SNMP configuration.

```
Get-UcsSnmpTrap -Hostname 168.65.120.32 | Set-UcsSnmpTrap -Community public -NotificationType informs -Port 162
-V3Privilege auth -Version v1
```

Remove UCS SNMP configuration.

```
Get-UcsSnmpTrap -Hostname 168.65.120.32 | Remove-UcsSnmpTrap
```

Communication Services – HTTP

Get UCS http configuration.

```
Get-UcsHttp
```

Set UCS http configuration to enable http and enable http to https redirection.

```
Set-UcsHttp -AdminState enabled -RedirectState enabled
```

Communication Services - HTTPS

Get UCS https configuration.

```
Get-UcsHttps
```

Create a keyring with a key size of 1024 bits.

```
Add-UcsKeyring -Name keyring1024 -Modulus mod1024
```

Create a certificate request passing the required subject name(hostname of the machine).

```
Get-UcsKeyRing -Name keyring2048 | Add-UcsCertRequest -SubjName savbu-pti01
```

Get the certificate for the generated certificate request and have it installed on the client machine. Verify it by running "certmgr.msc"

Add a Trust Point.

```
Add-UcsTrustPoint -Name TPkeyring1024
```

Set a certificate chain for TP.

```
Get-UcsTrustPoint -Name TPkeyring1024 | Set-UcsTrustPoint -CertChain "
-----BEGIN CERTIFICATE----- MIIEoDCCA4igAwIBAgIQMjE/6XYi/a9CU8PPgR20ZDANBgkqhkiG9w0BAQUFADBU
MRIwEAYKCZImiZPyLGQBGRYCaW4xGTAXBgoJkiaJk/IsZAEZFglxYXNhbS1sYWIx
FDASBgoJkiaJk/IsZAEZFgR1Y3NtMQ0wCwYDVQQDEwRVQ1NNMB4XDTEwMDcxNjEy
MzM1N1oXDTE1MDcxNjEyNDIzNFowVDESMBAGCgmSJomT8ixkARkWAmluMRkwFwYK
CZImiZPyLGQBGRYJcWFzYW0tbGFiMRQwEgYKCZImiZPyLGQBGRYEdWNzbTENMAsG
A1UEAxMEVUNTTTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAOh2Cgcm
EVzdGCMf8FQy4SpLgeDXAn8DbobDdKbcH7txYRUMPCRmktYeEjV1QhfMPu1hAs5B
cDCbAG0wN7InoGexsNQVhdAQpY7S18h0iml/GiR9XWbhcfaanbxDXUBepOve07UU
6kDnVwxGh9uQrgAgrI5oPatbbiE4zbjUlD2WYjZQ3UH+UGOP+Ub3OcaL+OHteHQh
dQWt/EuAprJeUp4jVjZwiaTbC8URAedMy8DjzP3WsbxMS+CHtF/TZ/dHBt+Z3ptK
syomrXro2/Kv0HWl9o921ryXHnz133sSDmFJ//LVbvZLqD2PM2UzZuX/+4C5S+44
Hghlv1uiNQ3yRDcCAwEAAaOCAWwwggFoMAsGA1UdDwQEAwIBhjAPBgNVHRMBAf8E
BTADAQH/MB0GA1UdDgQWBBRG3l1HsV1u/dVTpUmIc9MNs4r/+DCCARUGA1UdHwSC
AQwwggEIMIIBBKCCAQCggf2GgbxsZGFwOi8vL0NOPVVDU00sQ049YmxyLXNhbS1x
YS1hYWExLENOPUNEUCxDTj1QdWJsaWMlMjBLZXklMjBTZXJ2aWNlcyxDTj1TZXJ2
```

```
aWNlcyxDTj1Db25maWd1cmF0aW9uLERDPXVjc20sREM9cWFzYW0tbGFiLERDPWlu
P2NlcnRpZmljYXRlUmV2b2NhdGlvbkxpc3Q/YmFzZT9vYmplY3RDbGFzcz1jUkxE
aXN0cmlidXRpb25Qb2ludIY8aHR0cDovL2Jsci1zYW0tcWEtYWFhMS51Y3NtLnFh
c2FtLWxhYi5pbi9DZXJ0RW5yb2xsL1VDU00uY3JsMBAGCSsGAQQBgjcVAQQDAgEA
MA0GCSqGSIb3DQEBBQUAA4IBAQBeuIZYIeI07ZhXa1PjCs/YeBdR+S7+i0GKDYJq
nLtyWAua8YMyJQ57vJFB0I5MbEmHPt2JaKmFGRSYTMfLH4l7Z7vQUsPkaW5hlkWk
zQ4/VQusHEasioazFHbfSDPVzA9IRd71TNHGp5ruVoaThQJUouavcnYSp5FFeOCM
xQcFUtGTkl/1XHoRv8ROwHjv24YXLPpxC+7DwMtmKLS00MGP8su9+nf4OrLGB2Ml
0cVhfAqwliMoVTfg6uzkI6xcss3xI1y7tuFOBZ60CkBvD+1C7ZhYe212RN75Uo6Z
jL77g422uodkMO5TSqj6pbI/wJmIQMsS45NDitoM90x7TpvZ -----END CERTIFICATE-----"
```


