Lab 14 Use the Cisco IOS XE NETCONF API

Do Lab 14 Tasks 1 and 2 in full before starting the following task 3.

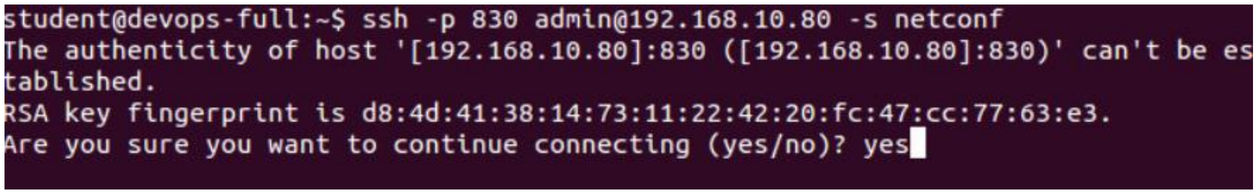
Task 3: NETCONF on IOS XE initial configuration with complete XML configuration data.

Ensure that Labs **9, 10, 13, 14** and Lab **14** tasks 1 and 2 are completed in full before starting this task.

In this task you will use **NETCONF** on your **CSR1000v** to extract the entire XML configuration and use that to perform configuration changes with Python.

How-to Steps

1. Connect to your **Win7** VM.
2. Open Putty as needed and connect to your IOS XE **CSR1000v**.
3. To demonstrate extracting XML in later steps, configure basic OSPF into the IOS XE router.
```
config t
router ospf 1
      network 10.0.0.0 0.255.255.255 area 0
      end
wr
```
4. Connect to your **Ubuntu Configured** VM.
5. Open a Terminal widow.
6. Type in the command to connect to your **CSR1000v** router with NETCONF.
```
ssh -p 830 admin@192.168.10.80 -s netconf
```
7. Respond with a **yes** if prompted about an RSA key. (you will likely not be prompted for later ssh sessions)



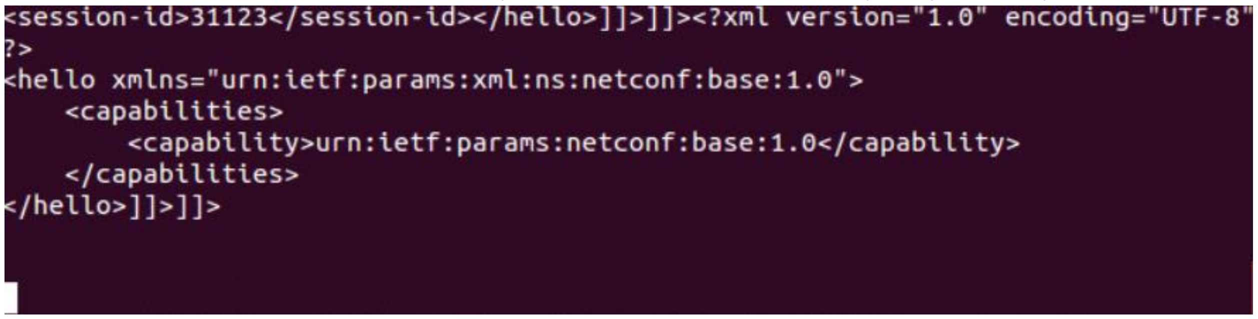
```
student@devops-full:~$ ssh -p 830 admin@192.168.10.80 -s netconf
The authenticity of host '[192.168.10.80]:830 ([192.168.10.80]:830)' can't be es
tablished.
RSA key fingerprint is d8:4d:41:38:14:73:11:22:42:20:fc:47:cc:77:63:e3.
Are you sure you want to continue connecting (yes/no)? yes
```

8. Enter the password **cisco** when prompted. Simply note all the XML output of this device capabilities.
9. Enter in this exact XML code directly after the NETCONF prompt. This is stating your NETCONF client capabilities. You can type this in exactly or copy the identical content from the file in your **Class_Share** link on your Ubuntu desktop: **Z:\XML\NetconfClient.xml**.

```
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <capabilities>
        <capability>urn:ietf:params:netconf:base:1.0</capability>
    </capabilities>
</hello>]]>]]>
```

Note: Additional capabilities can be entered above obtained from the NETCONF device during the initial capabilities exchange.

