

Lab – NETCONF w/Python: Device Configuration

Objectives

Part 1: Retrieve the IOS XE VMs' existing running configuration

Part 2: Update the device's configuration

Background / Scenario

In this lab, you will learn how to use the NETCONF ncclient to retrieve the device's configuration, update and create new interface configuration. You will also learn why the transactional support of NETCONF is important for getting consistent network changes.

Required Resources

- Access to a router with the IOS XE operating system version 16.6 or higher
- Python 3.x environment

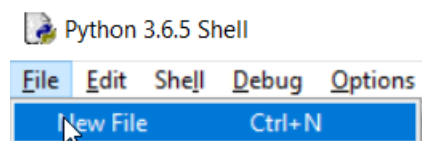
Part 1: Retrieve the IOS XE VMs' existing running configuration

In this part, you will use the ncclient module to retrieve the device's running configuration. The data are returned back in XML form that in the following steps is being transformed into more human readable format.

Step 1: Use ncclient to retrieve the device's running configuration.

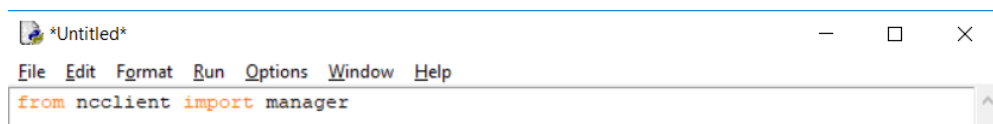
The ncclient module provides a “manager” class with “connect ()” function to setup the remote NETCONF connection. After a successful connection, the returned object represents the NETCONF connection to the remote device.

- a. In Python IDLE, create a new Python script file:



- b. In the new Python script file editor, import the “manager” class from the ncclient module:

```
from ncclient import manager
```



- c. Setup an nc connection object using the manager.connect () function to the IOS XE device.

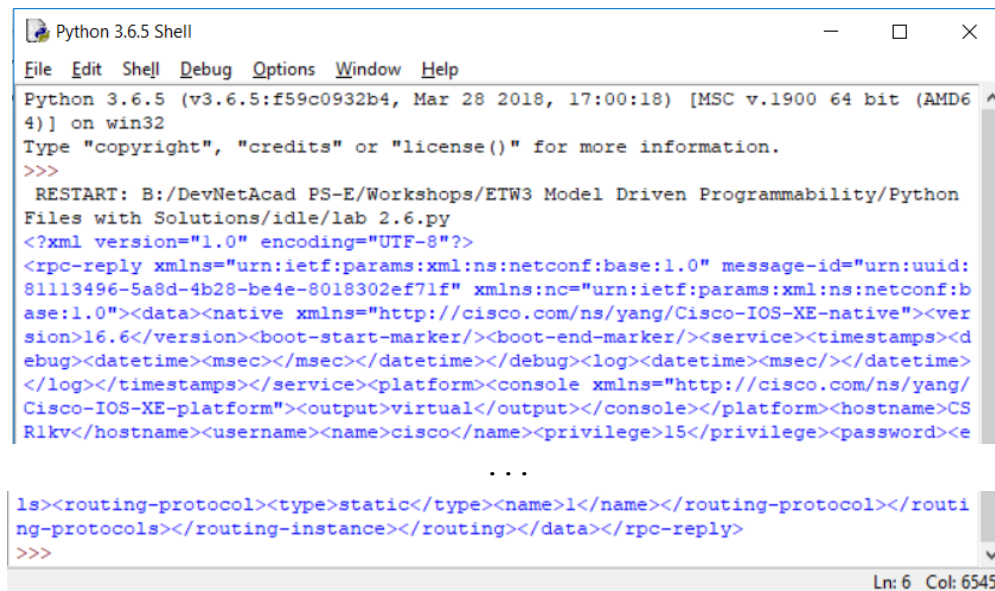
```
m = manager.connect(  
    host="192.168.56.101",  
    port=830,  
    username="cisco",  
    password="cisco123!",  
    hostkey_verify=False  
)
```

