Systems & Network Training
Simplifying New Technology

Python for network engineers

A five day course

Administration

Toilets

Messages
& Phone calls

Tea / Coffee

Lunch

Fire
Exits

Time Keeping

Presentation
Material

Questions

Personal
Introductions

Course objectives

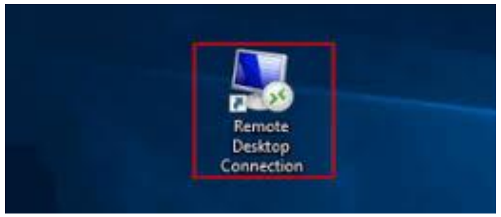
By the end of the course you will be able to:

- ✓ Run Python programs.
- ✓ Read Python programs.
- ✓ Write Python programs.
- ✓ Debug Python programs.
- ✓ Automate network tasks with Python programs.
- ✓ Configure network devices with Python.
- ✓ Collect data from network devices with Python.

Table of contents

Platform for the course

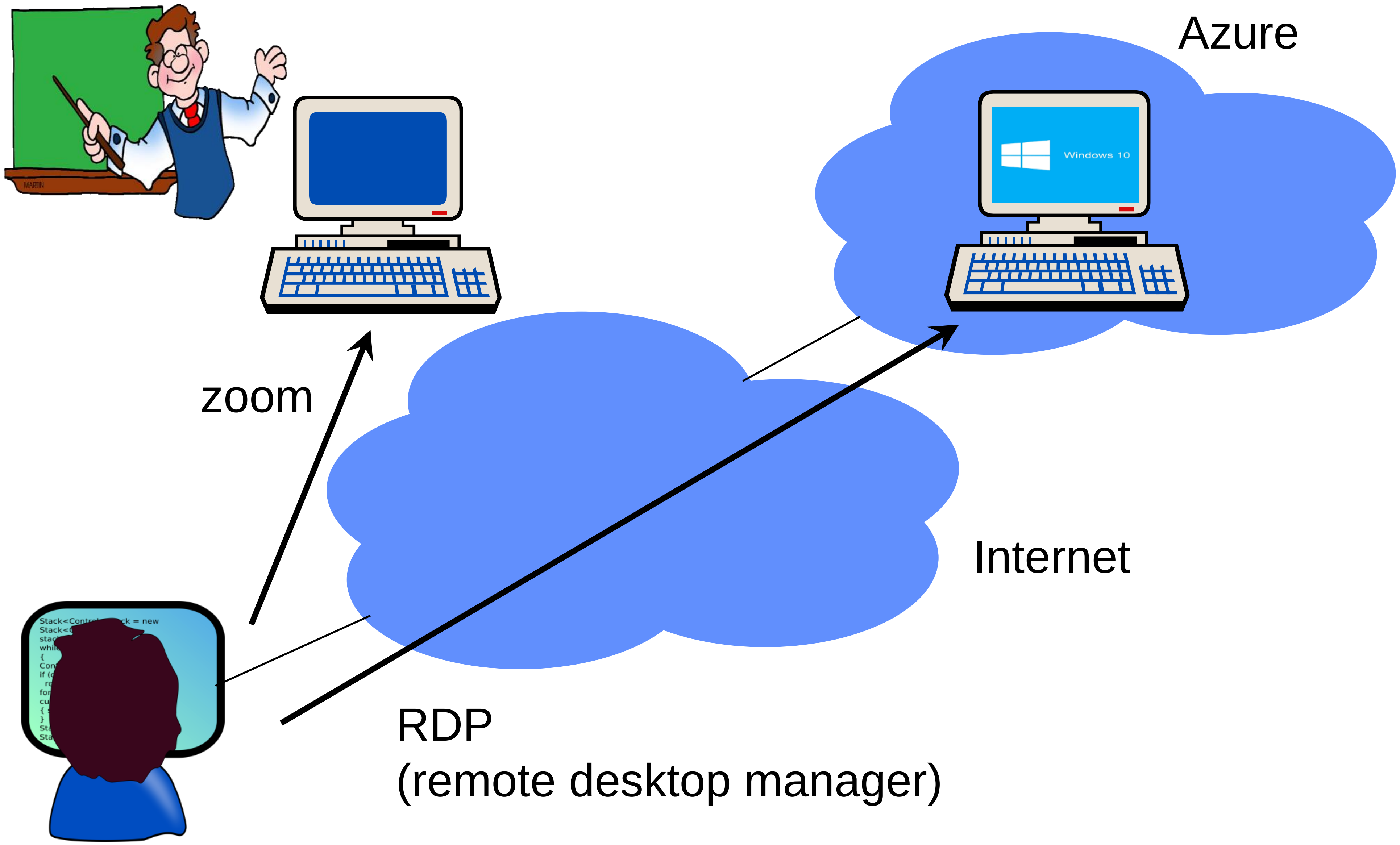
python™



Emulated Virtual Environment
Next Generation

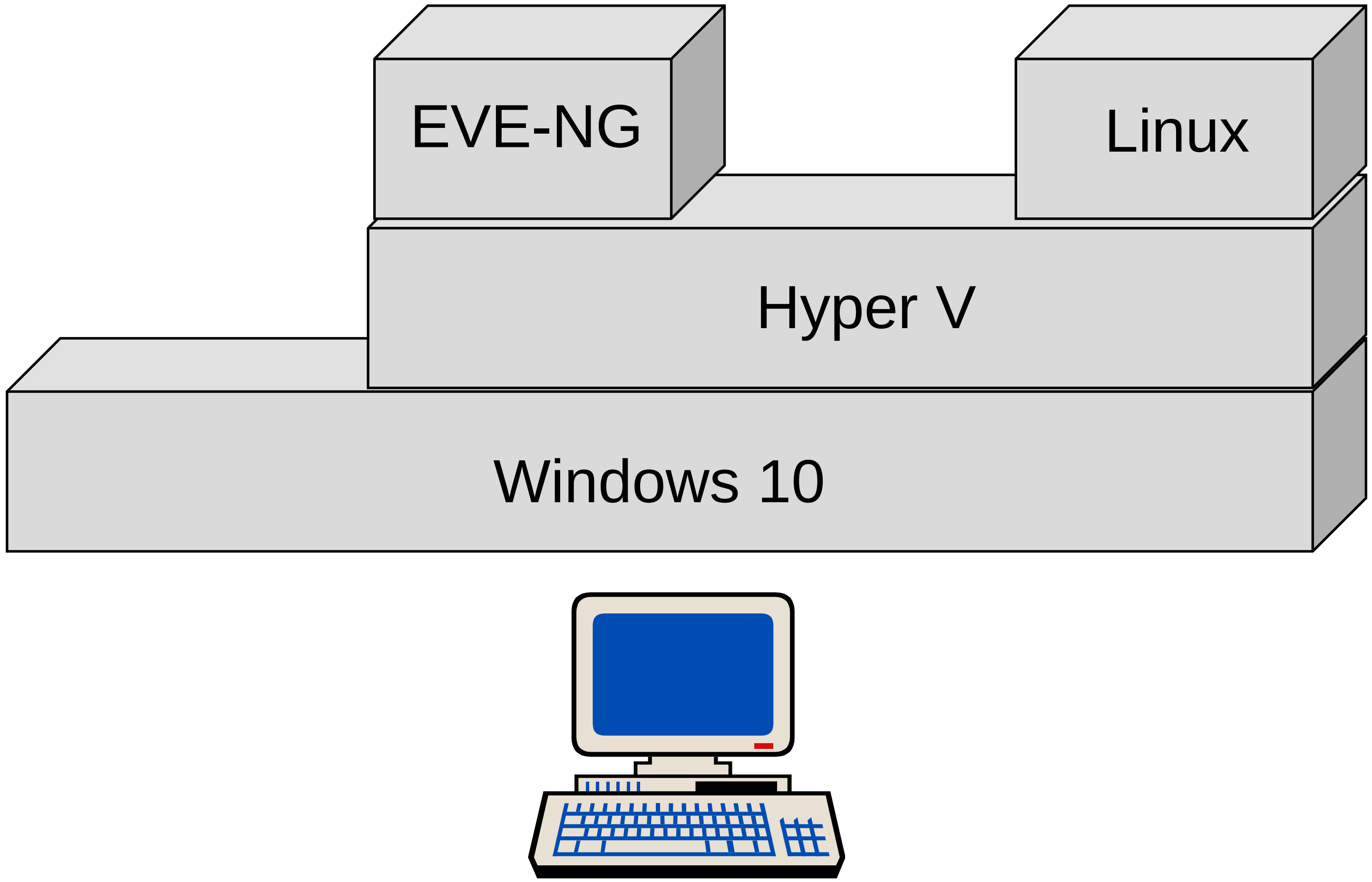
Systems & Network Training
Simplifying New Technology

Platform for hands on



Systems & Network Training
Simplifying New Technology

Windows 10 setup



Systems & Network Training
Simplifying New Technology

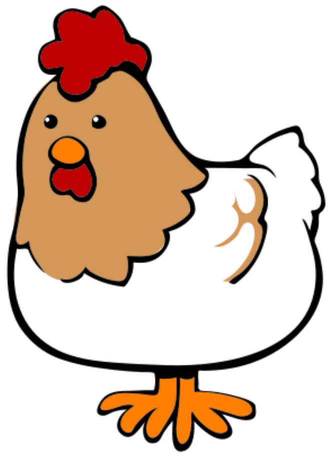
Code and other files for the course

https://github.com/snt000/p4ne-class

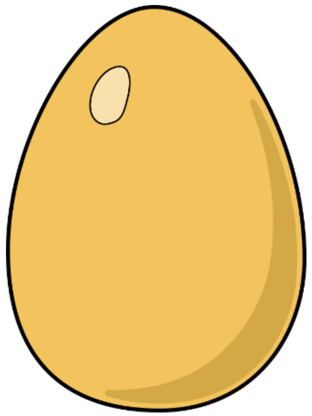
Network devices require base configuration

Systems & Network Training
Simplifying New Technology

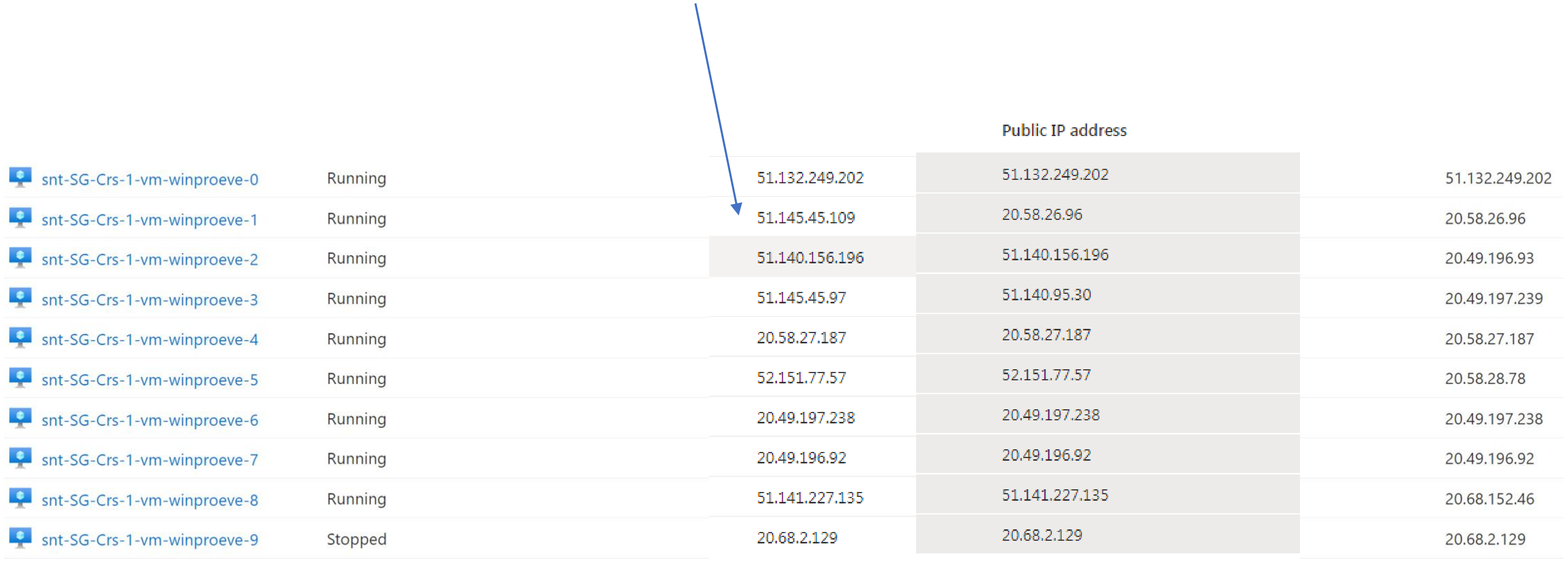
Hostname
Password
IP address
Default gateway
System management protocol (SSH)



ZTP
POAP



The VMs



			Public IP address	
snt-SG-Crs-1-vm-winproeve-0	Running	51.132.249.202	51.132.249.202	51.132.249.202
snt-SG-Crs-1-vm-winproeve-1	Running	51.145.45.109	20.58.26.96	20.58.26.96
snt-SG-Crs-1-vm-winproeve-2	Running	51.140.156.196	51.140.156.196	20.49.196.93
snt-SG-Crs-1-vm-winproeve-3	Running	51.145.45.97	51.140.95.30	20.49.197.239
snt-SG-Crs-1-vm-winproeve-4	Running	20.58.27.187	20.58.27.187	20.58.27.187
snt-SG-Crs-1-vm-winproeve-5	Running	52.151.77.57	52.151.77.57	20.58.28.78
snt-SG-Crs-1-vm-winproeve-6	Running	20.49.197.238	20.49.197.238	20.49.197.238
snt-SG-Crs-1-vm-winproeve-7	Running	20.49.196.92	20.49.196.92	20.49.196.92
snt-SG-Crs-1-vm-winproeve-8	Running	51.141.227.135	51.141.227.135	20.68.152.46
snt-SG-Crs-1-vm-winproeve-9	Stopped	20.68.2.129	20.68.2.129	20.68.2.129

Exercise

Go to https://github.com/snt000/p4ne

Follow steps in first connect guide to connect to your Windows 10 machine

We will stop at EVE-NG

Note you can ping each others machines

Look at ipconfig

Chapter 2: What is Python?

By the end of the chapter you will be able to:

✓ Describe Python.

Python is a programming language

```
print (“Hello world”)
```

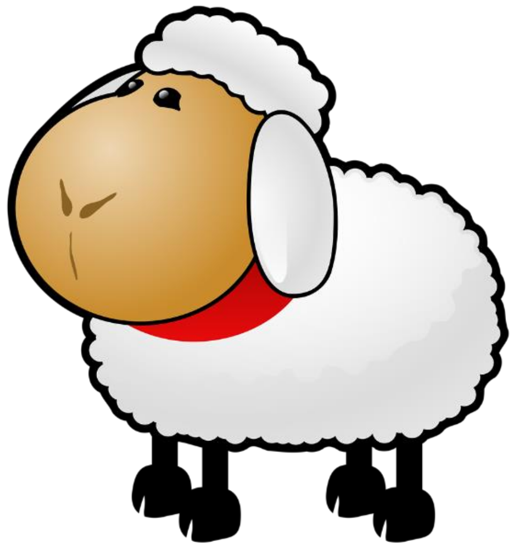
Systems & Network Training
Simplifying New Technology

Programming languages

Programming Language	2018	2013	2008	2003	1998	1993	1988
Java	1	2	1	1	17	-	-
C	2	1	2	2	1	1	1
C++	3	4	3	3	2	2	4
Python	4	7	6	11	24	13	-
C#	5	5	7	8	-	-	-
Visual Basic .NET	6	11	-	-	-	-	-
PHP	7	6	4	5	-	-	-
JavaScript	8	9	8	7	21	-	-
Ruby	9	10	9	18	-	-	-
R	10	23	48	-	-	-	-
Objective-C	14	3	40	50	-	-	-
Perl	16	8	5	4	3	9	22
Ada	29	19	18	15	12	5	3
Lisp	30	12	16	13	8	6	2
Fortran	31	24	21	12	6	3	15

source "www.tiobe.com"

Why Python for network automation?