Set a TP and certificate for a key ring.


```
Get-UcsKeyRing keyring1024  | Set-UcsKeyRing -Tp TPkeyring1024 -Cert "
 -----BEGIN CERTIFICATE----- MIIFnzCCBIegAwIBAgIKRy4WzAAAAAAADTANBgkqhkiG9w0BAQUFADBUMRIwEAYK
CZImiZPyLGQBGRYCaW4xGTAXBgoJkiaJk/IsZAEZFglxYXNhbS1sYWIxFDASBgoJ
kiaJk/IsZAEZFgR1Y3NtMQ0wCwYDVQQDEwRVQ1NNMB4XDTExMTIwMTEzMzYwOVoX
DTEyMTIwMTEzNDYwOVowFjEUMBIGA1UEAxMLc2F2YnUtdHBpMDEwggEiMA0GCSqG
SIb3DQEBAQUAA4IBDwAwggEKAoIBAQC4eSJyX6J/I1ZSwSFXu+NmEW0BE0I0EEX/
zpMJ/yxh/SJKsgybicPAr0SRzgDKRhEIoIsMSMXigTFpErMgF4tkT32HNUeLlb5M
N+e/lcx3M7ogQfDWUOMBFVP9qMCTkpn7cPAnOEoYaCx4J79XQJ6RyX1+uI1qAiCh
tz1jPWnTvzNGTacp/opZYwtJ0f5iY6ERNQ8WKJke56oulzUhcq40y3oKX/i1GfkI
IG8GT26Yv6a+KPKdRDSO+q+GZSqkmIcghETPYThCt3CWDO7AYxRyQtNnGDzN1OEd
YaCQhcbzoD8qfogpnsWIMARzgYC2HWAN9suZ0zO3NGrFKkeg6ep7AgMBAAGjggKv
MIICqzAfBgNVHREBAf8EFTATggtzYXZidS10cGkwMYcECkF4JTAdBgNVHQ4EFgQU
ns86LcentpqeJmT814OjcfYt2DQwHwYDVR0jBBgwFoAURt5dR7Fdbv3VU6VJiHPT
DbOK//gwggEVBgNVHR8EggEMMIIBCDCCAQSgggEAoIH9hoG8bGRhcDovLy9DTj1V
Q1NNLENOPWJsci1zYW0tcWEtYWFhMSxDTj1DRFAsQ049UHVibGljJTIwS2V5JTIw
U2VydmljZXMsQ049U2VydmljZXMsQ049Q29uZmlndXJhdGlvbixEQz11Y3NtLERD
PXFhc2FtLWxhYixEQz1pbj9jZXJ0aWZpY2F0ZVJldm9jYXRpb25MaXN0P2Jhc2U/
b2JqZWN0Q2xhc3M9Y1JMRGlzdHJpYnV0aW9uUG9pbnSGPGh0dHA6Ly9ibHItc2Ft
LXFhLWFhYTEudWNzbS5xYXNhbS1sYWIuaW4vQ2VydEVucm9sbC9VQ1NNLmNybDCC
AS0GCCsGAQUFBwEBBIIBHzCCARswgawGCCsGAQUFBzAChoGfbGRhcDovLy9DTj1V
Q1NNLENOPUFJQSxDTj1QdWJsaWMlMjBLZXklMjBTZXJ2aWNlcyxDTj1TZXJ2aWNl
cyxDTj1Db25maWd1cmF0aW9uLERDPXVjc20sREM9cWFzYW0tbGFiLERDPWluP2NB
Q2VydGlmaWNhdGU/YmFzZT9vYmplY3RDbGFzcz1jZXJ0aWZpY2F0aW9uQXV0aG9y
aXR5MGoGCCsGAQUFBzAChl5odHRwOi8vYmxyLXNhbS1xYS1hYWExLnVjc20ucWFz
YW0tbGFiLmluL0NlcnRFbnJvbGwvYmxyLXNhbS1xYS1hYWExLnVjc20ucWFzYW0t
bGFiLmluX1VDU00uY3J0MA0GCSqGSIb3DQEBBQUAA4IBAQB01hNPBrDqfu9hrI1E
o6Y9GghHNZ4cxwPlhz0U9w4iskWNVHlw7IJdf7U+WUvulGWcyln73i2r2sOeQqy3
Isx/2dKS4n3YX7x1hYpMubPCCL1fHIPqQwh9ddlHyKFtxqMd6/jQJyhLNOX5yz4h
HpORfl4xGGWYsv1Jjqqr2jREbV3kE/uOq0NNi+2efWS0YHq9SESKqu1cXgMl5LyC
ZKQYolUseboYK90XgLc2yww+75UcgynLZRxgbAPstNeqPTWh12kWogrO4zkpo18Y
Vz2yB2BA6/ugCbtJuIw352HzzHU9FM4Y7R0r9k75CNjA9wScu56hX2rfIFUwnSMT gWvg -----END CERTIFICATE----- "
```