The parameters of the `manager.connect()` function are:

- `host` – the address (host or IP) of the remote device (adjust the IP address to match the router's current address)
 - `port` – the remote port of the ssh service
 - `username` – remote ssh username (in this lab "cisco" for that was setup in the IOS XE VM)
 - `password` – remote ssh password (in this lab "cisco123!" for that was setup in the IOS XE VM)
 - `hostkey_verify` – whether to verify the ssh fingerprint (in lab it is safe to set to False, in production environments you should always verify the ssh fingerprints)
- d. After a successful NETCONF connection, using the "`get_config()`" function of the "m" NETCONF session object retrieve and print the device's running configuration. The `get_config()` function expects a "source" string parameter that defines the source NETCONF data-store.

```
netconf_reply = m.get_config(source="running")  
print(netconf_reply)
```

- e. Execute the Python script and explore the output:



```
Python 3.6.5 Shell  
File Edit Shell Debug Options Window Help  
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 17:00:18) [MSC v.1900 64 bit (AMD64)] on win32  
Type "copyright", "credits" or "license()" for more information.  
>>>  
RESTART: B:/DevNetAcad PS-E/Workshops/ETW3 Model Driven Programmability/Python  
Files with Solutions/idle/lab 2.6.py  
<?xml version="1.0" encoding="UTF-8"?>  
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="urn:uuid:  
81113496-5a8d-4b28-be4e-8018302ef71f" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"><data><native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native"><version>16.6</version><boot-start-marker/><boot-end-marker/><service><timestamps><debug><datetime><msec></msec></datetime></debug><log><datetime><msec></msec></datetime></log></timestamps></service><platform><console xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-platform"><output>virtual</output></console></platform><hostname>CSR1kv</hostname><username><name>cisco</name><privilege>15</privilege><password><e  
...  
ls><routing-protocol><type>static</type><name>l</name></routing-protocol></routing-protocols></routing-instance></routing></data></rpc-reply>  
>>>
```

Step 2: Use CodeBeautify.com to evaluate the response

Code Beautify maintains a website for viewing code in a more human readable format. The XML viewer URL is <https://codebeautify.org/xmlviewer>

- f. Copy the XML from IDLE to XML Viewer.
- g. Click **Tree View** or **Beautify / Format** to render the raw XML output into a more human readable format.