Systems & Network
Simplifying New Technology Training

What is Python?

High level programming language

General purpose

Interpreted

Systems & Network
Simplifying New Technology Training

Python interactive mode

```
#python3
python 3.6.0 (default, Jan 13 2017, 00:00:00) [GCC 4.8.4] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print (“hello world”)
>>> quit()
```

```
root@NetworkAutomation-1:~# python
Python 2.7.12 (default, Nov 19 2016, 06:48:10)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> print “hello world”
>>> #quit() or ctrl d can be used in either version
>>> ctrl-d
```

Systems & Network Training
Simplifying New Technology

Running “proper” Python scripts

filename.py

```
# python filename.py
```

```
print (“hello world”)
```

filename.py

```
# filename.py
```

```
#!/usr/bin/python3
print (“hello world”)
```

Python version

2.7

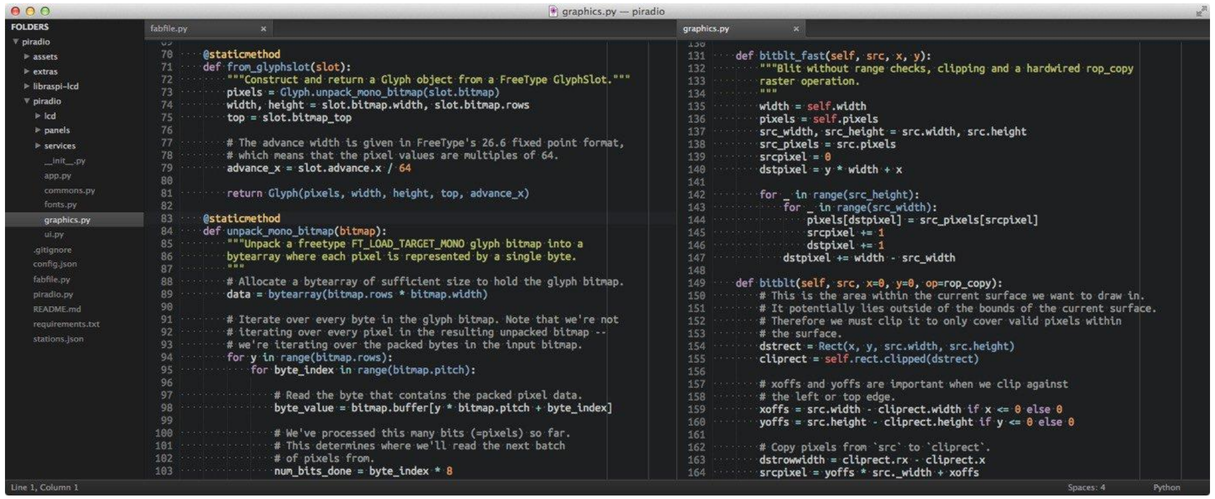
3

Which platform?

A simple python script

```
#Comments begin with a #
#print “Strings are printed with print statement”
print (“Strings are printed with print statement”)

#A variable example
a=“hello world”
#print a
print (a)
```

Editors and IDEs

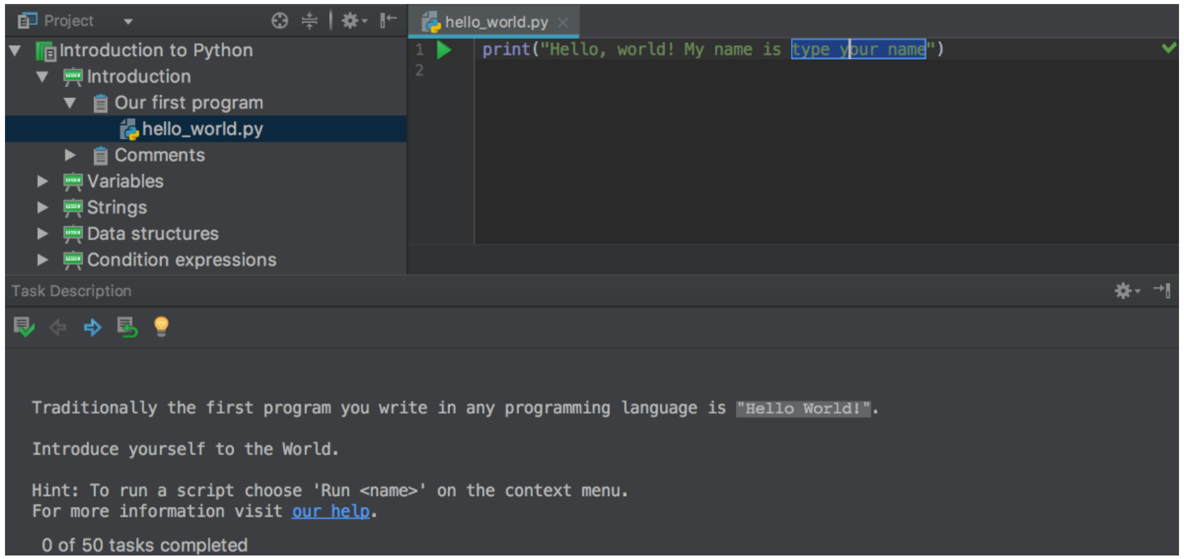


```python
    @staticmethod
    def from_glyphslot(slot):
        """Construct and return a Glyph object from a FreeType GlyphSlot."""
        pixels = Glyph.unpack_mono_bitmap(slot.bitmap)
        width, height = slot.bitmap.width, slot.bitmap.rows
        top = slot.bitmap_top

        # The advance width is given in FreeType's 26.6 fixed point format,
        # which means that the pixel values are multiples of 64.
        advance_x = slot.advance.x / 64

        return Glyph(pixels, width, height, top, advance_x)

    @staticmethod
    def unpack_mono_bitmap(bitmap):
        """Unpack a freetype FT_LOAD_TARGET_MONO glyph bitmap into a
        bytearray where each pixel is represented by a single byte.
        """
        # Allocate a bytearray of sufficient size to hold the glyph bitmap.
        data = bytearray(bitmap.rows * bitmap.width)

        # Iterate over every byte in the glyph bitmap. Note that we're not
        # iterating over every pixel in the resulting unpacked bitmap --
        # we're iterating over the packed bytes in the input bitmap.
        for y in range(bitmap.rows):
            for byte_index in range(bitmap.pitch):

                # Read the byte that contains the packed pixel data.
                byte_value = bitmap.buffer[y * bitmap.pitch + byte_index]

                # We've processed this many bits (=pixels) so far.
                # This determines where we'll read the next batch
                # of pixels from.
                num_bits_done = byte_index * 8
```

```python
    def bitblt_fast(self, src, x, y):
        """Blit without range checks, clipping and a hardwired rop_copy
        raster operation.
        """
        width = self.width
        pixels = self.pixels
        src_width, src_height = src.width, src.height
        src_pixels = src.pixels
        srcpixel = 0
        dstpixel = y * width + x

        for _ in range(src_height):
            for _ in range(src_width):
                pixels[dstpixel] = src_pixels[srcpixel]
                srcpixel += 1
                dstpixel += 1
            dstpixel += width - src_width

    def bitblt(self, src, x=0, y=0, op=rop_copy):
        # This is the area within the current surface we want to draw in.
        # It potentially lies outside of the bounds of the current surface.
        # Therefore we must clip it to only cover valid pixels within
        # the surface.
        dstrect = Rect(x, y, src.width, src.height)
        cliprect = self.rect.clipped(dstrect)

        # xoffs and yoffs are important when we clip against
        # the left or top edge.
        xoffs = src.width - cliprect.width if x <= 0 else 0
        yoffs = src.height - cliprect.height if y <= 0 else 0

        # Copy pixels from `src` to `cliprect`.
        dstrowwidth = cliprect.rx - cliprect.x
        srcpixel = yoffs * src._width + xoffs
```

IDEs

Editors



```python
print("Hello, world! My name is type your name")
```

Traditionally the first program you write in any programming language is "Hello World!".

Introduce yourself to the World.

Hint: To run a script choose 'Run <name>' on the context menu.
For more information visit our help.

0 of 50 tasks completed

Python IDEs

Visual Studio Code

PyCharm

Sublime text

Jupyter

vi
vim
nano

Installing git

```
apt-get update
apt-get install git

git config --global user.name “Stephen Groombridge"
git config --global user.email “steve@snt.co.uk"
```

Using github to get files

```
git clone https://github.com/snt000/exercise
```

```
git init
git pull https://github.com/snt000/exercise
```

How to use github: Add to github

Systems & Network Training
Simplifying New Technology

```
git init

git add .

git commit –m “First commit”

git remote add origin https://github.com/snt000/netops-class.git

git push origin master
```

Quiz

1. What is Python?
2. Why use Python for network automation rather than C, C++, Java? (2 reasons)
3. What are the three ways to run Python?
4. Which version of Python should you use?
5. Which platform should you use?
6. Comments begin with a …?
7. Is print a statement or a function?
8. What is the difference between sublime and vi?
9. What is the difference between a code editor and an IDE?

Exercise

```
print("Hello, world!")
```

Chapter 3: EVE-NG

By the end of the chapter you will be able to:

- ✓ Configure EVE-NG.
- ✓ Recognise the role of EVE-NG in network DevOps.

Systems & Network
Simplifying New Technology Training

Network simulators

- GNS3
- VIRL
- EVE-NG

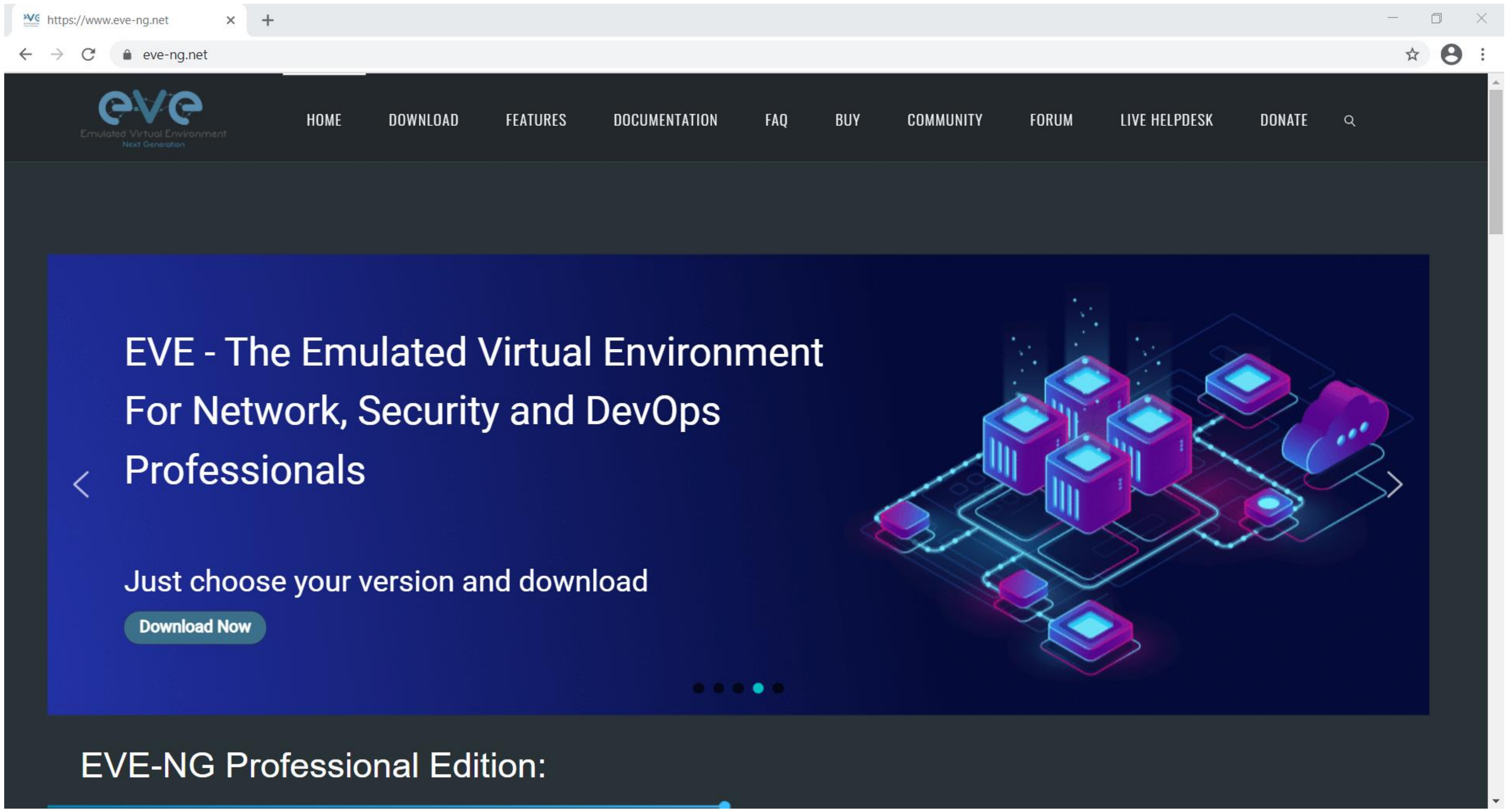
VIRL

Emulated Virtual Environment
Next Generation

Systems Network
Simplifying New Technology Training

What is EVE-NG

Emulated Virtual Environment
Next Generation


https://www.eve-ng.net
eve-ng.net

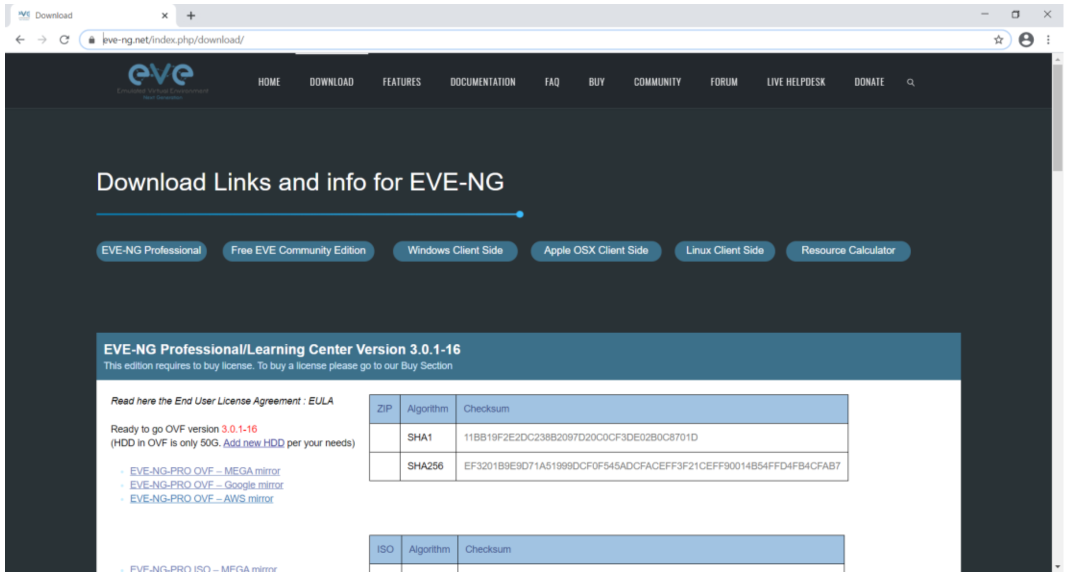
EVE - The Emulated Virtual Environment For Network, Security and DevOps Professionals