Set the keyring for https.

```
Get-UcsHttps | Set-UcsHttps -KeyRing keyring1024 -AdminState enabled
```

Access UCSM through https should now give no "untrusted connection" message.

Generic Managed Object Queries

Get Managed Object of a specific DN.

```
Get-UcsManagedObject -Dn "sys/chassis-1"
```

Get all Managed Objects of a particular class.

```
Get-UcsManagedObject -ClassId faultInst
```

Get DNs of Managed Objects of a particular class.

```
Get-UcsManagedObject -ClassId faultInst -DnList
```

Get names of all Service Profiles from org-root.

```
Get-UcsOrg -Level root | Get-UcsManagedObject -ClassId lsServer | Select Name
```

Get immediate children of org-root.

```
Get-UcsOrg -Level root | Get-UcsChild
```

Get parent of a Managed Object.

```
Get-UcsOrg -Name Finance | Get-UcsParent
```

Generic Managed Object Cmdlets

Create a VLAN using parent object.

```
$propMap = @{Name = "lan_cloud_vlan"; Id = 500} Get-UcsLanCloud | Add-UcsManagedObject -ClassId FabricVlan
-PropertyMap $propMap
```

Create a VLAN using parent object, modify if already existing.

```
$propMap = @{Name = "lan_cloud_vlan"; Id = 500} Add-UcsManagedObject -ClassId FabricVlan -PropertyMap $propMap
-Parent (Get-UcsLanCloud) -ModifyPresent
```

Create a VLAN using DN.

```
$propMap = @{Dn = "fabric/lan/net-lan_cloud_vlan"; Name = "lan_cloud_vlan"; Id = 500} Add-UcsManagedObject
-ClassId FabricVlan -PropertyMap $propMap
```

Modify a VLAN using Managed Object.

```
$vlan = Get-UcsVlan -Name 'lan_cloud_vlan' $propMap = @{DefaultNet = "yes"; Id = 501; Sharing = "primary"}
Set-UcsManagedObject -PropertyMap $propMap -ManagedObject $vlan
```

Modify a VLAN using DN.

```
$propMap = @{Dn = "fabric/lan/net-lan_cloud_vlan"; DefaultNet = "yes"; Id = 501; Sharing = "primary"}
Set-UcsManagedObject -PropertyMap $propMap -ClassId FabricVlan
```

Remove a Managed Object.

```
Get-UcsOrg -Name Finance | Remove-UcsManagedObject
```

Generic Cmdlet –XmlTag

The XmlTag parameter enables us to work with unknown Managed Objects.

Create a multicast policy.

```
Add-UcsManagedObject -XmlTag fabricMulticastPolicy -PropertyMap @{Dn="org-root/mc-policy-multicastpolicy";
Name="multicastpolicy";   PolicyOwner="local"; SnoopingState="enabled"; QuerierState="disabled";}
```

Set snooping state to disabled for multicast policy.

```
Set-UcsManagedObject -XmlTag fabricMulticastPolicy -PropertyMap @{Dn = "org-root/mc-policy-multicastpolicy";
SnoopingState="disabled";}
```

XtraProperty in Get/Add/Set cmdlets

The XtraProperty parameter ensures that unknown Managed Object properties can also be used in Get/Add/Set cmdlets.