Lab – NETCONF w/Python: Device Configuration

Code Beautify

XML Viewer

XML Input

sample

Load Url

Browse

Tree View

Beautify / Format

Minify

Free Online Converter

My Converter Hub

Free Online Converter

OPEN

XML to JSON

Export to CSV

Download

Result : Beautify XML

```
<?xml version="1.0" encoding="UTF-8"?>
1
2 <rpc-reply xmlns:urnietf:params:xml:ns:netconf:base:1.0 message-id="urn:uuid:81113496-548d-4b28-b4de-8018302e7f1f" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"><data><native xmlns="http://cisco.com/ns/yang/Cisco-10S-XE-native" version=
3
4 "10.6.6" version=
5
6 "10.6.6" start-marker="
7
8 <boot-start-marker"
9
10 <boot-end-marker"
11
12 <service"
13
14 <timestamps"
15
16 <debug"
17
18 <date-time"
19
20 <exec"
21
22 <date-time"
23
24 </log"
25
26 </timestamps"
27
28 </platform"
29
30 <platform"
31
32 <hostname"
33
34 <username"
35
36 <cryptokey"
37
38 <privilege"
39
40 <password"
41
42 <encryption"
43
44 <password"
45
46 <password"
47
48 <password"
49
50 <password"
51
52 <password"
53
54 <password"
55
56 <password"
57
58 <password"
59
60 <password"
61
62 <password"
63
64 <password"
65
66 <password"
67
68 <password"
69
70 <password"
71
72 <password"
73
74 <password"
75
76 <password"
77
78 <password"
79
80 <password"
81
82 <password"
83
84 <password"
85
86 <password"
87
88 <password"
89
90 <password"
91
92 <password"
93
94 <password"
95
96 <password"
97
98 <password"
99
100 <password"
101
102 <password"
103
104 <password"
105
106 <password"
107
108 <password"
109
110 <password"
111
112 <password"
113
114 <password"
115
116 <password"
117
118 <password"
119
120 <password"
121
122 <password"
123
124 <password"
125
126 <password"
127
128 <password"
129
130 <password"
131
132 <password"
133
134 <password"
135
136 <password"
137
138 <password"
139
140 <password"
141
142 <password"
143
144 <password"
145
146 <password"
147
148 <password"
149
150 <password"
151
152 <password"
153
154 <password"
155
156 <password"
157
158 <password"
159
160 <password"
161
162 <password"
163
164 <password"
165
166 <password"
167
168 <password"
169
170 <password"
171
172 <password"
173
174 <password"
175
176 <password"
177
178 <password"
179
180 <password"
181
182 <password"
183
184 <password"
185
186 <password"
187
188 <password"
189
190 <password"
191
192 <password"
193
194 <password"
195
196 <password"
197
198 <password"
199
200 <password"
201
202 <password"
203
204 <password"
205
206 <password"
207
208 <password"
209
210 <password"
211
212 <password"
213
214 <password"
215
216 <password"
217
218 <password"
219
220 <password"
221
222 <password"
223
224 <password"
225
226 <password"
227
228 <password"
229
230 <password"
231
232 <password"
233
234 <password"
235
236 <password"
237
238 <password"
239
240 <password"
241
242 <password"
243
244 <password"
245
246 <password"
247
248 <password"
249
250 <password"
251
252 <password"
253
254 <password"
255
256 <password"
257
258 <password"
259
260 <password"
261
262 <password"
263
264 <password"
265
266 <password"
267
268 <password"
269
270 <password"
271
272 <password"
273
274 <password"
275
276 <password"
277
278 <password"
279
280 <password"
281
282 <password"
283
284 <password"
285
286 <password"
287
288 <password"
289
290 <password"
291
292 <password"
293
294 <password"
295
296 <password"
297
298 <password"
299
300 <password"
301
302 <password"
303
304 <password"
305
306 <password"
307
308 <password"
309
310 <password"
311
312 <password"
313
314 <password"
315
316 <password"
317
318 <password"
319
320 <password"
321
322 <password"
323
324 <password"
325
326 <password"
327
328 <password"
329
330 <password"
331
332 <password"
333
334 <password"
335
336 <password"
337
338 <password"
339
340 <password"
341
342 <password"
343
344 <password"
345
346 <password"
347
348 <password"
349
350 <password"
351
352 <password"
353
354 <password"
355
356 <password"
357
358 <password"
359
360 <password"
361
362 <password"
363
364 <password"
365
366 <password"
367
368 <password"
369
370 <password"
371
372 <password"
373
374 <password"
375
376 <password"
377
378 <password"
379
380 <password"
381
382 <password"
383
384 <password"
385
386 <password"
387
388 <password"
389
390 <password"
391
392 <password"
393
394 <password"
395
396 <password"
397
398 <password"
399
400 <password"
401
402 <password"
403
404 <password"
405
406 <password"
407
408 <password"
409
410 <password"
411
412 <password"
413
414 <password"
415
416 <password"
417
418 <password"
419
420 <password"
421
422 <password"
423
424 <password"
425
426 <password"
427
428 <password"
429
430 <password"
431
432 <password"
433
434 <password"
435
436 <password"
437
438 <password"
439
440 <password"
441
442 <password"
443
444 <password"
445
446 <password"
447
448 <password"
449
450 <password"
451
452 <password"
453
454 <password"
455
456 <password"
457
458 <password"
459
460 <password"
461
462 <password"
463
464 <password"
465
466 <password"
467
468 <password"
469
470 <password"
471
472 <password"
473
474 <password"
475
476 <password"
477
478 <password"
479
480 <password"
481
482 <password"
483
484 <password"
485
486 <password"
487
488 <password"
489
490 <password"
491
492 <password"
493
494 <password"
495
496 <password"
497
498 <password"
499
500 <password"
501
502 <password"
503
504 <password"
505
506 <password"
507
508 <password"
509
510 <password"
511
512 <password"
513
514 <password"
515
516 <password"
517
518 <password"
519
520 <password"
521
522 <password"
523
524 <password"
525
526 <password"
527
528 <password"
529
530 <password"
531
532 <password"
533
534 <password"
535
536 <password"
537
538 <password"
539
540 <password"
541
542 <password"
543
544 <password"
545
546 <password"
547
548 <password"
549
550 <password"
551
552 <password"
553
554 <password"
555
556 <password"
557
558 <password"
559
560 <password"
561
562 <password"
563
564 <password"
565
566 <password"
567
568 <password"
569
570 <password"
571
572 <password"
573
574 <password"
575
576 <password"
577
578 <password"
579
580 <password"
581
582 <password"
583
584 <password"
585
586 <password"
587
588 <password"
589
590 <password"
591
592 <password"
593
594 <password"
```