Just choose your version and download

EVE-NG Professional Edition:

Systems & Network Training
Simplifying New Technology

Installing EVE-NG



Download Links and info for EVE-NG

EVE-NG Professional/Learning Center Version 3.0.1-16
This edition requires to buy license. To buy a license please go to our Buy Section

Read here the End User License Agreement : EULA

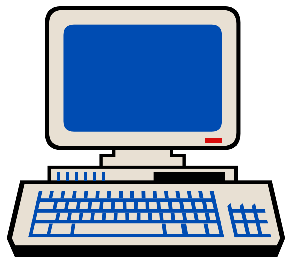
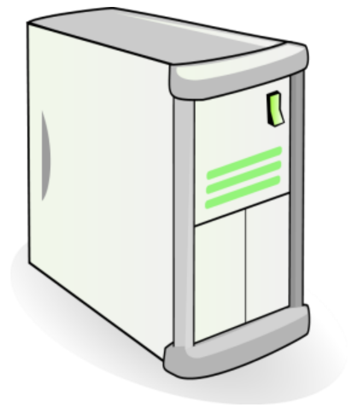
Ready to go OVF version 3.0.1-16
(HDD in OVF is only 50G. Add new HDD per your needs)

- EVE-NG-PRO OVF – MEGA mirror
- EVE-NG-PRO OVF – Google mirror
- EVE-NG-PRO OVF – AWS mirror

ZIP	Algorithm	Checksum
	SHA1	11BB19F2E2DC238B2097D20C0CF3DE02B0C8701D
	SHA256	EF3201B9E9D71A51999DCF0F545ADCFACEFF3F21CEFF90014B54FFD4FB4CFAB7

ISO	Algorithm	Checksum

- EVE-NG-PRO ISO – MEGA mirror

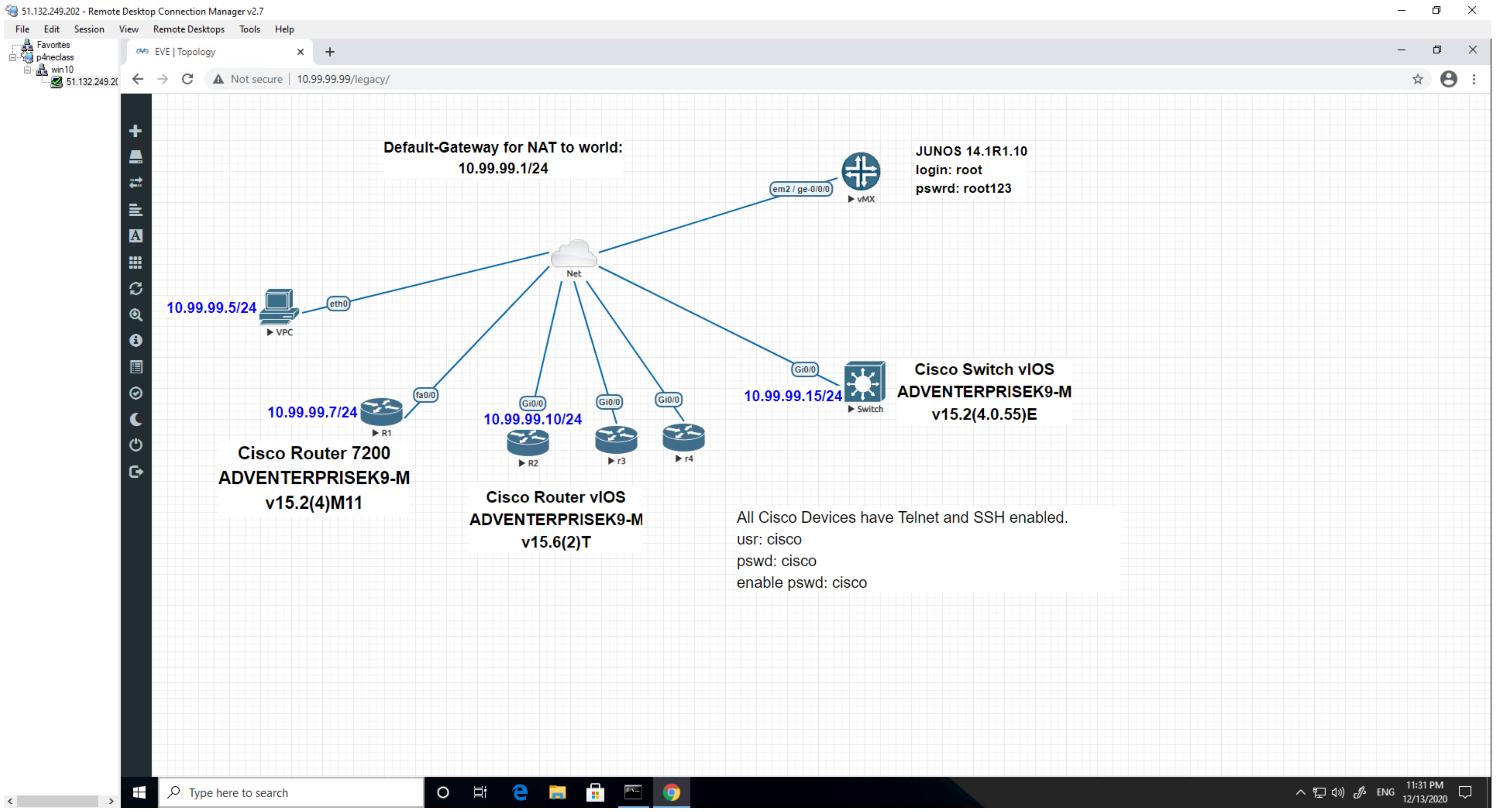



Systems & Network Training
Simplifying New Technology

Quiz

1. What is EVE-NG?
2. What does the NAT cloud do?
3. What operating system does EVE-NG use?

Exercise



Default-Gateway for NAT to world:
10.99.99.1/24

JUNOS 14.1R1.10
login: root
pswrd: root123

10.99.99.5/24

10.99.99.7/24

Cisco Router 7200
ADVENTERPRISEK9-M
v15.2(4)M11

10.99.99.10/24

Cisco Router vIOS
ADVENTERPRISEK9-M
v15.6(2)T

10.99.99.15/24

Cisco Switch vIOS
ADVENTERPRISEK9-M
v15.2(4.0.55)E

All Cisco Devices have Telnet and SSH enabled.
usr: cisco
pswd: cisco
enable pswd: cisco

Exercise: Set IP address in Linux

```
NetworkAutomation-1 console is now available... Press RETURN to get started.
root@NetworkAutomation-1:~# cat /etc/network/interfaces
#
# This is a sample network config uncomment lines to configure the network
#

# Static config for eth0
#auto eth0
#iface eth0 inet static
#       address 192.168.0.2
#       netmask 255.255.255.0
#       gateway 192.168.0.1
#       up echo nameserver 192.168.0.1 > /etc/resolv.conf

# DHCP config for eth0
 auto eth0
 iface eth0 inet dhcp
```

Exercise: Enable ssh on Cisco devices

```
hostname r11
username steve password cisco
username steve privilege 15
line vty 0 4
    login local
    transport input all
    exit
ip domain-name snt.co.uk
crypto key generate rsa
int gi 0/0
ip add 192.168.122.11
no shut
end
copy run start
```

Exercise: Enable ssh on Juniper devices

Systems & Network Training
Simplifying New Technology

```
configure
set system host-name j1
set system login user steve class super-user
    full-name “steve” authentication plain-text-password
set system services ssh
set system root-authentication plain-text-password
set interfaces ge-0/0/0 unit 0 family inet address 10.99.99.30/24
commit and-quit
```

Some more Juniper

```
delete security
set security forwarding-options family
                    mpls mode packet-based
commit and-quit
```

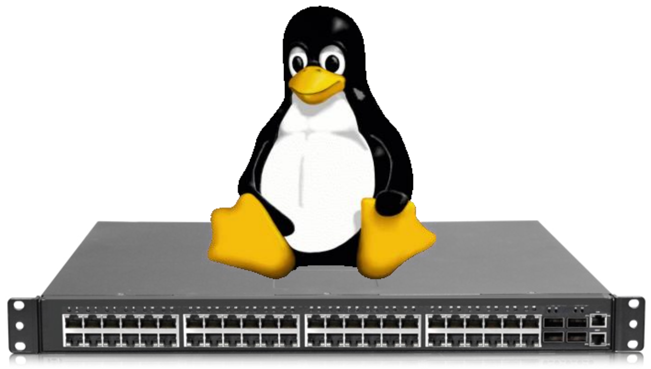
Chapter 4: A network example

By the end of the chapter you will be able to:

- ✓ Recognise the difference between on and off box Python.
- ✓ Recognise the role of APIs.
- ✓ Write a Python script to telnet to a network device.

Systems & Network Training
Simplifying New Technology

Is Linux on your network device?



