Managing Cisco UCS with the Python SDK

David Soper
Technical Marketing Engineer

DEVWKS-2060

cisco Live!
Barcelona | January 27–31, 2020

Agenda

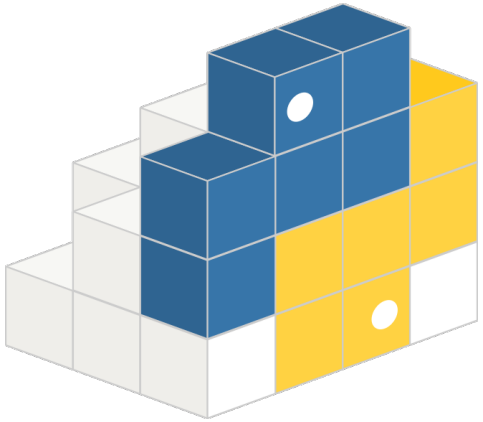
- Introduction – Managing Cisco UCS with the Python SDK
- Installation Overview
- Connect / Query / Filter / Dump XML
- Get Help / Variable Inspection / Metadata
- Configure / Transactions
- Compare / Sync / Code Generation
- Conclusion
- Code – https://github.com/dsoper2/DEVWKS-2060

UCS Python SDK

Cisco UCS Python SDKs

- Hosted on Github
  - UCS Manager SDK
    - Source – https://github.com/CiscoUcs/ucsmsdk
    - Samples – https://github.com/CiscoUcs/ucsmsdk_sample
    - Documents – https://CiscoUcs.github.io/ucsmsdk_docs
  - UCS IMC SDK
    - Source – https://github.com/CiscoUcs/imcsdk
    - Documents – https://ciscoucs.github.io/imcsdk_docs
  - UCS Central Python SDK
    - Source – https://github.com/CiscoUcs/ucscentralsdk

Cisco UCS Python SDKs - Install



- Install Python 2.7.X or 3.7.X OR BOTH!
  - pip – Preferred Installer Program (package manager)
    - Installs latest “Release” from https://pypi.python.org/pypi

      pip install ucsmsdk
  - git –clone the lab repo
    - Latest example source code from https://github.com/dsoper2/DEVWKS-2060

      git clone https://github.com/dsoper2/DEVWKS-2060.git

      cd DEVWKS-2060
  - Repo uses public UCSPEs by default, if needed change UCSM ips:
    - sed -i -e 's/13.58.22.56/198.18.133.91/g' *.py
    - sed -i -e 's/18.217.19.216/198.18.134.249/g' *.py

Connect
Query
Filter
Dump XML

Connect / Query / Dump XML

ex-01.py

DEVWKS-2060 ▸ ex-01.py ▸ ...

```
1   """
2   ex-01.py
3   """
4
5   from ucsmsdk.ucshandle import UcsHandle
6   handle = UcsHandle("198.18.133.91","admin","password")
7   handle.login()
8
9   handle.cookie
10
11  blades = handle.query_classid("ComputeBlade")
12  len(blades)
13
14  for blade in blades:
15      print blade.dn, blade.serial, blade.model
16
17  handle.set_dump_xml()
18  blades = handle.query_classid("ComputeBlade")
19  handle.unset_dump_xml()
```

Query - Filter

ex-02.py

DEVWKS-2060 ▸ ex-02.py ▸ ...

```
1  """
2      ex-02.py
3  """
4
5  filter_exp='(model,"UCSB-B200-M4")'
6  blades = handle.query_classid("ComputeBlade",filter_str=filter_exp)
7  len(blades)
8
9  filter_exp='(model,"UCSB-B200-M4", type="eq")'
10 blades = handle.query_classid("ComputeBlade",filter_str=filter_exp)
11 len(blades)
12
13 filter_exp='(model,"UCSB-B200-M4", type="ne")'
14 blades = handle.query_classid("ComputeBlade",filter_str=filter_exp)
15 len(blades)
16
17 filter_exp='(model,"ucsB-B200-m4", flag="I")'
18 blades = handle.query_classid("ComputeBlade",filter_str=filter_exp)
19 len(blades)
20
21 for blade in blades:
22     print blade.dn, blade.model
```

Query - Returns

ex-03.py

DEVWKS-2060 ▸ ex-03.py ▸ ...

```
"""
    ex-03.py
"""

# List of Objects Returned
blades = handle.query_classid("ComputeBlade")

# Single Object Returned
blade_by_dn = handle.query_dn("sys/chassis-1/blade-2")
print blade_by_dn

# Dictionary of Object Lists Returned
blades_and_chassis = handle.query_classids("ComputeBlade","EquipmentChassis")
print blades_and_chassis

print blades_and_chassis['ComputeBlade']
for blade in blades_and_chassis['ComputeBlade']:
    print blade.dn

# Dictionary of Objects Returned
blade_and_chassis = handle.query_dns("sys/chassis-1/blade-1","sys/chassis-2")
print blade_and_chassis
print blade_and_chassis['sys/chassis-3/blade-1'].dn
```