10. You can hit enter a few times. Note how you don't have a left indented prompt at this point.



```
<session-id>31123</session-id></hello>]]>]]><?xml version="1.0" encoding="UTF-8"
?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <capabilities>
        <capability>urn:ietf:params:netconf:base:1.0</capability>
    </capabilities>
</hello>]]>]]>
```

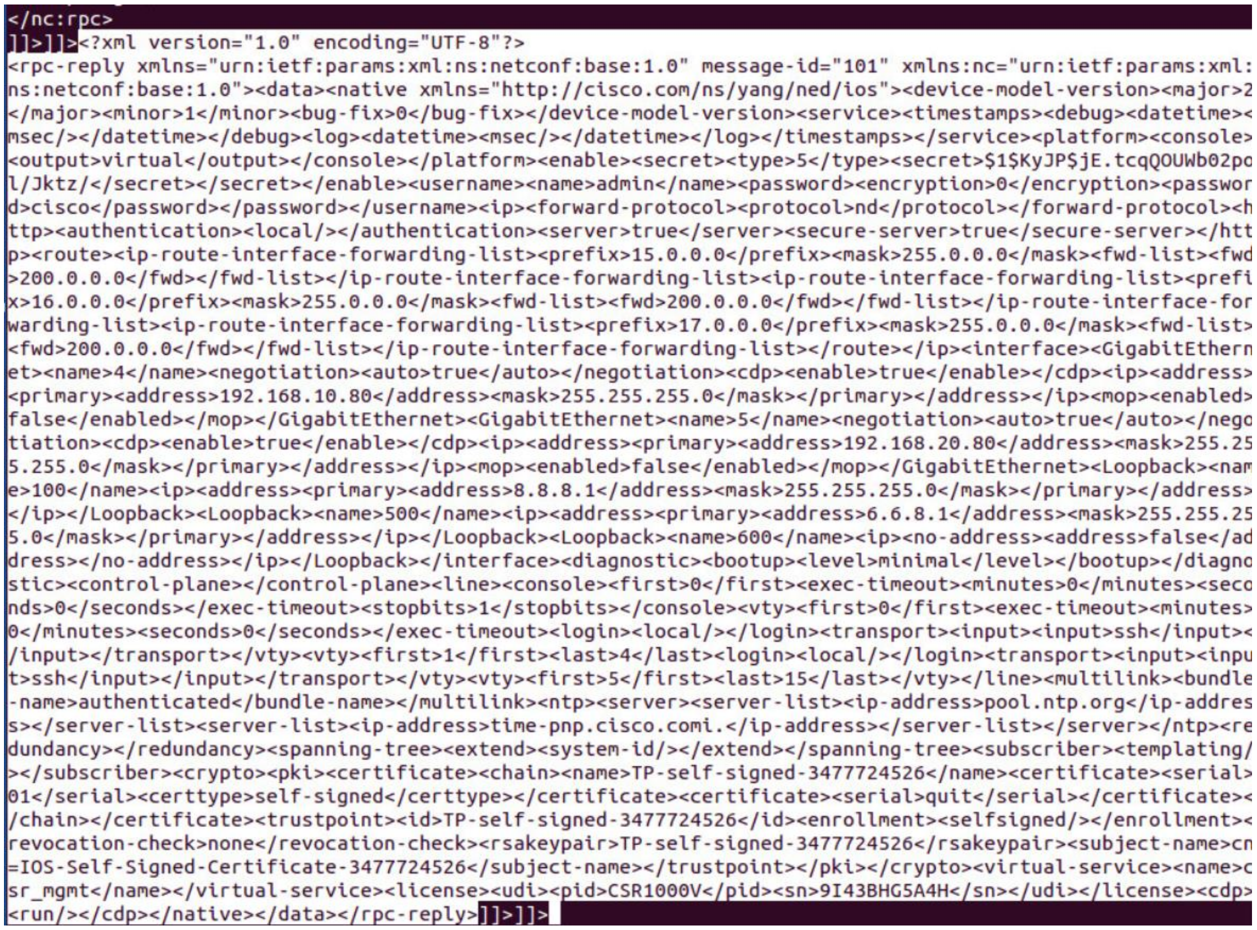
11. Maximize your terminal window to minimize the character returns within longer XML lines.