- h. To simplify the view, close the XML elements that are under the `rpc-reply/data` structure:

Result : Beautify XML

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <rpc-reply
3      xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="urn:uuid
        :81113496-5a8d-4b28-be4e-8018302ef71f"
4      xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
5      <data>
6          <native></native>
186         <interfaces></interfaces>
289         <network-instances></network-instances>
350         <interfaces></interfaces>
398         <routing></routing>
412     </data>
413 </rpc-reply>
```

- i. Note that the `rpc-reply/data/native` element when opened, it contains an attribute `xmlns` that points to “Cisco-IOX-XE-native” YANG model. That means this part of the configuration is Cisco Native for IOS XE.

```

6 <native
7   xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
8     <version>16.6</version>
9     <boot-start-marker/>
10    <boot-end-marker/>
11    <service>
12      <timestamps>

```

- j. Also note that there are two “interfaces” elements – one with xmlns pointing to “http://openconfig.net/yang/interfaces” YANG model, while the other pointing to “ietf-interfaces” YANG model.

```
5 <data>
6 <native></native>
186 <interfaces
187     xmlns="http://openconfig.net/yang/interfaces">
188     <interface>
189         <name>GigabitEthernet1</name>
190         <config>
```

```

350 <interfaces
351     xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
352     <interface>
353         <name>GigabitEthernet1</name>
354         <type
355             xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana
                 -if-type">ianaift:ethernetCsmacd
356         </type>
357         <enabled>true</enabled>

```

Both are used to describe the configuration of the interfaces, with a difference that the openconfig.net YANG model does support sub-interfaces, while the ietf-interfaces YANG model does not.

Step 3: Use `toprettyxml()` function to prettify the output.

- Python has built in support to work with XML files. The “`xml.dom.minidom`” module can be used to prettify the output with the `toprettyxml()` function.
- Import the “`xml.dom.minidom`” module:

```
import xml.dom.minidom
```
- Replace the simple print function “`print(netconf_reply)`” with a version that prints prettified XML output:

```
print( xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml() )
```
- Execute the updated Python script and explore the output.

Step 4: Use filters to retrieve a configuration defined by a specific YANG model

- NETCONF has support to return only data that are defined in a filter element.
- Create the following `netconf_filter` variable that contains an XML NETCONF filter element to only retrieve data defined by the Cisco IOS XE Native YANG model:

```

netconf_filter = """
<filter>
  <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native" />
</filter>
"""

```

- Include the `netconf_filter` variable in the `get_config()` call using the “filter” parameter:

```
netconf_reply = m.get_config(source="running", filter=netconf_filter)
print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())
```
- Execute the updated Python script and explore the output:

```
RESTART: B:/DevNetAcad PS-E/Workshops/ETW3 Model Driven Programmability/Python
Files with Solutions/idle/lab 2.6.py
<?xml version="1.0" ?>
<rpc-reply message-id="urn:uuid:7a860e08-0447-4482-9ce6-7ed0efe2f24a" xmlns="urn
:ietf:params:xml:ns:netconf:base:1.0" xmlns:nc="urn:ietf:params:xml:ns:netconf:b
ase:1.0">
  <data>
    <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
      <version>16.6</version>
      <boot-start-marker/>
      <boot-end-marker/>
      <service>
        <timestamps>
          <debug>
            <datetime>
              <msec/>
            </datetime>
          </debug>
          <log>
            <datetime>
              <msec/>
            </datetime>
          </log>
        </timestamps>
      </service>
    </native>
  </data>
</rpc-reply>
>>>
```

...