Get Help
Variable Inspection
Metadata

Get Help / Variable Inspection / Metadata

ex-04.py

DEVWKS-2060 ▸ ex-04.py ▸ ...

```
"""
ex-04.py
"""

vars(handle)
dir(UcsHandle)
help(UcsHandle)

from ucsmsdk.ucscoreutils import get_meta_info
meta = get_meta_info(class_id="FabricVlan")
print meta

meta = get_meta_info(class_id="FabricVlan", include_prop=False, show_tree=False)
print meta

meta = get_meta_info(class_id="FabricVlan", include_prop=False, show_tree=True)
print meta
```


Configure Transactions

Configure

ex-05.py

DEVWKS-2060 ▸ ex-05.py ▸ ...

```
1  """
2      ex-05.py
3  """
4
5  from ucsmsdk.ucshandle import UcsHandle
6  from ucsmsdk.mometa.fabric.FabricVlan import FabricVlan
7  handle = UcsHandle("198.18.133.91", "admin", "password")
8  handle.login()
9
10 fabric_lan_cloud = handle.query_classid("FabricLanCloud")
11 vlan = FabricVlan(parent_mo_or_dn=fabric_lan_cloud[0],
12         name="vlan100",
13         id="100")
14
15 handle.add_mo(vlan)
16 handle.commit()
17 handle.logout()
```

Transactions

ex-06.py

DEVWKS-2060 ▸ ex-06.py ▸ ...

```
1  """
2      ex-06.py
3  """
4
5  from ucsmsdk.ucshandle import UcsHandle
6  from ucsmsdk.mometa.fabric.FabricVlan import FabricVlan
7  handle = UcsHandle("198.18.133.91", "admin", "password")
8  handle.login()
9
10 fabric_lan_cloud = handle.query_classid("FabricLanCloud")
11 vlans = ['200','300', '400', '500']
12
13 for vlan in vlans:
14     vlan = FabricVlan(parent_mo_or_dn=fabric_lan_cloud[0],
15                       name="vlan" + vlan,
16                       id=vlan)
17
18     handle.add_mo(vlan)
19
20 handle.commit()
21 handle.logout()
```