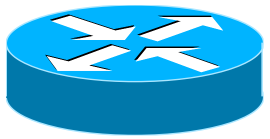
cumulus networks

ARISTA

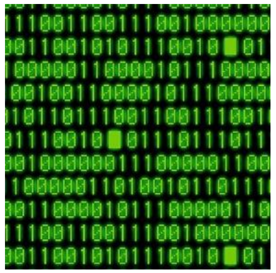
CLI vs API

TELNET/SSH

```
enable
conf t
hostname r1
end
copy run start
```

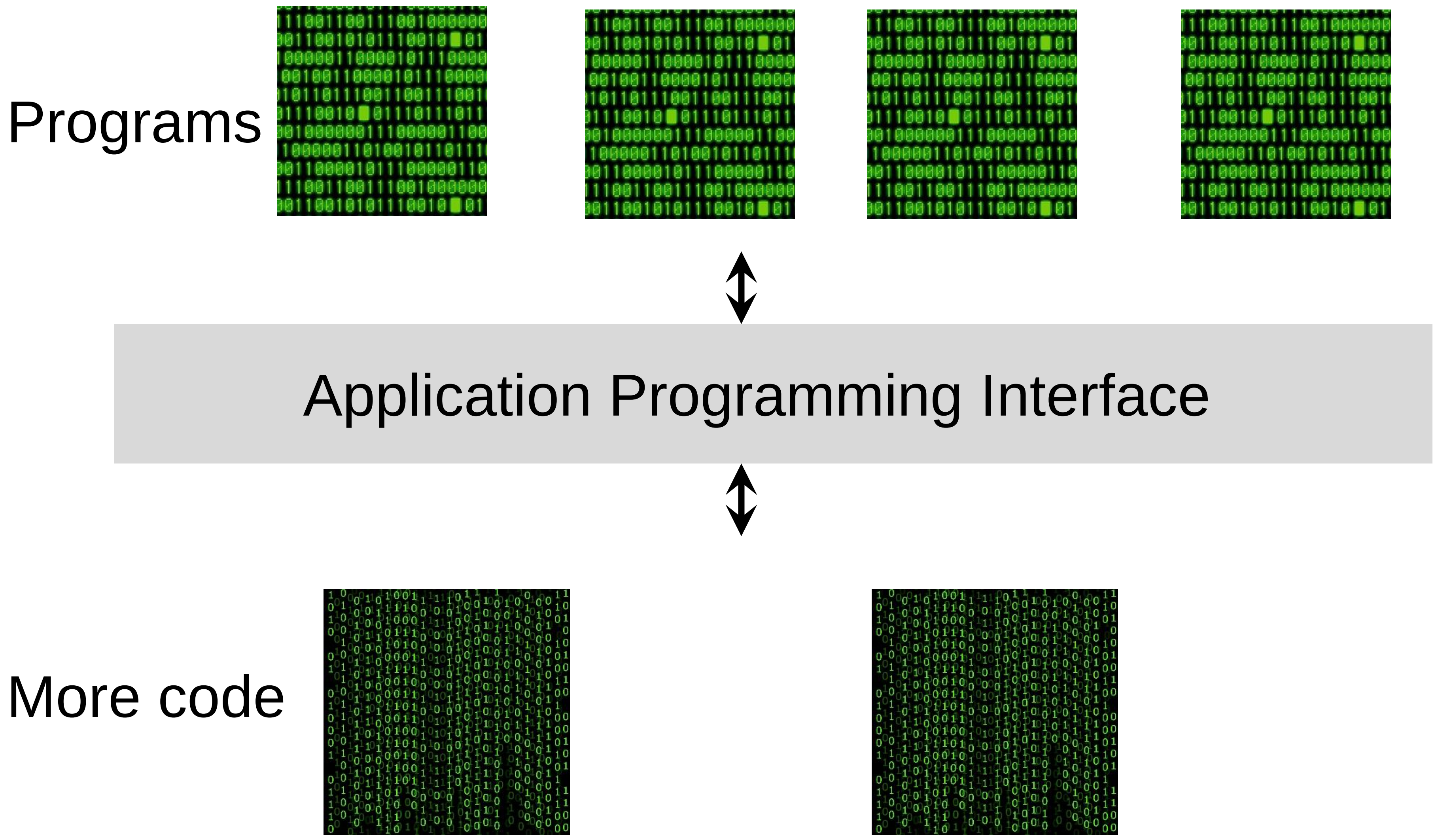


NETCONF/RESTCONF
Proprietary



YANG/XML/JSON…

APIs

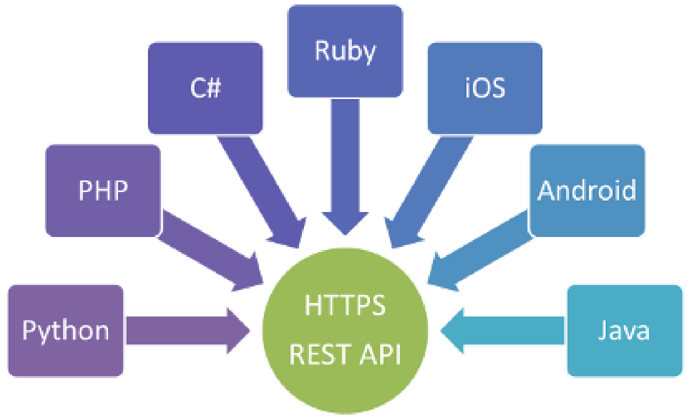


Systems & Network Training
Simplifying New Technology

Network device APIs

Cisco Nexus

NX-API



ARISTA

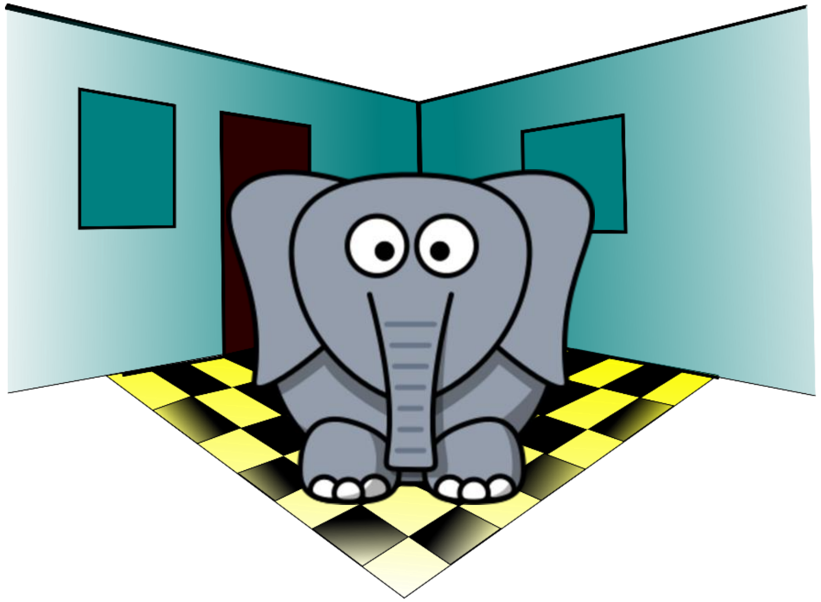
eAPI

I E T F®

NETCONF

Systems & Network Training
Simplifying New Technology

The problem

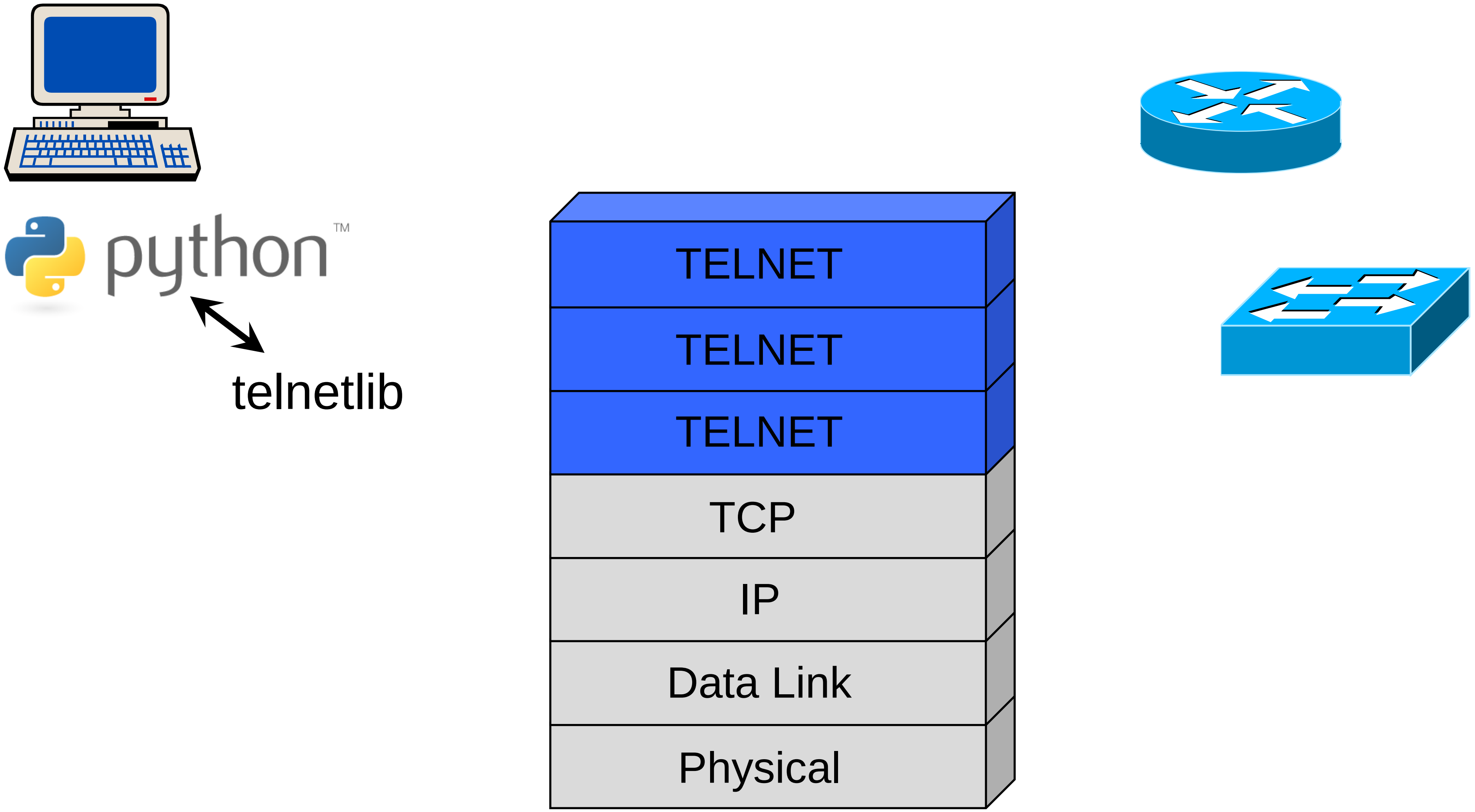


IOS

How to handle

```
if device in [‘NETCONF’, ‘REST’, ‘anyAPI’]:
    use(API)
else:
    use(SSH)
```

telnetlib


python
telnetlib
TELNET
TELNET
TELNET
TCP
IP
Data Link
Physical

Systems & Network Training
Simplifying New Technology

telnetlib Python 2.7

```
import getpass
import sys
import telnetlib

HOST = "localhost"
user = raw_input("Enter your remote account: ")
password = getpass.getpass()

tn = telnetlib.Telnet(HOST)

tn.read_until("login: ")
tn.write(user + "\n")
if password:
    tn.read_until("Password: ")
    tn.write(password + "\n")

tn.write("ls\n")
tn.write("exit\n")

print tn.read_all()
```

telnetlib Python3

```
import getpass
import telnetlib

HOST = "localhost"
user = input("Enter your remote account: ")
password = getpass.getpass()

tn = telnetlib.Telnet(HOST)

tn.read_until(b"login: ")
tn.write(user.encode('ascii') + b"\n")
if password:
    tn.read_until(b"Password: ")
    tn.write(password.encode('ascii') + b"\n")

tn.write(b"ls\n")
tn.write(b"exit\n")

print(tn.read_all().decode('ascii'))
```

Quiz

1. What is the difference between off box and on box python?

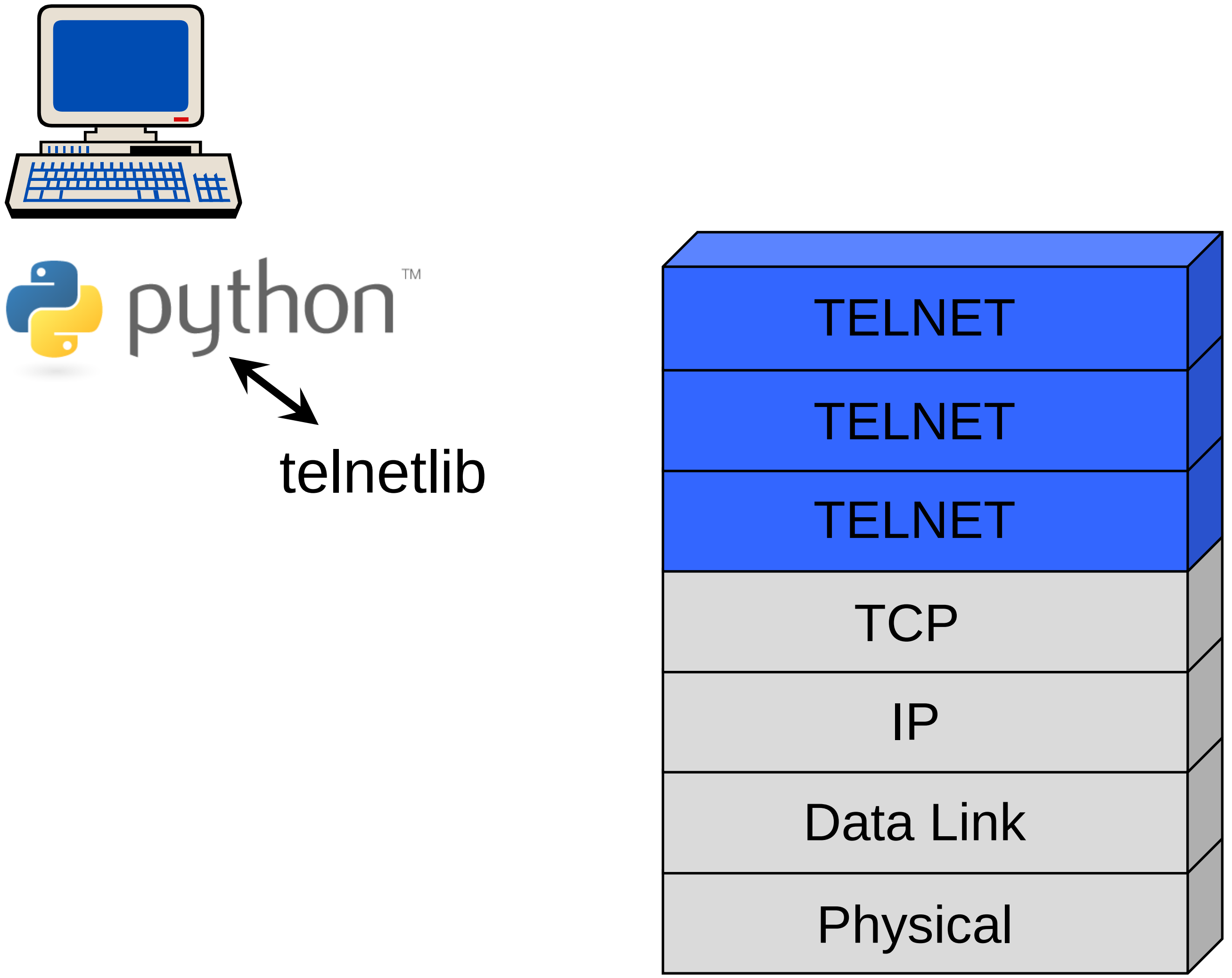
2. What is the difference between CLI and API?

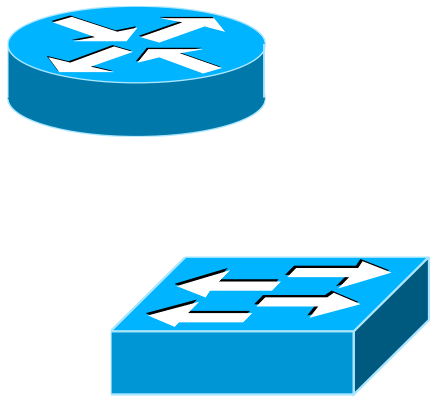
3. Name two network APIs.

4. If there is no API how do you use the CLI remotely?

5. What library is used for telnet access?

Exercise


python
telnetlib
TELNET
TELNET
TELNET
TCP
IP
Data Link
Physical



Chapter 5: Python basics

By the end of the chapter you will be able to:

- ✓ Use variables.
- ✓ Use operators.
- ✓ Use loops and conditionals.

Operators

Mathematical operators	Comparison operators
+	<
-	<=
*	>
/	>=
//	==
%	!=
**	

Boolean operators

and

or

not

Variables

Names are case sensitive
Variables are dynamically typed

```
interfaces = 4
uptime = 1000.4
hostname = “r1”
#print hostname – what error do you get?
print (hostname)
# How is this different from
# print (“hostname”)
```

Try not to hard code values

10.1.1.1 versus host

But if you’re not a full time programmer…..

Data types

- Numbers
  - int, long, bool, float, complex
- Strings
  - ‘ or “
- List/tuples/dictionaries
  - Tuples are immutable lists

Advanced: Everything in Python is an object

Python control statements

```
if condition:
    statements
[elif condition:
    statements]…
[else:
    statements]
```

```
while condition:
    statements

for var in sequence:
    statements
```

```
break
continue
```

Systems & Network Training
Simplifying New Technology

Example for loops

```
for n in range (1,7):
    host=“192.168.122.” + str(n)
    print host


bgplist = ['192.168.122.72',
           '192.168.122.73'
           ]
for ip in bgplist:
    print ("Connecting to " + ip)
```

Indentation matters

PEP8

“Know when to be inconsistent”

4 spaces per level for indentation.

Quiz

1. Do Python variables need a declaration of data type?
2. What marks the begin and end of an if?
3. What are the two ways to denote a string? Why are there two ways?
4. What is PEP8?
5. Should you use spaces or tabs? (Does it matter in an IDE?)

Systems & Network Training
Simplifying New Technology

Exercise: Basic Python

python™

Day 1 review

Chapter 6: Functions, classes and methods

By the end of the chapter you will be able to:

- ✓ Use functions.
- ✓ Use objects.
- ✓ Handle files.

Python functions

Code, written once, can be used many times.

```
>>> def hello (x):
...     text = "Hello, " + x + "!"
...     print text
...
>>> hello ("World")
Hello, World!
```

Systems & Network Training
Simplifying New Technology

Built-in functions

69 of them at the time of writing in Python 3

Already seen
- print()
- input()

Useful
- open()
- range()

Others
- int()
- bin()
- len()
- max()
- sum()

Interesting
- help()
- type()
- dir()

Python file handling

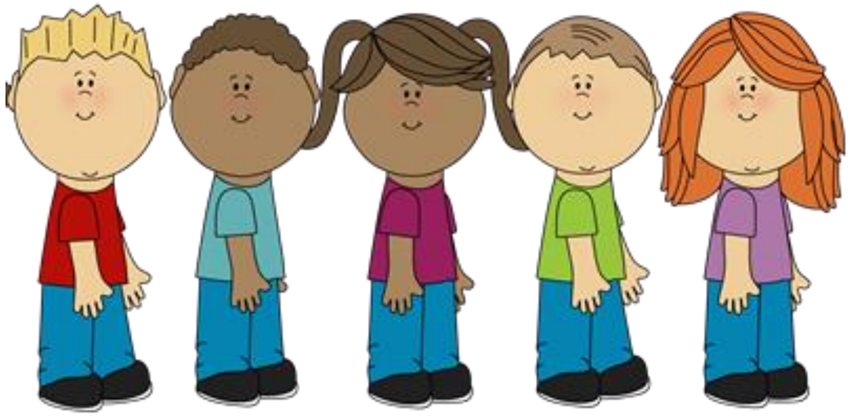
op

```
f = open("hello.txt")
for line in f:
        print line,
f.close()
```

Classes and objects

Class

Objects

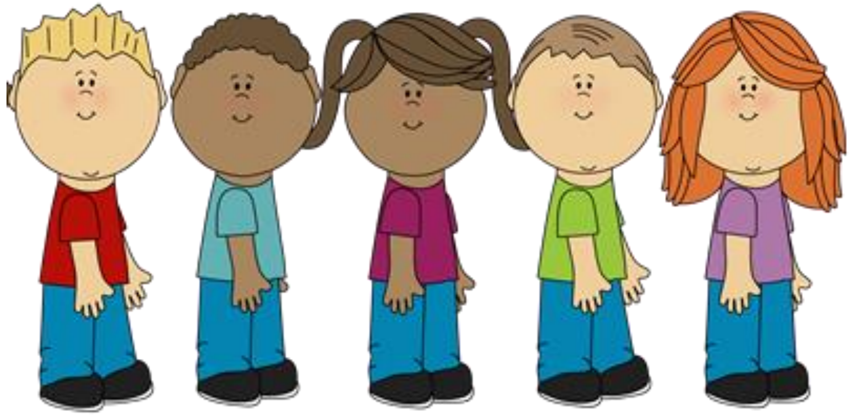


ID:	s1	s2	s3	s4	s5
Name:	Bob	Fred	Ana	Mike	Eve
Age:	8	7	9	8	7
Gender:	M	M	F	M	F

Attributes and methods

Class

Objects



Attributes

ID: s1 s2 s3 s4 s5

Name:
Age:
Gender:

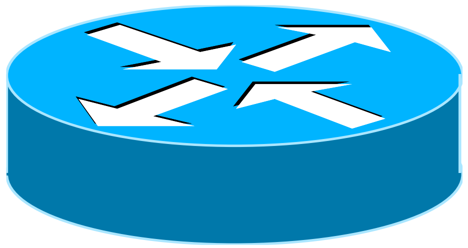
Example attribute use
s1.name

Methods
sleep()
learn()

Example method use
s1.sleep()

Another example

Class: router



Attributes:

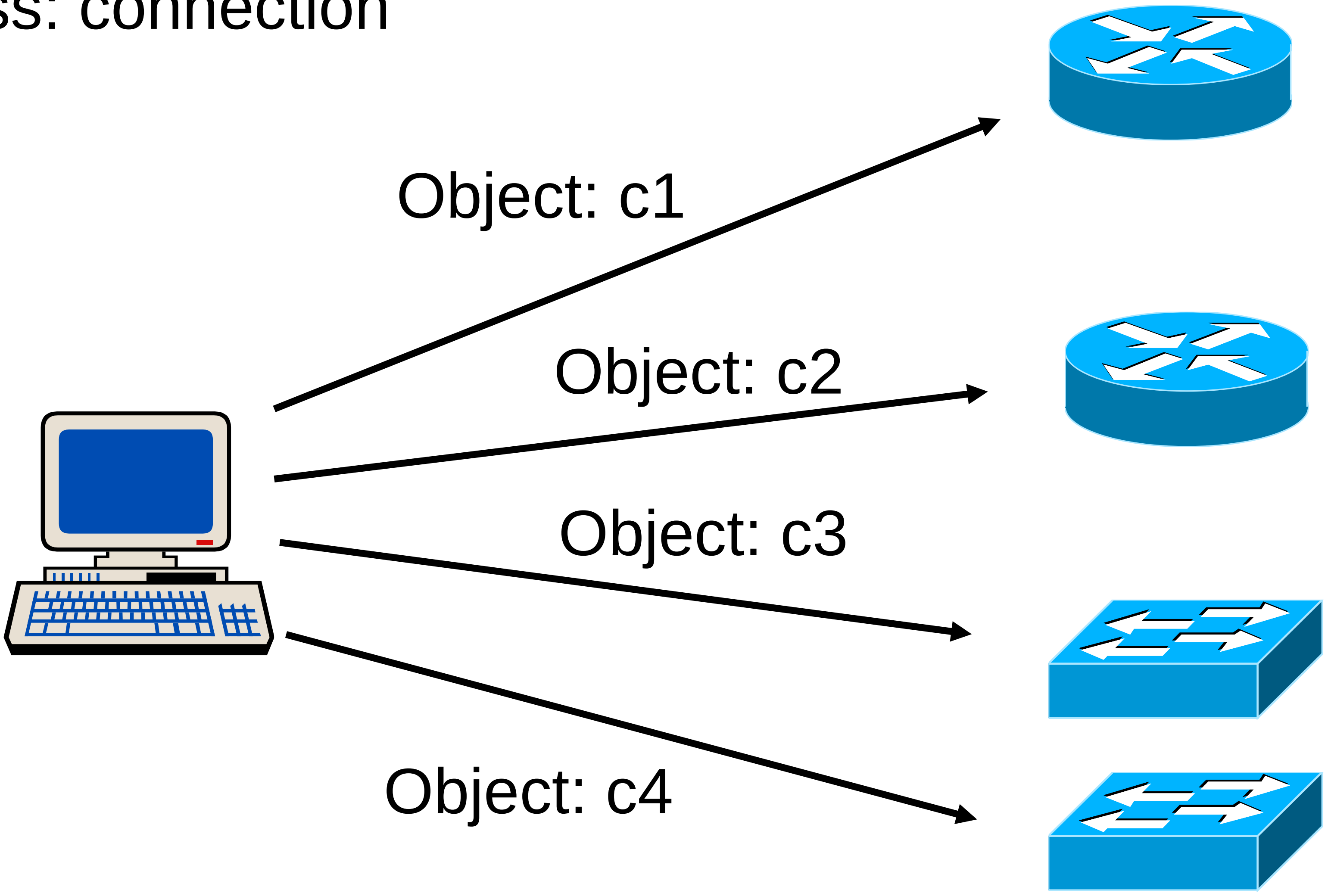
- hostname
- model
- serial number
- location

Methods:

- show(command)
- configure(file)
- reboot()

Network automation example

Class: connection



Python is object oriented

Class
- Data
- Methods

```
>>> class Student:
...     def __init__ (self, name, age, gender):
...         self.name   = name
...         self.age    = age
...         self.gender = gender
...
```

Python instances

Instances

```
>>> Steve = Student(“Steve Groombridge", 50, “m")
>>> print Steve.age
50
```

Python strings

```
s1 = "Hello "
s2 = "World“
print s1 + s2

print (list(enumerate(s1)))
print (len(s1))
print s1.lower()
print s1.upper()
```

```
count = 0
for letter in 'Hello World':
    if(letter == 'l'):
        count += 1
print(count,'letters found')
```

Quiz

1. What is a function?

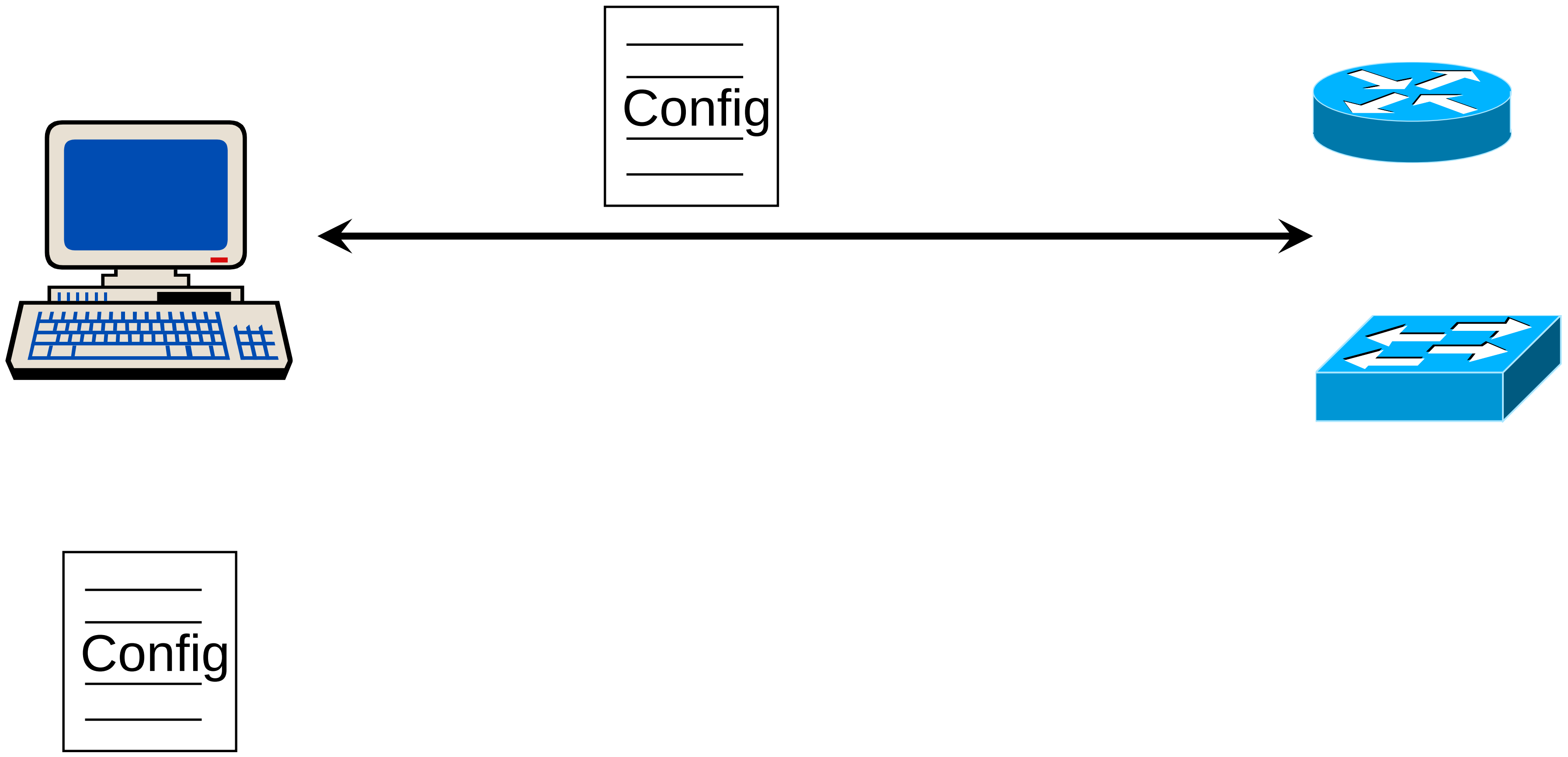
2. Name 3 built in functions.

3. What built-in function allows you to use a file in Python?

4. What is the difference between a class and an object?

5. What are methods and attributes?

Exercise: Functions, classes and methods



Chapter 7: Libraries and modules

By the end of the chapter you will be able to:

- ✓ Explain what ansible is and how it works.
- ✓ Configure network devices with ansible.
- ✓ Troubleshoot network devices with ansible.

Don’t reinvent the wheel

Use other peoples code
(and their time and effort)

Python modules and libraries

Module

file.py

pip – Pip Installs Packages

Package manager for Python

pip list

pip install

pip uninstall

Dependencies

pip handles dependencies

(where a library uses another library)

PyPI

python™
Package
Index

Python network libraries

sockets
telnetlib
pysnmp
ncclient
ciscoconfparse
pyez

Paramiko
Netmiko
pyntc
NAPALM
Nornir

csv library

```
>>> import csv
>>> with open('names.csv') as csvfile:
...     reader = csv.DictReader(csvfile)
...     for row in reader:
...         print(row['first_name'], row['last_name'])
...
Steve Groombridge
Julian James
Michael Connor
```

IP address libraries

Python standard library

- ipaddress

Others

- netaddr
- ipy

Systems & Network Training
Simplifying New Technology

netaddr library

```
>>> import netaddr
>>> mynet = netaddr.ipaddress.ip_network(u’10.1.1.192/30’)
>>> mynet.netmask
IPv4address(u’255.255.255.252’)
```

```
>>> from netaddr import IPAddress
>>> IPAddress(“255.0.0.0”).netmask_bits()
8
```