```
    <vty>
      <first>5</first>
      <last>15</last>
      <login>
        <local/>
      </login>
    </vty>
  </line>
  <diagnostic xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE
-diagnostics">
    <bootup>
      <level>minimal</level>
    </bootup>
  </diagnostic>
</native>
</data>
</rpc-reply>
>>>
```

Part 2: Update the device's configuration

Step 1: Create a new Python script file

- a. In IDLE create a new Python script file
- b. Import the required modules and setup the NETCONF session:

```
from ncclient import manager
import xml.dom.minidom

m = manager.connect(
    host="192.168.56.101",
    port=830,
    username="cisco",
    password="cisco123!",
    hostkey_verify=False
)
```

Step 2: Change the hostname

- f. In order to update an existing setting in the configuration, you can extract the setting location from the configuration retrieved in Step 1:



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <rpc-reply
3   xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="urn
   :uuid:ffad8001-6b82-4094-acc4-6f456ba9e088"
4   xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
5   <data>
6     <native
7       xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
8       <version>16.6</version>
9       <boot-start-marker/>
10      <boot-end-marker/>
11      <service>
12        <timestamps>
13          <debug>
14            <datetime>
15              <msec></msec>
16            </datetime>
17          </debug>
18          <log>
19            <datetime>
20              <msec></msec>
21            </datetime>
22          </log>
23        </timestamps>
24      </service>
25      <platform>
26        <console
27          xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-
28            -platform">
29          <output>virtual</output>
30        </console>
31      </platform>
32      <hostname>CSR1kv</hostname>
```

- g. The configuration update is always enclosed in a “config” XML element that includes a tree of XML elements that require update.
- h. Create a `netconf_data` variable that holds a configuration update for the hostname element as defined in the Cisco IOS XE Native YANG Model:

```
netconf_data = ""
<config>
  <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
    <hostname>NEWHOSTNAME</hostname>
  </native>
</config>
```

```
"""
```

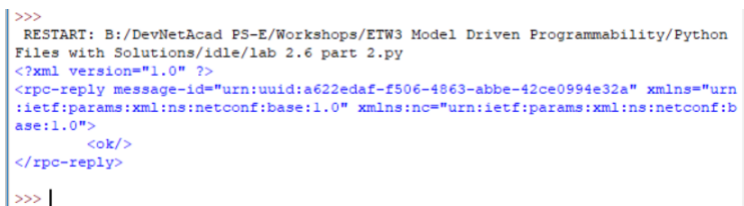
- i. Edit the existing device configuration with the “`edit_config()`” function of the “`m`” NETCONF session object. The `edit_config()` function expects two parameters:

- `target` – the target netconf data-store to be updated
- `config` – the configuration update

The `edit_config()` function returns an XML object containing information about the change success. After editing the configuration, print the returned value:

```
netconf_reply = m.edit_config(target="running", config=netconf_data)
print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())
```

- j. Before executing the new Python script, check the current hostname by connecting to the console of the IOS XE VM.
- k. Execute the Python script and explore the output:



```
>>>
RESTART: B:/DevNetAcad PS-E/Workshops/ETW3 Model Driven Programmability/Python
Files with Solutions/ide/lab 2.6 part 2.py
<?xml version="1.0" ?>
<rpc-reply message-id="urn:uuid:a622edaf-f506-4863-abbe-42ce0994e32a" xmlns="urn
:ietf:params:xml:ns:netconf:base:1.0" xmlns:nc="urn:ietf:params:xml:ns:netconf:b
ase:1.0">
  <ok/>
</rpc-reply>
>>> |
```

- l. After executing the Python script, if the reply contained the `<ok/>` element, verify whether current hostname has been changed by connecting to the console of the IOS XE VM.