Note: NETCONF messages always end with the unique prompt of **]]>]]>**. This prompt occurs after each communication, and is not itself valid XML.

12. To get the entire device config in XML, enter in this entire NETCONF **get** command after the prompt. You can type this in exactly, or copy the identical content from the file: **Z:\XML\NetconfConfig.xml**

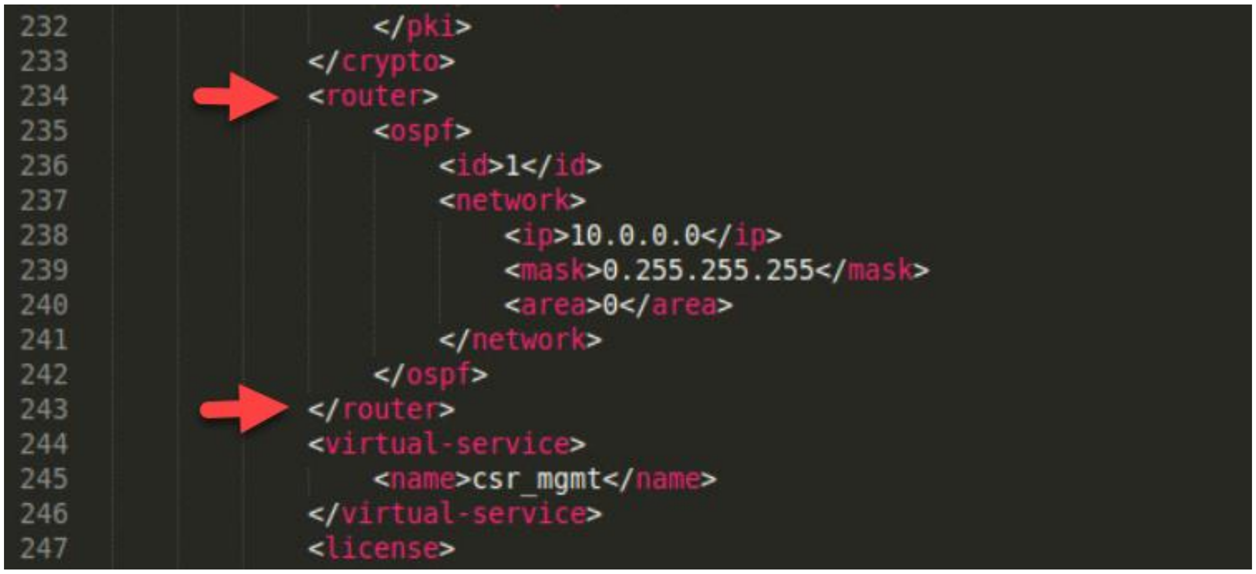
```
<?xml version="1.0"?>
<nc:rpc message-id="101" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
    <nc:get>
        <nc:filter type="subtree">
            <native xmlns="http://cisco.com/ns/yang/ned/ios">
            </native>
        </nc:filter>
    </nc:get>
</nc:rpc>
]]>]]>
```

13. Note all the output which is your IOS XE router running configuration in XML. Copy the entire XML output code starting right after the NETCONF prompt to the very end excluding the trailing NETCONF prompt of **]]>]]>**:

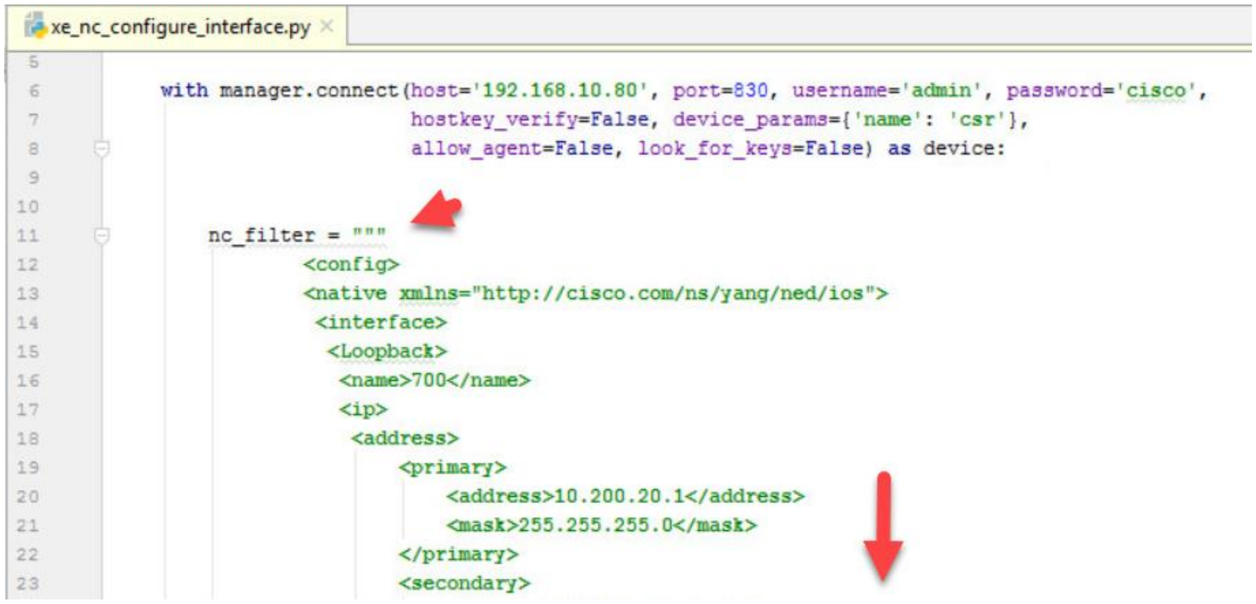


```
</nc:rpc>
]]>]]><?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101" xmlns:nc="urn:ietf:params:xml:
ns:netconf:base:1.0"><data><native xmlns="http://cisco.com/ns/yang/ned/ios"><device-model-version><major>2
</major><minor>1</minor><bug-fix>0</bug-fix></device-model-version><service><timestamps><debug><datetime><
msec/></datetime></debug><log><datetime><msec/></datetime></log></timestamps></service><platform><console>
<output>virtual</output></console></platform><enable><secret><type>5</type><secret>$1$KyJP$jE.tcqQOUWb02po
l/Jktz/</secret></secret></enable><username><name>admin</name><password><encryption>0</encryption><passwor
d>cisco</password></password></username><ip><forward-protocol><protocol>nd</protocol></forward-protocol><h
ttp><authentication><local/></authentication><server>true</server><secure-server>true</secure-server></htt
p><route><ip-route-interface-forwarding-list><prefix>15.0.0.0</prefix><mask>255.0.0.0</mask><fwd-list><fwd
>200.0.0.0</fwd></fwd-list></ip-route-interface-forwarding-list><ip-route-interface-forwarding-list><prefi
x>16.0.0.0</prefix><mask>255.0.0.0</mask><fwd-list><fwd>200.0.0.0</fwd></fwd-list></ip-route-interface-for
warding-list><ip-route-interface-forwarding-list><prefix>17.0.0.0</prefix><mask>255.0.0.0</mask><fwd-list>
<fwd>200.0.0.0</fwd></fwd-list></ip-route-interface-forwarding-list></route></ip><interface><GigabitEthern
et><name>4</name><negotiation><auto>true</auto></negotiation><cdp><enable>true</enable></cdp><ip><address>
<primary><address>192.168.10.80</address><mask>255.255.255.0</mask></primary></address></ip><mop><enabled>
false</enabled></mop></GigabitEthernet><GigabitEthernet><name>5</name><negotiation><auto>true</auto></nego
tiation><cdp><enable>true</enable></cdp><ip><address><primary><address>192.168.20.80</address><mask>255.25
5.255.0</mask></primary></address></ip><mop><enabled>false</enabled></mop></GigabitEthernet><Loopback><nam
e>100</name><ip><address><primary><address>8.8.8.1</address><mask>255.255.255.0</mask></primary></address>
</ip></Loopback><Loopback><name>500</name><ip><address><primary><address>6.6.8.1</address><mask>255.255.25
5.0</mask></primary></address></ip></Loopback><Loopback><name>600</name><ip><no-address><address>false</ad
dress></no-address></ip></Loopback></interface><diagnostic><bootup><level>minimal</level></bootup></diagno
stic><control-plane></control-plane><line><console><first>0</first><exec-timeout><minutes>0</minutes><seco
nds>0</seconds></exec-timeout><stopbits>1</stopbits></console><vty><first>0</first><exec-timeout><minutes>
0</minutes><seconds>0</seconds></exec-timeout><login><local/></login><transport><input><input>ssh</input><
/input></transport></vty><vty><first>1</first><last>4</last><login><local/></login><transport><input><inpu
t>ssh</input></input></transport></vty><vty><first>5</first><last>15</last></vty></line><multilink><bundle
-name>authenticated</bundle-name></multilink><ntp><server><server-list><ip-address>pool.ntp.org</ip-addres
s></server-list><server-list><ip-address>time-pnp.cisco.comi.</ip-address></server-list></server></ntp><re
dundancy></redundancy><spanning-tree><extend><system-id/></extend></spanning-tree><subscriber><templating/
></subscriber><crypto><pki><certificate><chain><name>TP-self-signed-3477724526</name><certificate><serial>
01</serial><certtype>self-signed</certtype></certificate><certificate><serial>quit</serial></certificate><
/chain></certificate><trustpoint><id>TP-self-signed-3477724526</id><enrollment><selfsigned/></enrollment><
revocation-check>none</revocation-check><rsakeypair>TP-self-signed-3477724526</rsakeypair><subject-name>cn
=IOS-Self-Signed-Certificate-3477724526</subject-name></trustpoint></pki></crypto><virtual-service><name>c
sr_mgmt</name></virtual-service><license><udi><pid>CSR1000V</pid><sn>9I43BHG5A4H</sn></udi></license><cdp>
<run/></cdp></native></data></rpc-reply>]]>]]>
```

