# LAB – CREATING COMPARISONS

## OBJECTIVE

In this lab, you will use simple comparisons and regular expressions to evaluate network device information.

## PART 1

Open a terminal and switch to the lab directory

### STEP 1: OPEN A TERMINAL WINDOW

Double-click the Terminal icon on the desktop to open the terminal window for use in this lab.

### STEP 2: CHANGE DIRECTORY

Change to the directory **labs/prne/** in the user home directory, which holds the files for the course labs.

```
~$ cd labs/prne/
```

## PART 2

Open **Visual Studio Code**, create a new file and save it with a filename of **creating-comparisons-part-2.py**, ensuring to save the file in the **~/labs/prne/** directory, as otherwise the code will require modification to find the associated files that are used. This python application will read the device information from the file, and identifies any devices running incorrect software versions.

### STEP 1: DISPLAY A HEADING

Display a header to describe what this application shows.

```
# Display heading
print('')
print('Devices with bad software versions')
print('----------------------------------')
```

### STEP 2: CREATE VARIABLE FOR CHECKING

Set the variable for the correct software version to be used when checking against the list of devices.

```
# Set Variable for current version comparison used in Step 4
current_version = 'Version 5.3.1'
```

## STEP 3: OPEN FILE FOR READING

Open the file **devices-06.txt** for reading and for each device enter the information into a list.

```
# Read all lines of device information from file
file = open('devices-06.txt', 'r')
for line in file:

    # Put device info into list
    device_info_list = line.strip().split(',')
```

## STEP 4: CREATE DICTIONARY

For each device, take the device information and put it into a dictionary, with fields for name, os-type, ip-address, version, username, and password.

```
    # Put device information into dictionary for this one device
    device_info = {}  # Create the inner dictionary of device info
    device_info['name'] = device_info_list[0]
    device_info['ip'] = device_info_list[2]
    device_info['version'] = device_info_list[3]
```

## STEP 5: COMPARE AND DISPLAY

For each device, compare the version to see if it is out of date, using a simple comparison of the string. Display a table of devices whose version does not match the given 'current version' 5.3.1.

```
    # If the version doesn't match our 'current version',
    # display a warning
    if device_info['version'] != current_version:
        print(' Device:', device_info['name'],
              '   Version:', device_info['version'])

# Display a blank line to make easier to read
print('')
```

## STEP 6: CLOSE FILE

Close the file.

```
# Close the file
file.close()
```

## STEP 7: SAVE, RUN AND VERIFY APPLICATION

Save you application and then run it from the terminal rather than from within visual studio code.

```
~/labs/prne$ python3 creating-comparisons-part-2.py
```

The output from your application will be displayed in your terminal window, verify that it is comparable to below.

```
devasc@labvm:~/labs/prne$ python3 creating-comparisons-part-2.py

Devices with bad software versions
----------------------------------
 Device: d02-is    Version: Version 4.22.18
 Device: d05-xr    Version: Version 4.16.9
 Device: d06-xr    Version: Version 5.3.0
 Device: d07-xe    Version: Version 4.16.0
 Device: d08-xe    Version: Version 5.3.0
```

## STEP 8: CHALLENGE (OPTIONAL)

Add a line to the output that shows how many devices were checked for the correct software version, your output should be comparable to below.

```
devasc@labvm:~/labs/prne$ python3 creating-comparisons-part-2.py

Devices with bad software versions
----------------------------------
 Device: d02-is    Version: Version 4.22.18
 Device: d05-xr    Version: Version 4.16.9
 Device: d06-xr    Version: Version 5.3.0
 Device: d07-xe    Version: Version 4.16.0
 Device: d08-xe    Version: Version 5.3.0

Number of devices checked:  8
```

# PART 3

Open **Visual Studio Code**, create a new file and save it with a filename of **creating-comparisons-part-3.py**, ensuring to save the file in the **~/labs/prne/** directory, as otherwise the code will require modification to find the associated files that are used. This python application will read the device information from the file, and extracts the IP address by using a regular expression.

## STEP 1: IMPORT RE

Import the regular expression module to enable the use of regular expressions.

```
# Import required modules/packages/library
import re
```

## STEP 2: DISPLAY A HEADING

Display a header to describe what this application shows.

```
# Display heading
print('')
print('Devices and their Management IP addresses')
print('-----------------------------------------')
```

## STEP 3: CREATE REGULAR EXPRESSION

Create the regular expression to be used when checking against the list of devices.

```
# Create regular expression to find the Mgmt IP address
ip_addr_pattern = re.compile(r'Mgmt:(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})')
```

## STEP 4: OPEN FILE FOR READING

Open the file **devices-06.txt** for reading and for each device enter the information into a list.

```
# Read all lines of device information from file
file = open('devices-06.txt', 'r')
for line in file:

    # Put device info into list
    device_info_list = line.strip().split(',')
```

## STEP 5: CREATE DICTIONARY

For each device, take the device name and put it into a dictionary. Do not store off the IP address – you will use a regular expression to do this.

```
    # Put device information into dictionary for this one device
    device_info = {}  # Create the inner dictionary of device info
    device_info['name'] = device_info_list[0]
```

## STEP 6: EXTRACT IP ADDRESS

For each device, use a regular expression to extract the IP address. Print the device information.

```
    # Find the Mgmt IP address from the line in the file,
    # and put it into device_info
    mgmt_addr = ip_addr_pattern.search(line)
    device_info['ip'] = mgmt_addr.group(1)
```

## STEP 7: DISPLAY DETAILS

For each device, display a table of device names with the management IP address.

```
    # Display device and management IP address
    print(' Device:', device_info['name'],
          '   Mgmt IP:', device_info['ip'])

# Display a blank line to make easier to read
print('')
```

## STEP 8: CLOSE FILE

Close the file.

```
# Close the file
file.close()
```

## STEP 9: SAVE, RUN AND VERIFY APPLICATION

Save you application and then run it from the terminal rather than from within visual studio code.

```
~/labs/prne$ python3 creating-comparisons-part-3.py
```

The output from your application will be displayed in your terminal window, verify that it is comparable to below.

```
devasc@labvm:~/labs/prne$ python3 creating-comparisons-part-3.py

Devices and their Management IP addresses
-----------------------------------------
 Device: d01-is    Mgmt IP: 192.168.122.1
 Device: d02-is    Mgmt IP: 192.168.122.2
 Device: d03-nx    Mgmt IP: 192.168.122.3
 Device: d04-nx    Mgmt IP: 192.168.122.4
 Device: d05-xr    Mgmt IP: 192.168.122.5
 Device: d06-xr    Mgmt IP: 192.168.122.6
 Device: d07-xe    Mgmt IP: 192.168.122.7
 Device: d08-xe    Mgmt IP: 192.168.122.8
```

# PART 4 (OPTIONAL BUT HIGHLY RECOMMENDED)

As this lab is completed in NETLAB+ and your code files will be erased when the reservation ends, it is advisable to save your files in GitHub under your repository for this course.