Step 3: Create a loopback interface

- m. Update the `netconf_data` variable to hold a configuration update that creates a new loopback **100** interface with the IP address **100.100.100.100/24**:

```
netconf_data = """
<config>
  <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
    <interface>
      <Loopback>
        <name>100</name>
        <description>TEST1</description>
        <ip>
          <address>
            <primary>
              <address>100.100.100.100</address>
              <mask>255.255.255.0</mask>
            </primary>
          </address>
        </ip>
      </Loopback>
    </interface>
  </native>
</config>
"""
```


- n. Add the new loopback 100 interface by editing the existing device configuration using the “edit_config()” function:


```
netconf_reply = m.edit_config(target="running", config=netconf_data)
print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())
```
- o. Before executing the updated Python script, check using “show ip int brief” and “show int desc” the existing loopback interface by connecting to the console of the IOS XE VM. Take a screenshot of the result.
- p. Execute the Python script and explore the output:

```
>>>
RESTART: B:/DevNetAcad PS-E/Workshops/ETW3 Model Driven Programmability/Python
Files with Solutions/idle/lab 2.6 part 2.py
<?xml version="1.0" ?>
<rpc-reply message-id="urn:uuid:a622edaf-f506-4863-abbe-42ce0994e32a" xmlns="urn
:ietf:params:xml:ns:netconf:base:1.0" xmlns:nc="urn:ietf:params:xml:ns:netconf:b
ase:1.0">
    <ok/>
</rpc-reply>
>>> |
```

- q. After executing the Python script, if the reply contained the <ok/> element, verify whether current loopback interfaces have changed by connecting to the console of the IOS XE VM. Check using “show ip int brief” command. Take a screenshot of the result.

ETW-CSR1000v [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

```
CSR1k>
*May 16 03:58:16.782: %DMI-5-AUTH_PASSED: R0/0: dmiauthd: User 'cisco' authentic
ated successfully from 10.128.207.104:56037 and was authorized for netconf over
ssh. External groups: PRIV15
*May 16 03:58:17.095: %DMI-5-CONFIG_I: R0/0: nesd: Configured from NETCONF/RESTC
ONF by cisco, transaction-id 194
NEWHOSTNAME>
NEWHOSTNAME>
NEWHOSTNAME>
*May 16 04:02:27.468: %DMI-5-AUTH_PASSED: R0/0: dmiauthd: User 'cisco' authentic
ated successfully from 10.128.207.104:56266 and was authorized for netconf over
ssh. External groups: PRIV15
*May 16 04:02:28.129: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback10
0, changed state to up
*May 16 04:02:28.136: %DMI-5-CONFIG_I: R0/0: nesd: Configured from NETCONF/RESTC
ONF by cisco, transaction-id 201
NEWHOSTNAME>
NEWHOSTNAME>
NEWHOSTNAME>sh ip int br
Interface                IP-Address      OK? Method Status      Protocol
GigabitEthernet1         10.128.207.132  YES DHCP    up          up
Loopback1                 2.2.2.2        YES manual  up          up
Loopback2                 unassigned     YES unset   up          up
Loopback100              100.100.100.100 YES other   up          up
NEWHOSTNAME>
```


Step 4: Attempt to create a new loopback interface with a conflicting IP address

- a. Update the `netconf_data` variable to hold a configuration update that creates a new loopback **111** interface with the same IP address as on loopback 100: 100.100.100.100/32:

```
netconf_data = """
<config>
  <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
    <interface>
      <Loopback>
        <name>111</name>
        <description>TEST1</description>
        <ip>
          <address>
            <primary>
              <address>100.100.100.100</address>
              <mask>255.255.255.0</mask>
            </primary>
          </address>
        </ip>
      </Loopback>
    </interface>
  </native>
</config>
"""
```