14. Within your lab environment, go to any online XML formatter such as:
    https://xmlvalidator.com
    https://www.webtoolkitonline.com
15. Paste in the XML code to any validator tool and ensure that the XML is validated.
16. Copy and paste the formatted XML back into **Sublime**. You now have the entire configuration of your IOS XE device in properly formatted XML. Note how your OSPF configuration is present in XML.

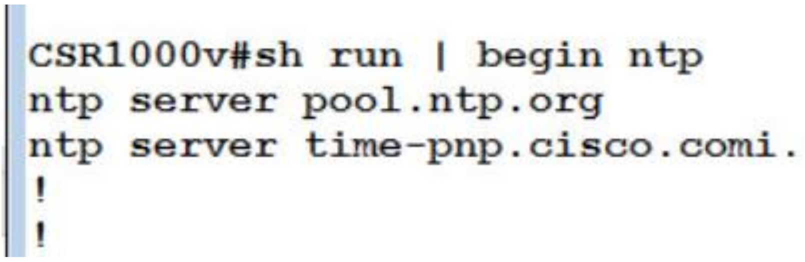

```
            </pki>
        </crypto>
        <router>
            <ospf>
                <id>1</id>
                <network>
                    <ip>10.0.0.0</ip>
                    <mask>0.255.255.255</mask>
                    <area>0</area>
                </network>
            </ospf>
        </router>
        <virtual-service>
            <name>csr_mgmt</name>
        </virtual-service>
        <license>
```

17. Save the file as **iosxe.xml** to your **Class_Share** directory.
18. Open **Win7** VM and open the saved **iosxe.xml** file in **Notepad++.** Read and study your entire router configuration in XML. Note carefully the hierarchy of XML entries.
19. In **PyCharm**, open the Python file as needed from **Z:\Python**.
    `xe_nc_configure_interface.py`
20. Ensure you have the proper router credentials in your Python code from prior lab steps.
21. Take careful note of the **XML** that exists between the triple quotes **"""**.