Create a service profile with extra property ExtIPPoolName.

```
Add-UcsServiceProfile -Name sp_name -XtraProperty @{ExtIPPoolName = "ext-mgmt";}
```

Get all service profiles that have ext-mgmt as ExtIPPoolName.

```
Get-UcsServiceProfile -XtraProperty @{ExtIPPoolName = "ext-mgmt";}
```

CCO Integration

```
There are 2 Cmdlets related to CCO Image handling:
```

Get a List of Images from CCO.

```
$images = Get-UcsCcoImageList
```

Select 2.0(1x) images and download the images. Get-UcsCcoImage first checks if the image is available locally. If the image exists and md5sum matches, no download is initiated. If not, the image is downloaded.

```
$images | where { $_.ImageName -like "ucs-k9-bundle*2.0.1*" } | Get-UcsCcoImage -Path C:\work\Images
```

Re-running the command should not initiate any downloads, if the previous download was successful.

```
$images | where { $_.ImageName -like "ucs-k9-bundle*2.0.1*" } | Get-UcsCcoImage -Path C:\work\Images
```

A proxy can be used if required.

```
$proxy = New-Object System.Net.WebProxy $proxy.Address = "http:\\<url>:<port>" $proxy.UseDefaultCredentials =
$false $proxy.Credentials = New-Object System.Net.NetworkCredential("<username>", "<password>") $images =
Get-UcsCcoImageList -Proxy $proxy
```

Upload Firmware

Upload an image to the Default Ucs system.

```
Send-UcsFirmware -LiteralPath C:\work\Images\ucs-k9-bundle-b-series.2.0.1q.B.bin
```

Export to XML

Export the configuration of a Managed Object.

This cmdlet exports the configuration of the managed object and the entire hierarchy.

```
Export-UcsXml -Dn org-root/org-Finance -Hierarchy -LiteralPath C:\cmd.xml
```

Export the xml of a Managed Object into a file.

```
Get-UcsServiceProfile -Name sp_name | Export-UcsMoXml | Out-File c:\mo.xml
```

Import from XML

Import the configuration from the XML file.

```
Import-UcsXml -LiteralPath C:\cmd.xml
```

Import xml of a Managed Object and convert it into objects.

```
Import-UcsMoXml -LiteralPath c:\mo.xml
```

KVM

Start a KVM session for service profile and add a customized title for the KVM window.

```
Get-UcsServiceProfile -Name sp_name -LimitScope | Start-UcsKvmSession -FrameTitle "Custom Frame Title & | 1 2 3"
```

Start a KVM session for blade 1.

```
Start-UcsKvmSession -Blade (Get-UcsBlade -SlotId 1 –ChassisId 1)
```

Start a KVM session for rack unit 1.

```
Start-UcsKvmSession -RackUnit (Get-UcsRackUnit -Id 1)
```

UCS Statistics

Get current Ucs statistics for Chassis Id 1 and Slot Id 1.

```
Get-UcsBlade -ChassisId 1 -SlotId 1 | Get-UcsStatistics –Current
```

Get Ucs statistics history for Chassis Id 1 and Slot Id 1.

```
Get-UcsBlade -ChassisId 1 -SlotId 1 | Get-UcsStatistics –History
```

Clear Ucs statistics.

```
Get-UcsManagedObject -Dn sys/chassis-1/blade-1/board/temp-stats | Clear-UcsStatistics
```

Transaction Impact

Get-UcsTransactionImpact cmdlet estimates the impact of a pending transaction. The cmdlet uses the ConfigEstimateImpact method and returns a UcsImpact object. A message that is similar to the message provided by UCS Manager GUI is provided as part of the UcsImpact object.

Start a transaction.

```
Start-UcsTransaction
```

Create a service profile.

```
$sp = Add-UcsServiceProfile -Name sp_name
```

Create a vNIC.

```
$eth0 = $sp | Add-UcsVnic -Name eth0 -IdentPoolName empty_pool
```

Add a VLAN for vNIC, make it Native VLAN.

```
$eth0 | Add-UcsVnicInterface -Name primary -DefaultNet true
```

Get Transaction Impact.

```
Get-UcsTransactionImpact
```

One can now observe that a UcsImpact object was returned, which indicates a config-failure for the service profile that would be created etc.