Quiz

1. List 7 Python network libraries.

2. What is an API?

3. What is the problem with legacy devices?

4. What is telnetlib for?

5. What is pySNMP for?

6. What is ncclient for?

Systems & Network Training
Simplifying New Technology

Exercise: Python networking

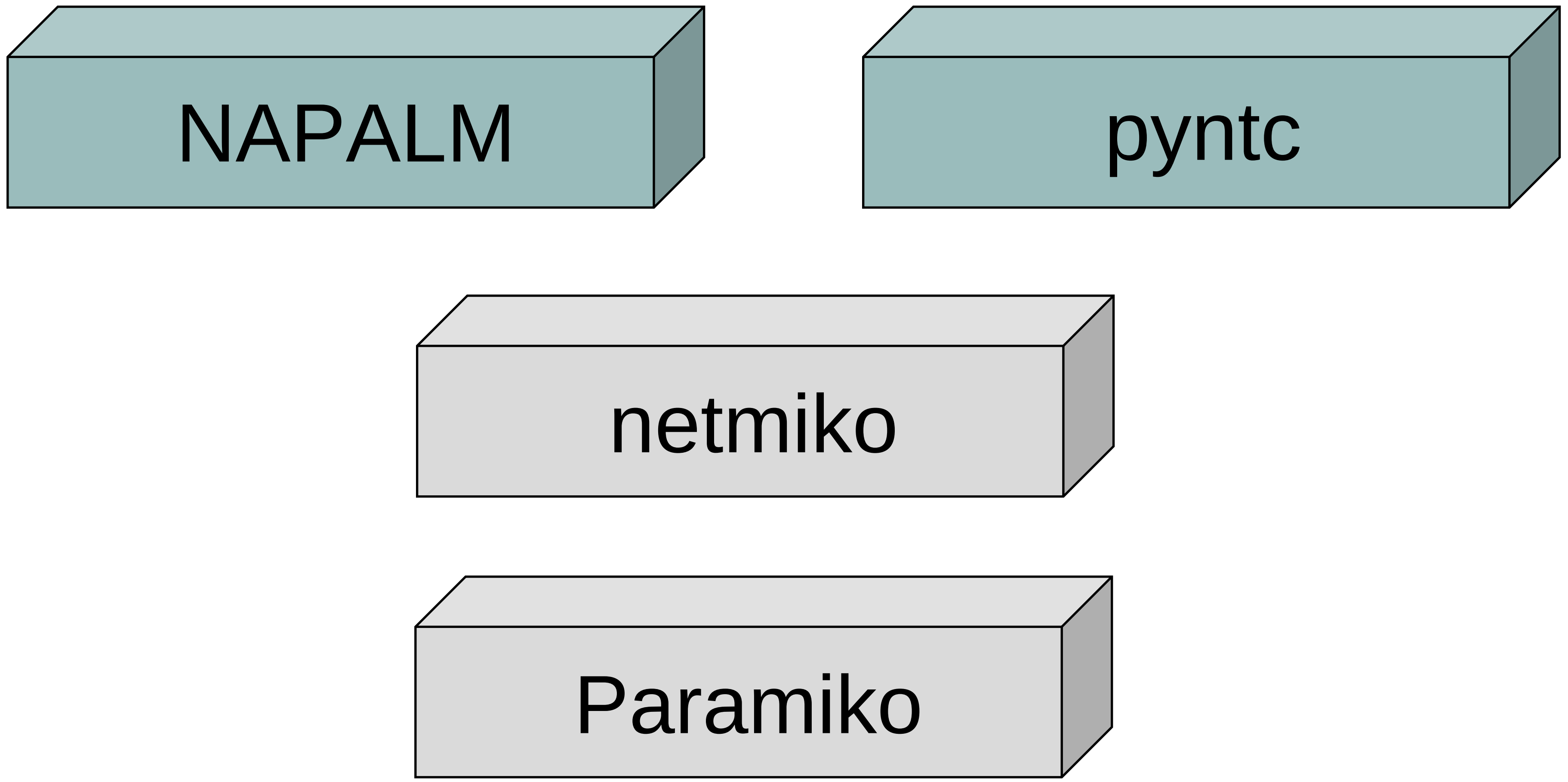
Exercise: Python libraries

Chapter 8: Paramiko and netmiko

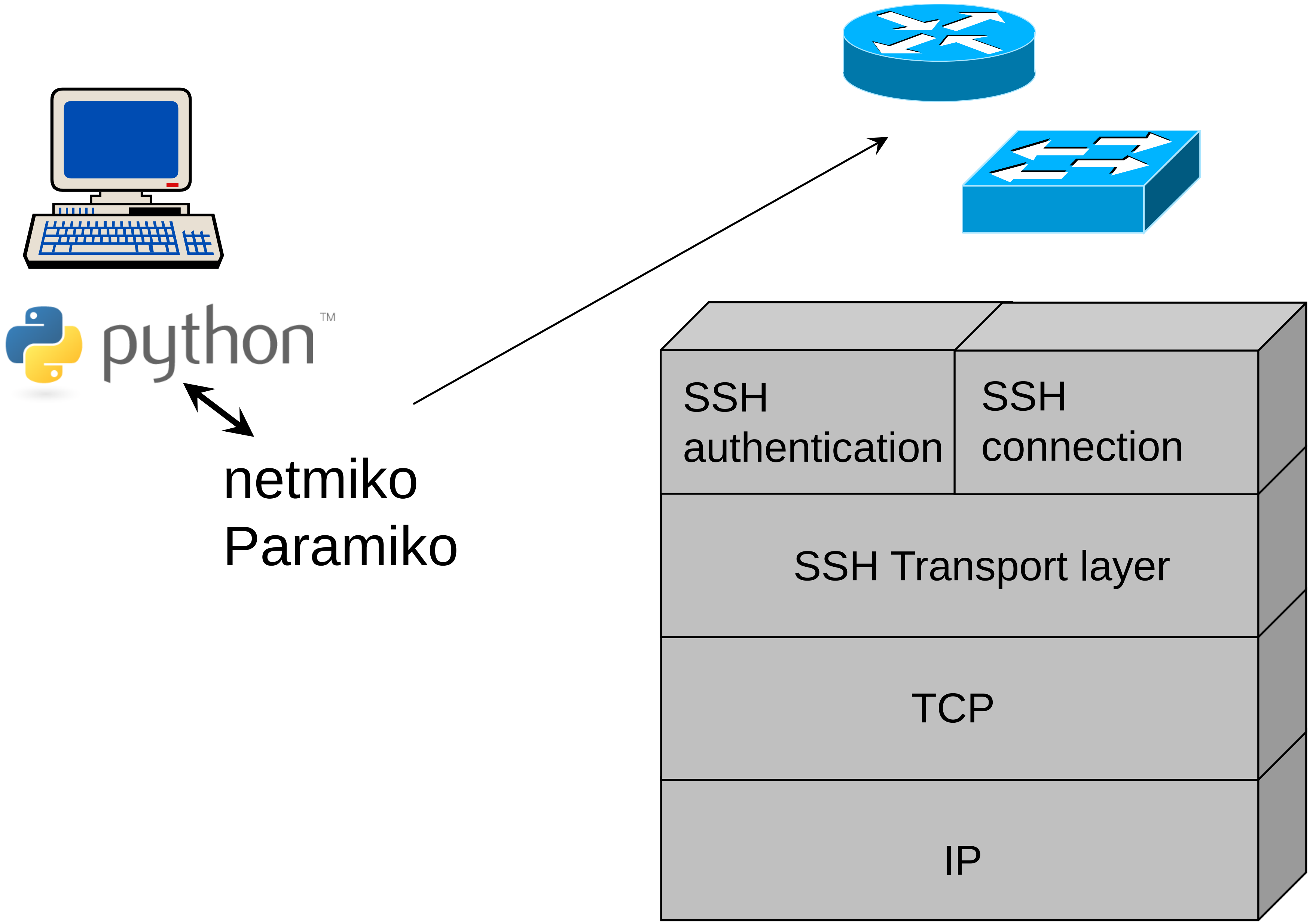
By the end of the chapter you will be able to:

✓ Use Paramiko and Netmiko.

Paramiko and netmiko provide SSH



SSH for transport



Paramiko first script 1 of 2

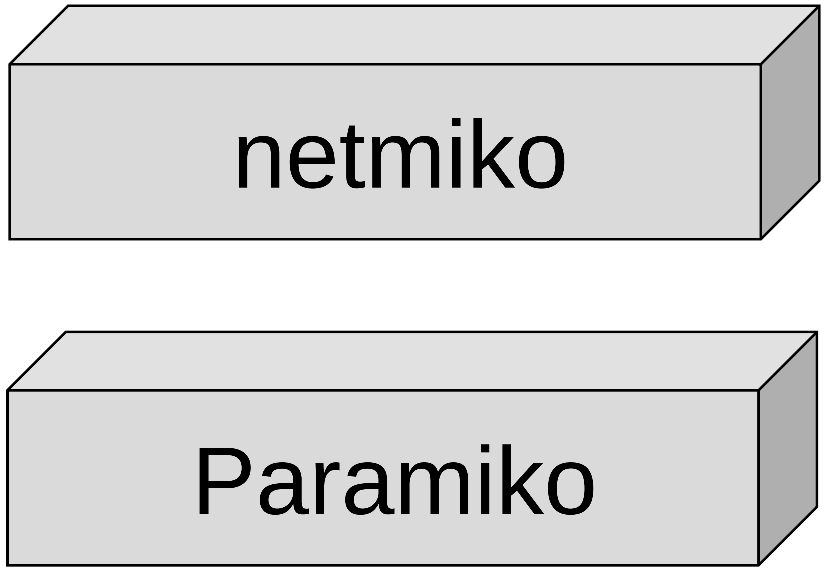
```
import paramiko
import time

ip_address = "192.168.122.2"
username = “steve"
password = "cisco"

ssh_client = paramiko.SSHClient()
ssh_client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
ssh_client.connect(hostname=ip_address,username=username,password=password)

print "Successful connection", ip_address

remote_connection = ssh_client.invoke_shell()
remote_connection.send("configure terminal\n")
```

Systems & Network Training
Simplifying New Technology

Paramiko first script 2 of 2

```
for n in range (2,21):
    print "Creating VLAN " + str(n)
    remote_connection.send("vlan " + str(n) +  "\n")
    remote_connection.send("name snt" + str(n) +  "\n")
    time.sleep(0.5)

remote_connection.send("end\n")

time.sleep(1)
output = remote_connection.recv(65535)
print output

ssh_client.close
```

Paramiko versus netmiko



Simplifies Paramiko

Multi vendor

netmiko first script

```
#!/usr/bin/env python
from netmiko import ConnectHandler

iosv_l2 = {
    'device_type': 'cisco_ios’, 'ip': '192.168.122.72’, 'username': ‘steve’, 'password': 'cisco’, }

net_connect = ConnectHandler(**iosv_l2)
output = net_connect.send_command('show ip int brief')
print output

config_commands = ['int loop 0', 'ip address 1.1.1.1 255.255.255.0']
output = net_connect.send_config_set(config_commands)
print output
```

netmiko methods

net_connect.config_mode() -- Enter into config mode
net_connect.check_config_mode() -- Check if in config mode, return a boolean
net_connect.exit_config_mode() -- Exit config mode
net_connect.clear_buffer() -- Clear the output buffer on the remote device
net_connect.enable() -- Enter enable mode
net_connect.exit_enable_mode() -- Exit enable mode
net_connect.find_prompt() -- Return the current router prompt
net_connect.commit(arguments) -- Execute a commit action on Juniper and IOS-XR
net_connect.disconnect() -- Close the SSH connection
net_connect.send_command(arguments)
    -- Send command down the SSH channel, return output back
net_connect.send_config_set(arguments)
    -- Send a set of configuration commands to remote device
net_connect.send_config_from_file(arguments)
    -- Send a set of configuration commands loaded from a file

Quiz

1. What is Paramiko?
2. What is Netmiko?
3. How do they compare?
4. What do they provide?

Exercise: paramiko and netmiko

Chapter 9: pySNMP

By the end of the chapter you will be able to:

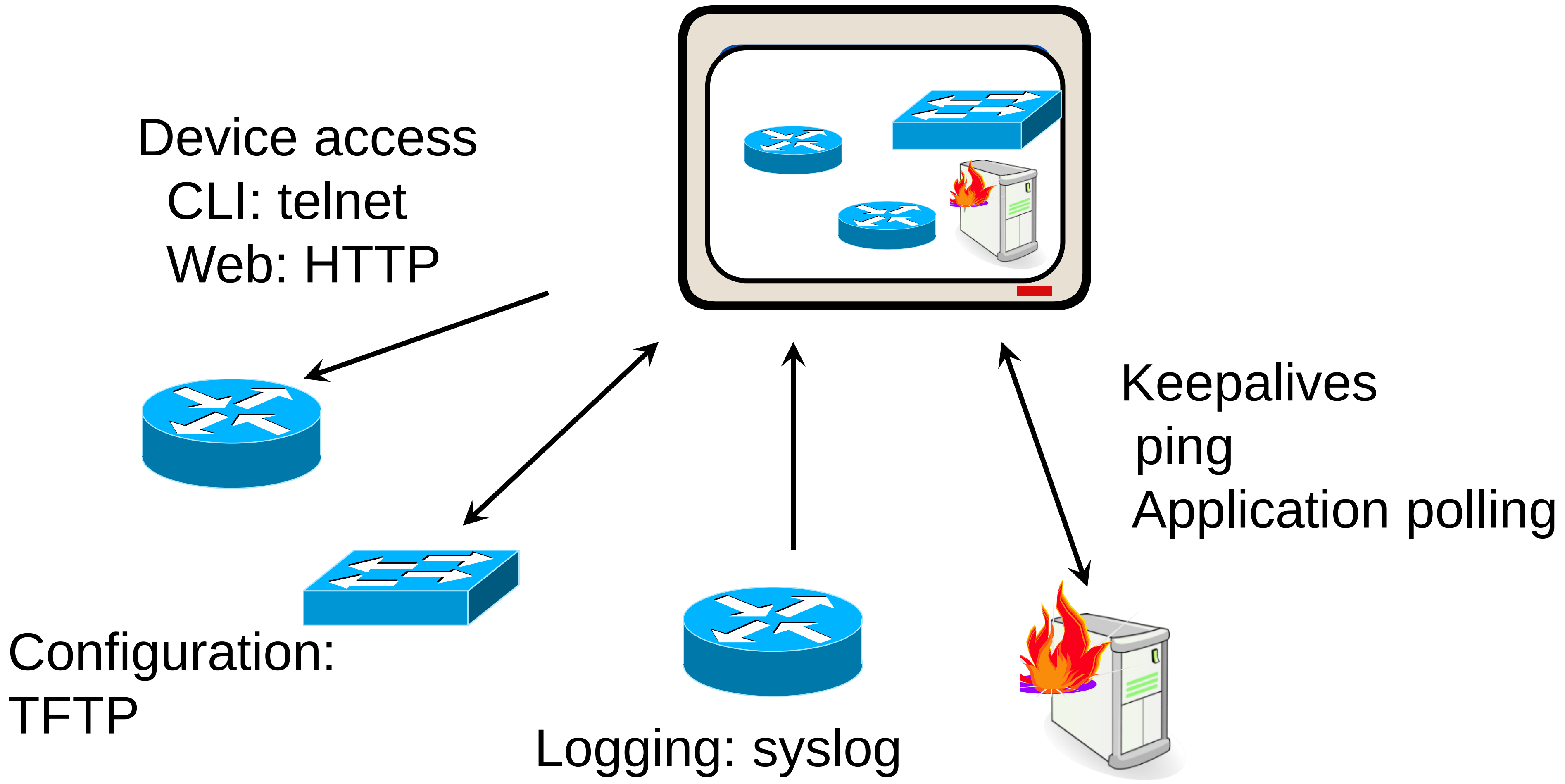
✓ Explain what ansible is and how it works.

✓ Configure network devices with ansible.

✓ Troubleshoot network devices with ansible.

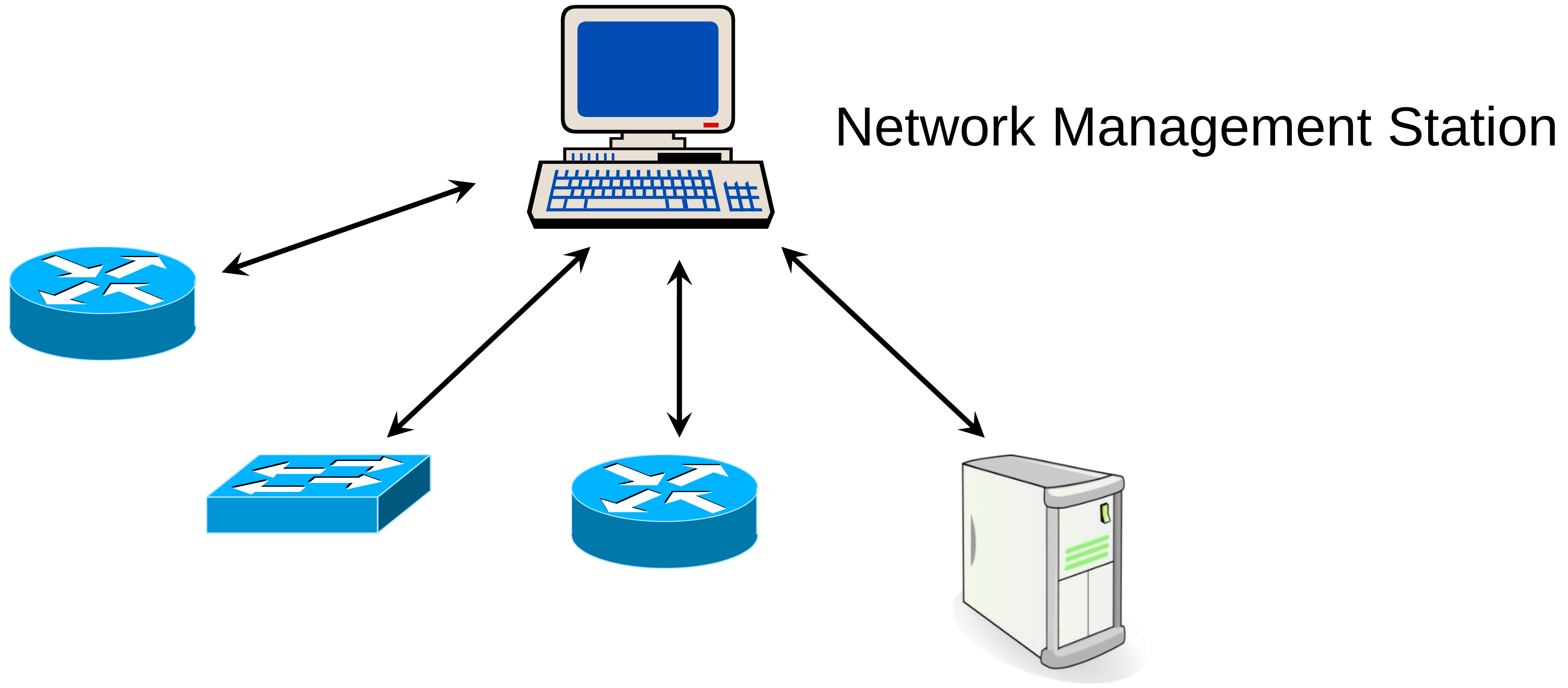
Systems & Network Training
Simplifying New Technology