xe_nc_configure_interface.py

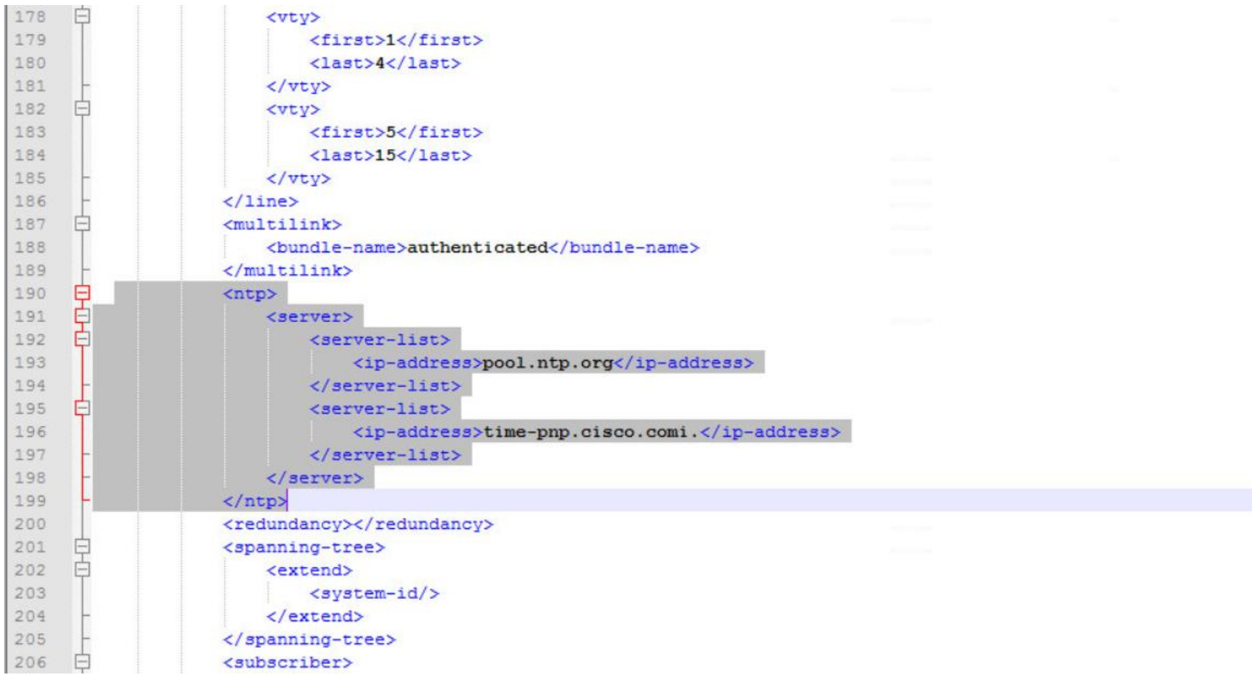
```
    with manager.connect(host='192.168.10.80', port=830, username='admin', password='cisco',
                         hostkey_verify=False, device_params={'name': 'csr'},
                         allow_agent=False, look_for_keys=False) as device:


        nc_filter = """
                <config>
                <native xmlns="http://cisco.com/ns/yang/ned/ios">
                 <interface>
                  <Loopback>
                   <name>700</name>
                   <ip>
                    <address>
                        <primary>
                            <address>10.200.20.1</address>
                            <mask>255.255.255.0</mask>
                        </primary>
                        <secondary>
```

22. At this point, you have all you need to make any router configuration change, based on your XML content and your Python script. As an example, review your existing NTP configurations on the CSR1000V device.



```
CSR1000v#sh run | begin ntp
ntp server pool.ntp.org
ntp server time-pnp.cisco.comi.
!
!
```

23. In **Notepad++**, highlight and copy exactly the NTP configuration in **XML**. Pay very close attention to every character that relates to NTP.