Cmdlet Meta Information

Get Meta information about all Managed Object mapped cmdlets.

```
Get-UcsCmdletMeta
```

Get Meta information about LsServer mapped cmdlets.

```
Get-UcsCmdletMeta -ClassId LsServer
```

View the hierarchy of the ServiceProfile (LsServer) class.

```
Get-UcsCmdletMeta -Noun UcsServiceProfile -Tree
```

Get Meta information for the UcsServiceProfile noun.

```
Get-UcsCmdletMeta -Noun UcsServiceProfile
```

See the Managed Object information for LsServer.

```
Get-UcsCmdletMeta -ClassId LsServer | Select -ExpandProperty MoMeta
```

See the Managed Object property information for LsServer.

```
Get-UcsCmdletMeta -ClassId LsServer | Select -ExpandProperty MoMeta | Select -ExpandProperty PropertyMeta
```

Compare-UcsManagedObject - Dn Translation

Create a service profile under org A. Assume that orgs A & B are in place already.

```
$org = Get-UcsOrg -Name A -LimitScope
$destOrg = Get-UcsOrg -Name B -LimitScope
$sp = Add-UcsServiceProfile -Org $org -Name abc
```

Create a translation map with DNs of entities that needs to be translated.

```
$xlateDn = @{ }
$xlateDn['org-root/org-A/ls-abc'] = 'org-root/org-B/ls-xyz'
```

Combine the translation map with Compare-UcsMo to see the changes required.

```
Compare-UcsManagedObject (Get-UcsServiceProfile  -Org $destOrg -Name xyz -LimitScope) (Get-UcsServiceProfile
-Org $org -Name abc -LimitScope ) -XlateMap $xlateDn
```

Combine the translation org with Compare to see the changes required.

```
Compare-UcsManagedObject (Get-UcsServiceProfile -Org $destOrg -Name xyz -LimitScope) (Get-UcsServiceProfile -Org
$org -Name abc -LimitScope ) -XlateOrg org-root/org-B
```

Sync a service profile from org A to org B while renaming it.

```
Sync-UcsManagedObject (Compare-UcsManagedObject (Get-UcsServiceProfile  -Org $destOrg -Name xyz -LimitScope)
(Get-UcsServiceProfile -Org $org -Name abc -LimitScope ) -XlateMap $xlateDn) -Force | select Dn
```

Compare-UcsManagedObject – GetPropertyDiff()

Use GetPropertyDiff() function on output of Compare- UcsManagedObject to see the difference in properties.

```
$sp1 = Get-UcsServiceProfile -Dn org-root/ls-abc
$sp2 = $sp1 | Set-UcsServiceProfile -Descr 'GetPropertyDiff Example' -Force
$diff = Compare-UcsManagedObject $sp1 $sp2
```

Display all the properties having difference. If $diff is an array of objects, then GetPropertyDiff works on $diff[<index>]

```
$diff.GetPropertyDiff()
```

For a specific property $diff.

```
GetPropertyDiff('descr')
```

Include all operational properties of MOs in comparison

```
Compare-UcsManagedObject $sp1 $sp2 -IncludeOperational
```

Add Cmdlet –ModifyPresent Flag

The ModifyPresent option ensures that the add-cmdlets modify the MO, if it already exists, instead of returning an error.

Create a csv file with Name, Id pairs.

```
$("Name,Id"; foreach ($vlan in 501..510) { "vlan${vlan}, ${vlan}" } ) | Out-File c:\vlans.csv
```

Import the Name, Vlan pairs from the file and create those vlans.

```
$lc = Get-UcsLanCloud
Start-UcsTransaction
Import-Csv C:\vlans.csv | % { $lc | Add-UcsVlan -ModifyPresent -Name $_.Name -Id $_.Id }
Complete-UcsTransaction
```

Edit the csv file to edit the ids or add new vlans. Re-running the same Add-UcsVlan snippet above will result in an error, if existing VLANs created again (with or without changes). Invoking Add-UcsVlanþwith the ModifyPresent option, addresses this by modifying the VLANs instead, if they already exist.

```
$lc = Get-UcsLanCloud
Start-UcsTransaction
Import-Csv C:\vlans.csv | % { $lc | Add-UcsVlan -ModifyPresent -Name $_.Name -Id $_.Id }
Complete-UcsTransaction
```

5 Samples

You will find samples such as the following packaged with the installation.

Fetch all Global Policies

```
$id=(Get-UcsPowerToolConfiguration).InstallDir
& "${id}\Samples\Get-UcsGlobalPolicy.ps1" bgl-abcd18
```

6 Related Cisco UCS Documentation

For more information, you can access related documents from the following links:

- *Cisco UCS Documentation Roadmap*
- *Release Bundle Contents for Cisco UCS Software, Release 2.1*

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation:

http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

Subscribe to the *What's New in Cisco Product Documentation* as an RSS feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service. Cisco currently supports RSS Version 2.0.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: www.cisco.com/go/trademarks. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

© 2012 Cisco Systems, Inc. All rights reserved.