- b. Attempt to add the new loopback 111 interface by editing the existing device configuration using the `"edit_config()"` function:

```
netconf_reply = m.edit_config(target="running", config=netconf_data)
print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())
```

- c. Before executing the updated Python script, check using `"show ip int brief"` and `"show int desc"` the existing loopback interface by connecting to the console of the IOS XE VM. Take a screenshot of the

result.

```

ETW-CSR1000v [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
NEWHOSTNAME>
NEWHOSTNAME>
NEWHOSTNAME>
*May 16 04:02:27.468: %DMI-5-AUTH_PASSED: R0/0: dmiauthd: User 'cisco' authenticated successfully from 10.128.207.104:56266 and was authorized for netconf over ssh. External groups: PRIV15
*May 16 04:02:28.129: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback100, changed state to up
*May 16 04:02:28.136: %DMI-5-CONFIG_I: R0/0: nesd: Configured from NETCONF/RESTCONF by cisco, transaction-id 201
NEWHOSTNAME>
NEWHOSTNAME>
NEWHOSTNAME>sh ip int br
Interface IP-Address OK? Method Status Protocol
GigabitEthernet1 10.128.207.132 YES DHCP up up
Loopback1 2.2.2.2 YES manual up up
Loopback2 unassigned YES unset up up
Loopback100 100.100.100.100 YES other up up
NEWHOSTNAME>show int desc
Interface Status Protocol Description
Gi1 up up UBox
Lo1 up up WHATEVER
Lo2 up up NewButSame
Lo100 up up TEST1
NEWHOSTNAME>_

```

- d. Execute the Python script and explore the output:

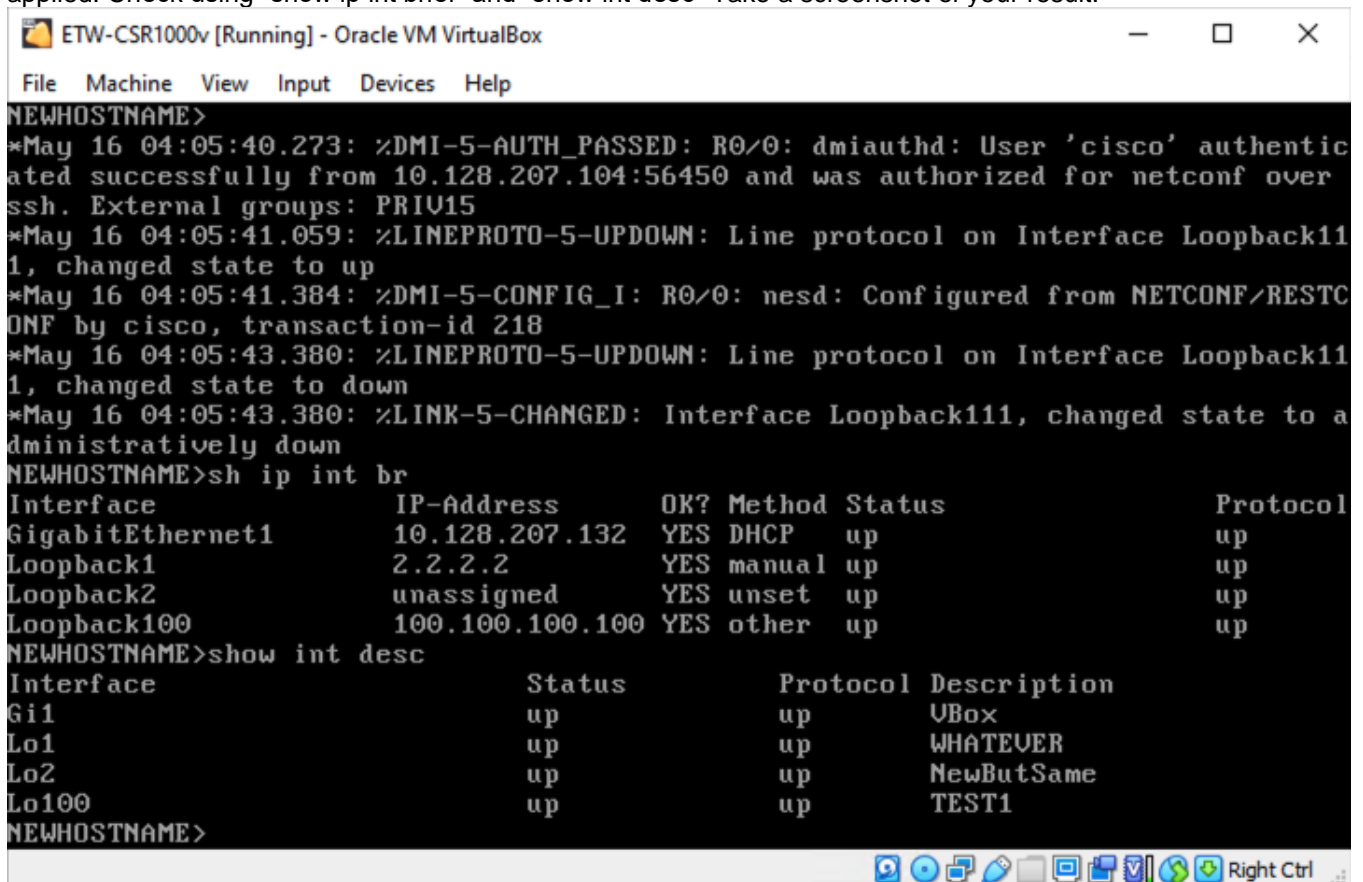
```

>>>
RESTART: B:/DevNetAcad PS-E/Workshops/ETW3 Model Driven Programmability/Python
Files with Solutions/idle/lab 2.6 part 2.py
Traceback (most recent call last):
  File "B:/DevNetAcad PS-E/Workshops/ETW3 Model Driven Programmability/Python Fi
les with Solutions/idle/lab 2.6 part 2.py", line 42, in <module>
    netconf_reply = m.edit_config(target="running", config=netconf_data)
  File "C:\Users\jjanitor\AppData\Local\Programs\Python\Python36\lib\site-packag
es\ncclient\manager.py", line 216, in execute
    raise_mode=self._raise_mode).request(*args, **kwargs)
  File "C:\Users\jjanitor\AppData\Local\Programs\Python\Python36\lib\site-packag
es\ncclient\operations\edit.py", line 67, in request
    return self._request(node)
ncclient.operations.rpc.RPCError: inconsistent value: Device refused one or more
commands
>>>