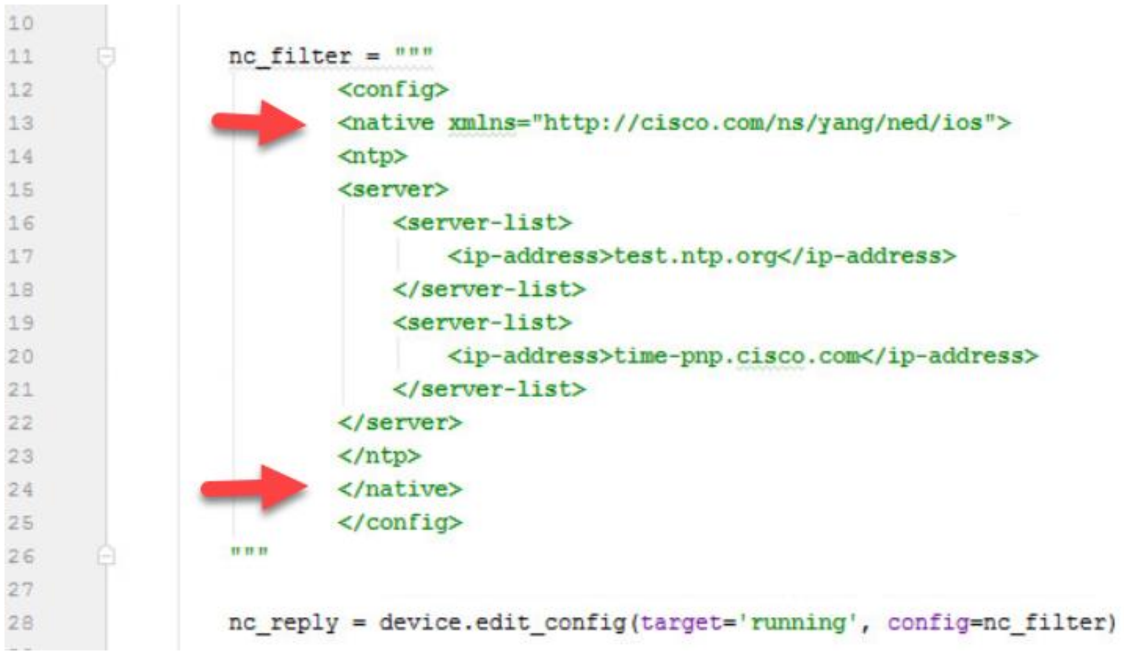


```
            <vty>
                <first>1</first>
                <last>4</last>
            </vty>
            <vty>
                <first>5</first>
                <last>15</last>
            </vty>
        </line>
        <multilink>
            <bundle-name>authenticated</bundle-name>
        </multilink>
        <ntp>
            <server>
                <server-list>
                    <ip-address>pool.ntp.org</ip-address>
                </server-list>
                <server-list>
                    <ip-address>time-pnp.cisco.comi.</ip-address>
                </server-list>
            </server>
        </ntp>
        <redundancy></redundancy>
        <spanning-tree>
            <extend>
                <system-id/>
            </extend>
        </spanning-tree>
        <subscriber>
```

24. Paste the XML into your Python script after the first 2 lines and before the last two XML lines. You may have to clean up your indenting after pasting. As an example, change the ntp settings to:

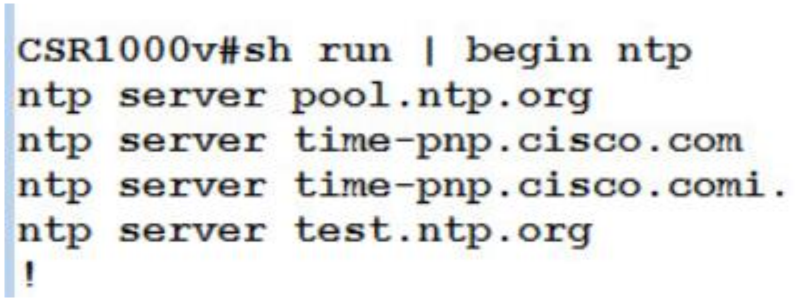
```
test.ntp.org
```

```
time-pnp.cisco.com
```



```
nc_filter = """
        <config>
        <native xmlns="http://cisco.com/ns/yang/ned/ios">
        <ntp>
        <server>
            <server-list>
                <ip-address>test.ntp.org</ip-address>
            </server-list>
            <server-list>
                <ip-address>time-pnp.cisco.com</ip-address>
            </server-list>
        </server>
        </ntp>
        </native>
        </config>
"""

nc_reply = device.edit_config(target='running', config=nc_filter)
```

25. From **PyCharm**, run or step through your code in full and ensure no errors.
26. Return to the **CSR1000v** device and verify the two new added NTP entries.



```
CSR1000v#sh run | begin ntp
ntp server pool.ntp.org
ntp server time-pnp.cisco.com
ntp server time-pnp.cisco.comi.
ntp server test.ntp.org
!
```

27. Optionally, take any other XML code from Notepad++ and repeat these steps to perform any other configuration change to your CSR1000v. Take very careful note to get the exact XML code for each configuration change.

Challenge results

This lab demonstrated extracting the source XML code of your CSR1000v router and editing the configuration of an IOS XE device with Python using NETCONF.