Compare / Sync

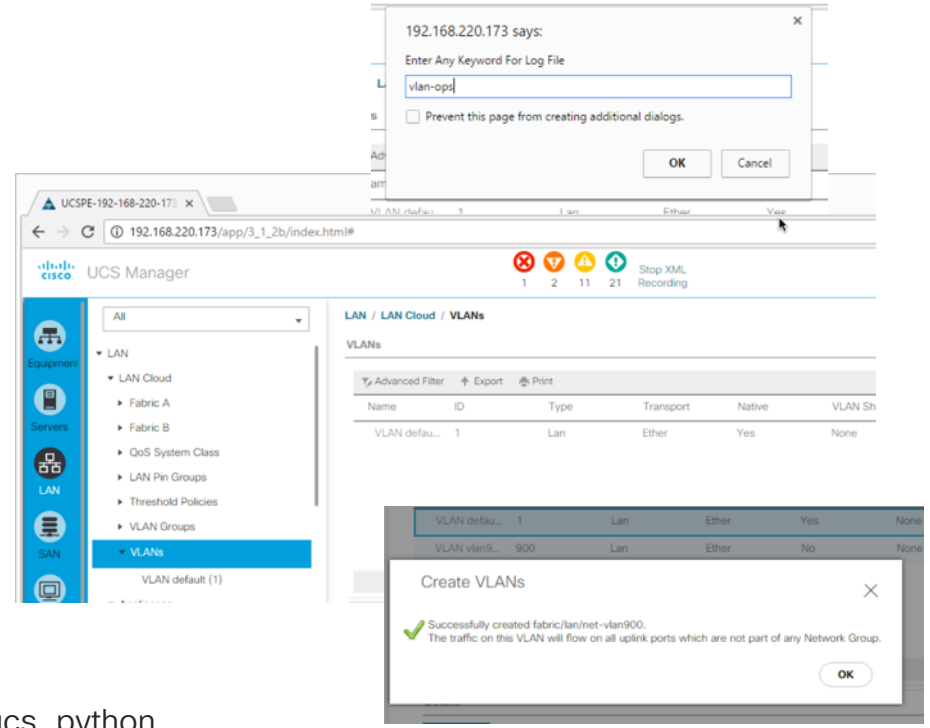
ex-07.py

DEVWKS-2060 ▸ ex-07.py ▸ ...

```
1  """
2  ex-07.py
3  """
4
5  from ucsmsdk.ucshandle import UcsHandle
6  from ucsmsdk.utils import comparesyncmo
7  source_ucs=UcsHandle("198.18.133.91", "admin", "password")
8  target_ucs=UcsHandle("198.18.134.249", "admin", "password")
9  source_ucs.login()
10 target_ucs.login()
11
12 source_ucs_vlans=source_ucs.query_classid("fabricVlan")
13 target_ucs_vlans=target_ucs.query_classid("fabricVlan")
14
15 difference_vlans=comparesyncmo.compare_ucs_mo(target_ucs_vlans, source_ucs_vlans)
16
17 # print the difference to the console
18 comparesyncmo.write_mo_diff(difference_vlans)
19
20 comparesyncmo.sync_ucs_mo(target_ucs, difference_vlans, delete_not_present=True)
21
22 source_ucs.logout()
23 target_ucs.logout()
24
```

Code Generation

1. control-option/alt-q (MAC)
   Ctrl-Alt-q (Windows)
2. Click Record XML
3. Do Configuration
4. Click Stop XML Recording
5. Download XML file
6. Feed XML file to UCS Python SDK conversion method




```
from ucsmsdk.utils.converttopython import convert_to_ucs_python
# Windows
convert_to_ucs_python(xml=True, literal_path="Z:\\Downloads\\vlan-ops_xmlReq.log")
# Linux
convert_to_ucs_python(xml=True, literal_path="/Users/demouser/Downloads/vlan-ops_xmlReq.log")
```

Complete your online session survey

- Please complete your session survey after each session. Your feedback is very important.
- Complete a minimum of 4 session surveys and the Overall Conference survey (starting on Thursday) to receive your Cisco Live t-shirt.
- All surveys can be taken in the Cisco Events Mobile App or by logging in to the Content Catalog on ciscolive.com/emea.

Cisco Live sessions will be available for viewing on demand after the event at ciscolive.com.

cisco Live!

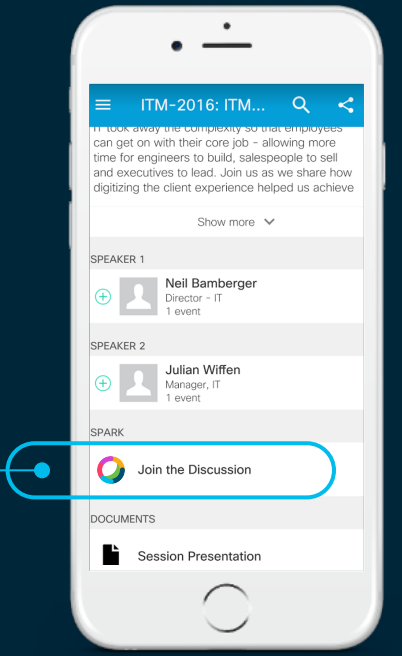
Cisco Webex Teams

Questions?

Use Cisco Webex Teams to chat with the speaker after the session

How

1. Find this session in the Cisco Events Mobile App
2. Click “Join the Discussion”
3. Install Webex Teams or go directly to the team space
4. Enter messages/questions in the team space



cs.co/ciscolivebot#

Continue your education

Demos in the Cisco Showcase

Walk-In Labs

Meet the Engineer 1:1 meetings

Related sessions

Learn more about the new DevNet Certifications and how you can prepare now!

	Associate Level	Specialist Level	Professional Level	Expert Level
Engineering				
Software				

Future Offering

Start Here | Upcoming Cisco DevNet Certifications

- Start at **Meet DevNet**

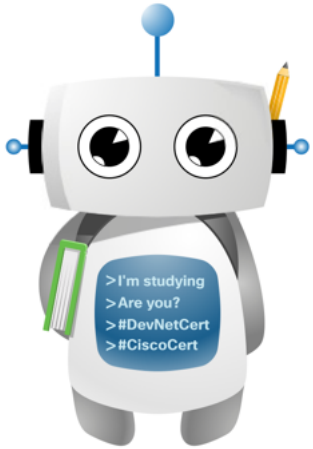
  DEVNET-2864: Getting ready for Cisco DevNet Certifications
  Offered daily at 9am, 1pm & 4pm at Meet DevNet

- Attend a **brownbag session**

  DEVNET-4099: DevNet Certifications: Bringing software practices & software skills to networking
  Offered daily 12:15-12:45 in the DevNet Zone Theater

- Visit the **Learning@Cisco** booth

- Scan this code to **sign up** for the latest updates or go to http://cs.co/20eur02



cisco Live!

Thank you

cisco Live!