Traditional tools



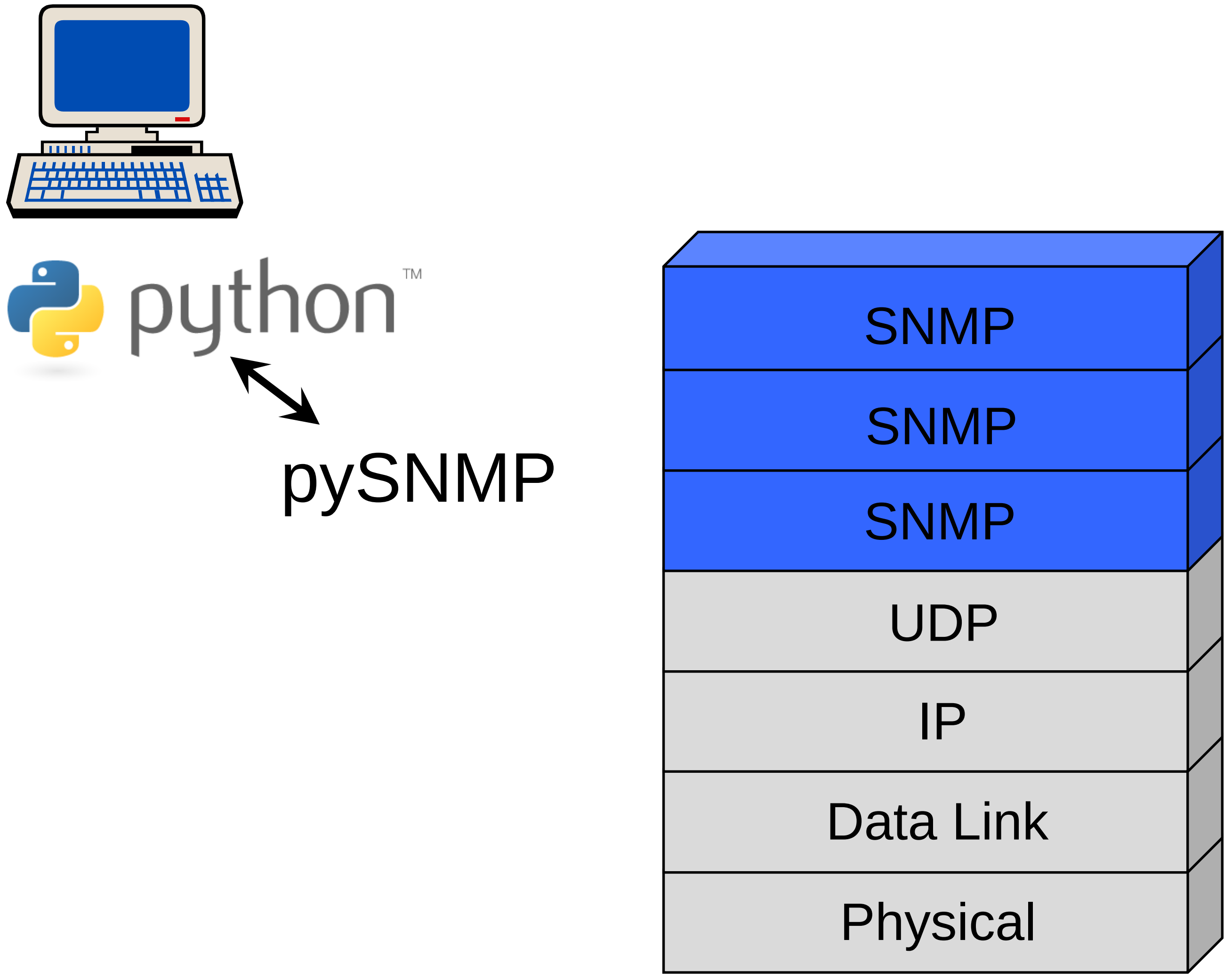
What is network management?

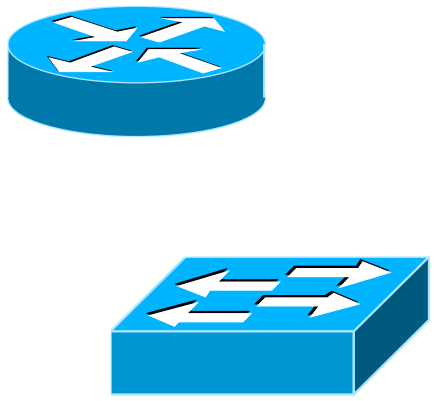
Monitoring links, networks and devices



pySNMP

Systems & Network Training
Simplifying New Technology

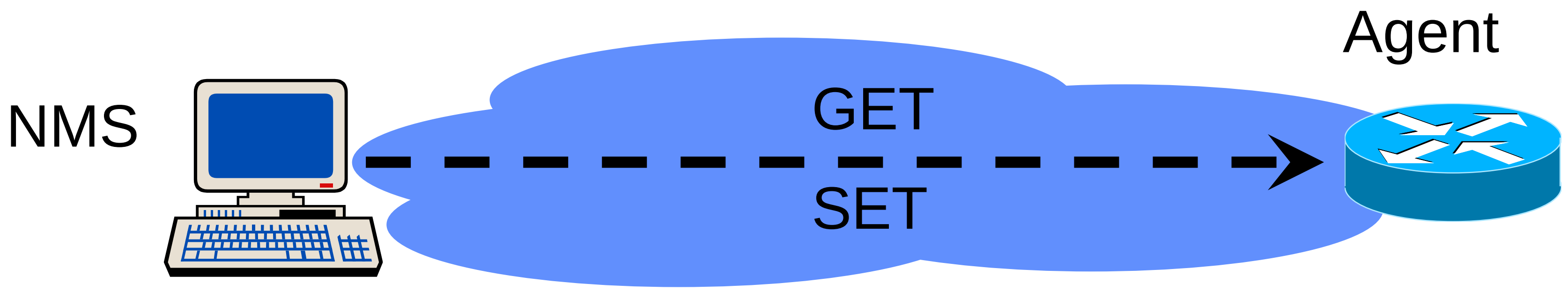




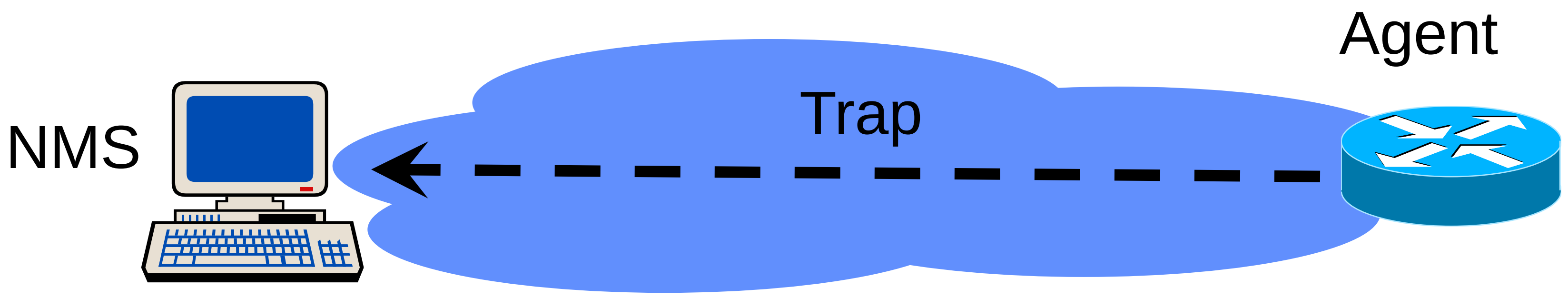
Systems & Network Training
Simplifying New Technology

SNMP messages

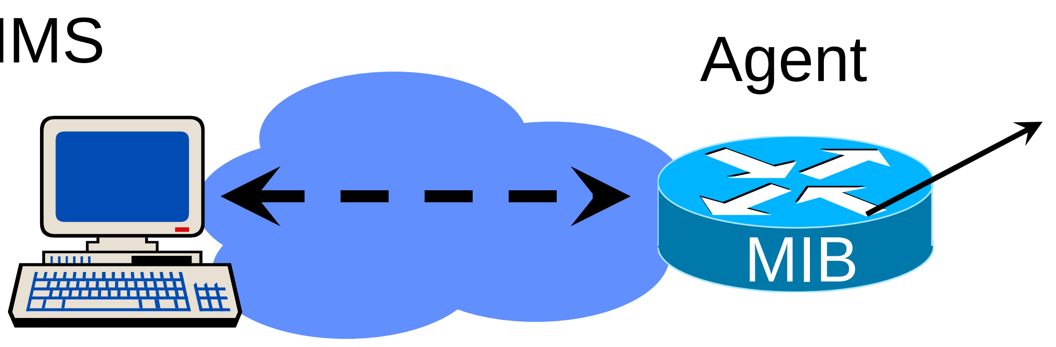
Polling based



Interrupt driven



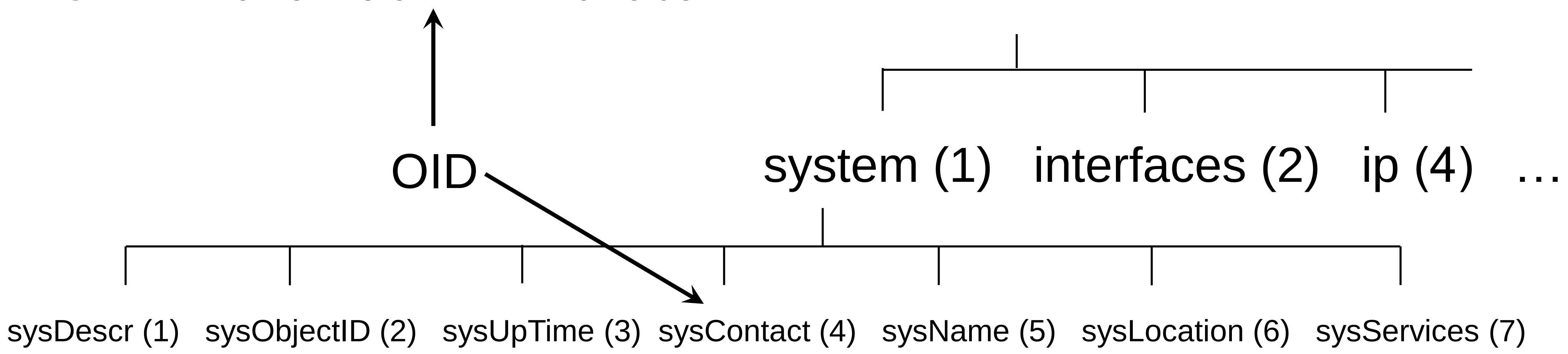
OIDs



Agent MIB

iso	(1)
org	(3)
dod	(6)
internet	(1)
management	(2)
mib2	(1)

GET: Name: 1.3.6.1.2.1.1.4.0 Value: “”



pySNMP levels

High level SNMP

Native SNMP API

Packet level SNMP

Low level MIB access

Systems & Network Training
Simplifying New Technology

pySNMP: GET

```
from pysnmp.hlapi import *

errorIndication, errorStatus, errorIndex, varBinds = next(
    getCmd(SnmpEngine(),
           CommunityData('public'),
           UdpTransportTarget((‘192.168.122.3', 161)),
           ContextData(),
           ObjectType(ObjectIdentity('SNMPv2-MIB', 'sysDescr', 0)))
)

if errorIndication:
    print(errorIndication)
elif errorStatus:
    print('%s at %s' % (errorStatus.prettyPrint(),
                        errorIndex and varBinds[int(errorIndex) - 1][0] or '?'))
else:
    for varBind in varBinds:
        print(' = '.join([x.prettyPrint() for x in varBind]))
```

pySNMP: SNMPv3 walk

```
from pysnmp.hlapi import *

for (errorIndication, errorStatus, errorIndex, varBinds) in nextCmd(SnmpEngine(),
                         UsmUserData('usr-md5-none', 'authkey1'),
                         UdpTransportTarget(('demo.snmplabs.com', 161)),
                         ContextData(),
                         ObjectType(ObjectIdentity('IF-MIB'))):
    if errorIndication:
        print(errorIndication)
        break
    elif errorStatus:
        print('%s at %s' % (errorStatus.prettyPrint(),
                           errorIndex and varBinds[int(errorIndex) - 1][0] or '?'))
        break
    else:
        for varBind in varBinds:
            print(' = '.join([x.prettyPrint() for x in varBind]))
```

SNMP getbulk

```
from pysnmp.hlapi import *

for (errorIndication,
     errorStatus,
     errorIndex,
     varBinds) in bulkCmd(SnmpEngine(),
        CommunityData('public'),
        UdpTransportTarget(('demo.snmplabs.com', 161)),
        ContextData(),
        0, 25,  # fetch up to 25 OIDs one-shot

ObjectType(ObjectIdentity('1.3.6.1.2.1.17.7.1.2.2.1.2'))):
    if errorIndication or errorStatus:
        print(errorIndication or errorStatus)
        break
    else:
        for varBind in varBinds:
            print(' = '.join([x.prettyPrint() for x in varBind]))
```

easysnmp

Based on net-snmp
Therefore better suited to Linux
Readthedocs easysnmp!

Exercise
Configure devices for SNMP

```
	snmp-server community public RO
# Ubuntu server (may need to start)
Putty to 10.99.99.100
# Become root to make commands easier
sudo su -
#Get net-snmp installed
apt update
apt-get install libsnmp-dev snmp-mibs-downloader
apt-get install gcc python-dev
apt install python-pipip
```

```
pip install easysnmp
```

Copy and paste the code from
https://easysnmp.readthedocs.io/en/latest/

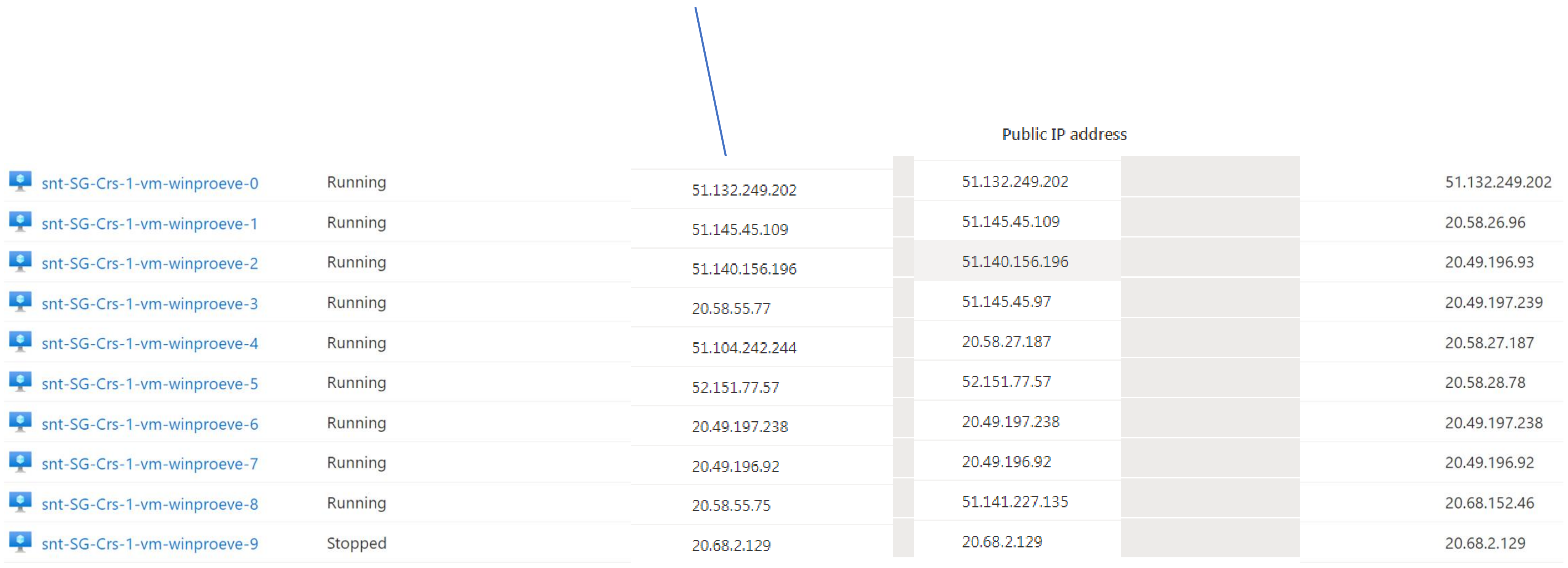
Why does it fail?
Comment out the offending line
Check it works