```

The device has refused one or more configuration settings. With NETCONF, thanks to the transactional behavior, no partial configuration change has been applied but the whole transaction was canceled.

- e. After executing the Python script, verify that no configuration changes, not even partial have been applied: Check using “show ip int brief” and “show int desc” Take a screenshot of your result.



```

NEWHOSTNAME>
*May 16 04:05:40.273: %DMI-5-AUTH_PASSED: R0/0: dmiauthd: User 'cisco' authentic
ated successfully from 10.128.207.104:56450 and was authorized for netconf over
ssh. External groups: PRIV15
*May 16 04:05:41.059: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback11
1, changed state to up
*May 16 04:05:41.384: %DMI-5-CONFIG_I: R0/0: nesd: Configured from NETCONF/RESTC
ONF by cisco, transaction-id 218
*May 16 04:05:43.380: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback11
1, changed state to down
*May 16 04:05:43.380: %LINK-5-CHANGED: Interface Loopback111, changed state to a
dministratively down
NEWHOSTNAME>sh ip int br
Interface                IP-Address      OK? Method Status      Protocol
GigabitEthernet1        10.128.207.132  YES DHCP    up          up
Loopback1                2.2.2.2         YES manual  up          up
Loopback2                unassigned      YES unset   up          up
Loopback100             100.100.100.100 YES other   up          up
NEWHOSTNAME>show int desc
Interface                Status          Protocol Description
Gi1                      up              up           UBox
Lo1                      up              up           WHATEVER
Lo2                      up              up           NewButSame
Lo100                   up              up           TEST1
NEWHOSTNAME>

```