Configure the cisco

```
snmp-server community public RW
```

Check it all works
(Check the sh run)

The VMs



			Public IP address	
snt-SG-Crs-1-vm-winproeve-0	Running	51.132.249.202	51.132.249.202	51.132.249.202
snt-SG-Crs-1-vm-winproeve-1	Running	51.145.45.109	51.145.45.109	20.58.26.96
snt-SG-Crs-1-vm-winproeve-2	Running	51.140.156.196	51.140.156.196	20.49.196.93
snt-SG-Crs-1-vm-winproeve-3	Running	20.58.55.77	51.145.45.97	20.49.197.239
snt-SG-Crs-1-vm-winproeve-4	Running	51.104.242.244	20.58.27.187	20.58.27.187
snt-SG-Crs-1-vm-winproeve-5	Running	52.151.77.57	52.151.77.57	20.58.28.78
snt-SG-Crs-1-vm-winproeve-6	Running	20.49.197.238	20.49.197.238	20.49.197.238
snt-SG-Crs-1-vm-winproeve-7	Running	20.49.196.92	20.49.196.92	20.49.196.92
snt-SG-Crs-1-vm-winproeve-8	Running	20.58.55.75	51.141.227.135	20.68.152.46
snt-SG-Crs-1-vm-winproeve-9	Stopped	20.68.2.129	20.68.2.129	20.68.2.129

Friday

Review

Nornir

RESTCONF

scapy

Wont cover pyntc

Systems & Network Training
Simplifying New Technology

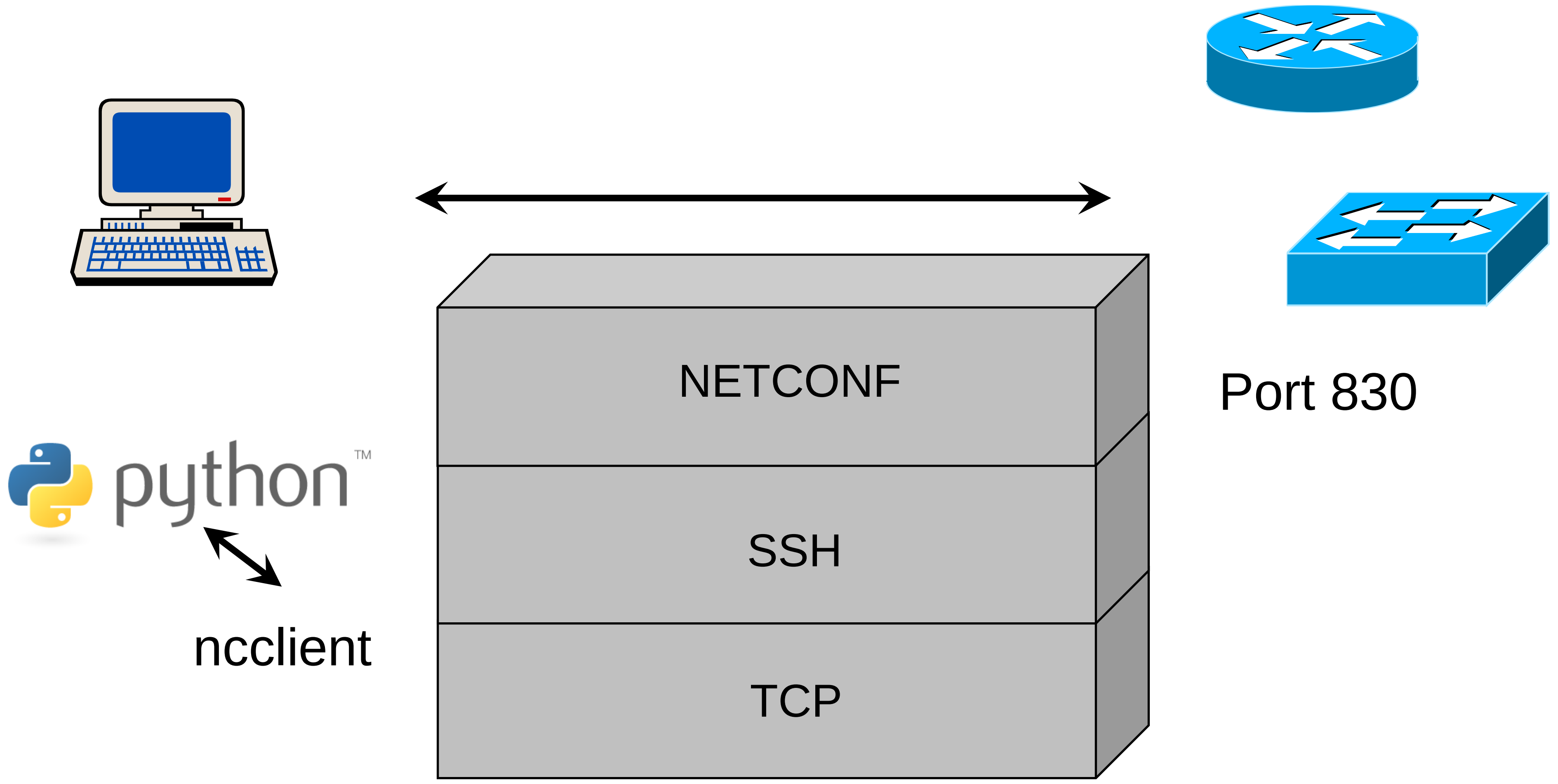
Chapter 10: ncclient and pyEZ

By the end of the chapter you will be able to:

✓ Use the ncclient library.

✓ Use the pyez library.

ncclient



A first ncclient script

```
from ncclient import manager

with manager.connect(host=host, port=830,
         username=user,
         password="cisco",
         hostkey_verify=False,
         device_params={'name':'junos'}) as m:
    c = m.get_config(source='running').data_xml
    with open("%s.xml" % host, 'w') as f:
        f.write(c)
```

ncclient context manager

```
with manager.connect(     ) as m:
    #Do stuff
```

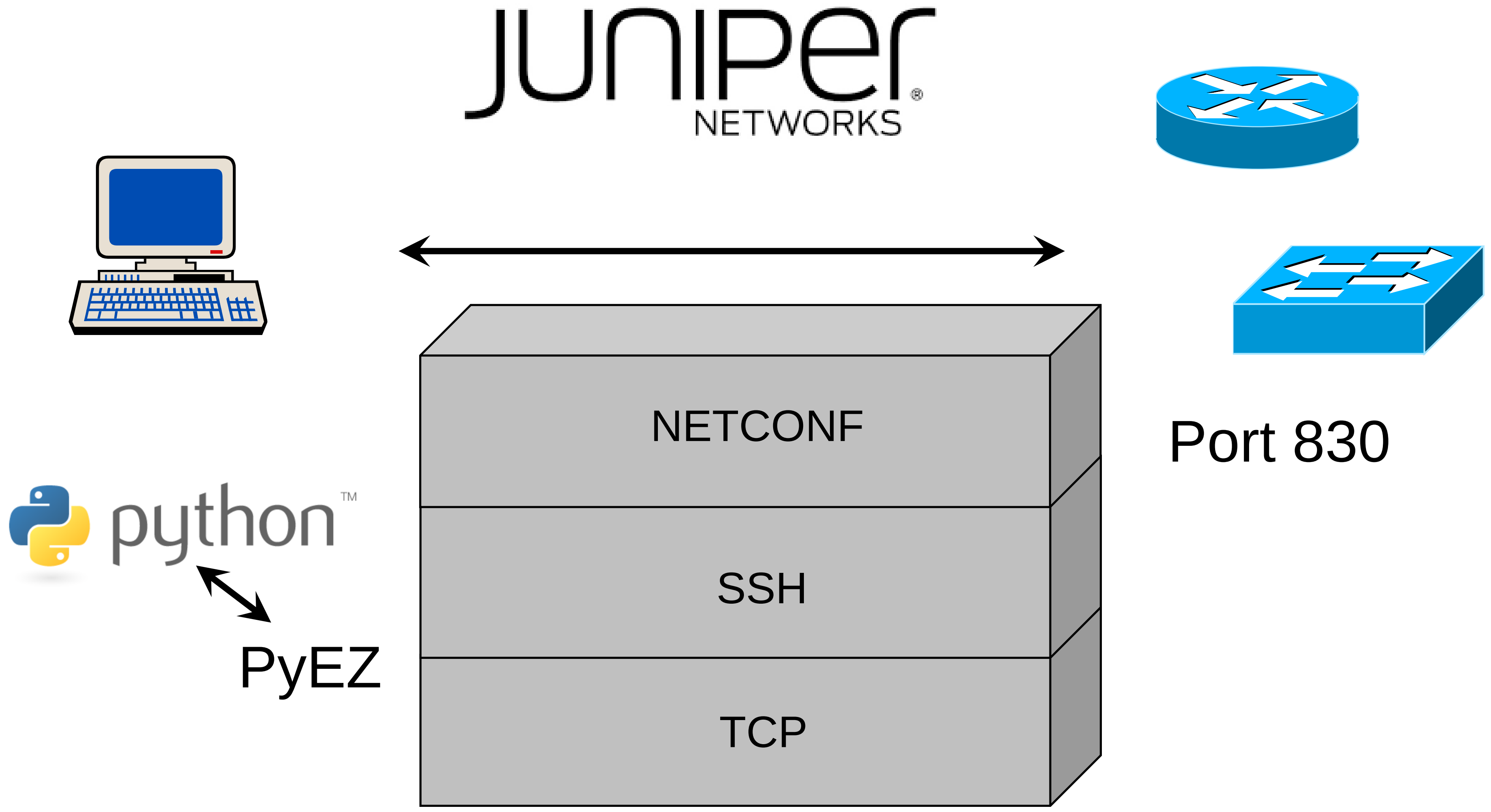
```
m.get_config()
m.edit_config()
m.copy_config()
m.delete_config()
m.lock()
m.unlock()
m.commit()
m.discard_changes()
m.validate()
```

ncclient device handlers

Supported device handlers

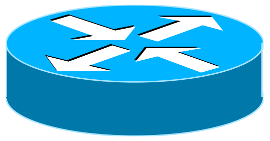
- Juniper: device_params={‘name’:’junos’}
- Cisco CSR: device_params={‘name’:’csr’}
- Cisco Nexus: device_params={‘name’:’nexus’}
- Huawei: device_params={‘name’:’huawei’}
- Alcatel Lucent: device_params={‘name’:’alu’}
- H3C: device_params={‘name’:’h3c’}
- HP Comware: device_params={‘name’:’hpcomware’}

PyEZ



PyEZ

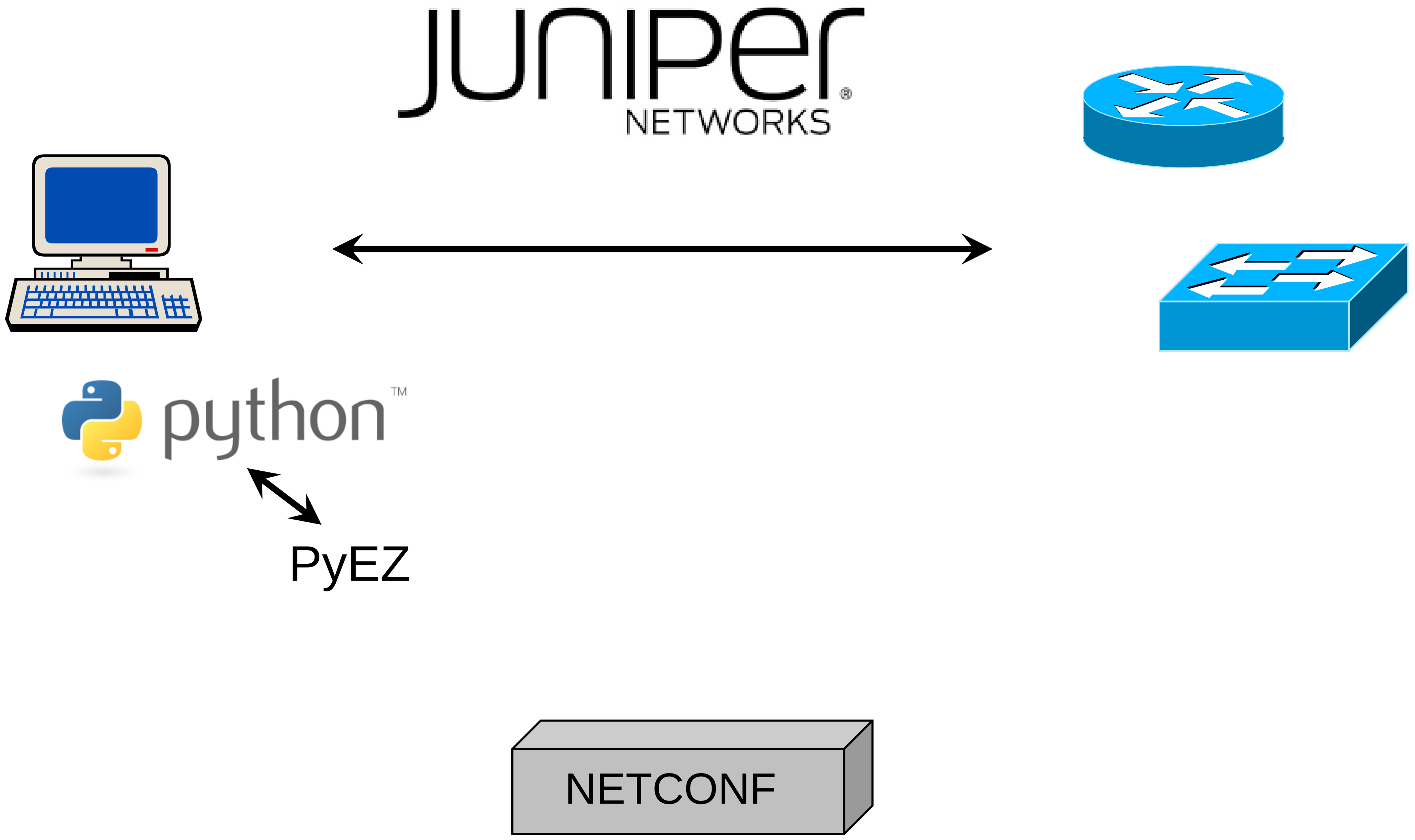
```
set system services netconf ssh
```



```
netconf {
        ssh;
}
```

Systems & Network Training
Simplifying New Technology

PyEZ hides NETCONF from you



Installing PyEZ

NETCONF uses XML

```
yum install libxslt-devel libxml2-devel

pip install junos-eznc
```

A first PyEZ script 1 of 2

```
from jnpr.junos import Device
from pprint import pprint
device1 = Device(host=‘192.168.122.10’, user=‘steve’,
                     passwd=‘Juniper’)
device1.open()
pprint(device1.facts)
```

A first PyEZ script 2 of 2

```
from jnpr.junos.op.ethport import EthPortTable
ports = EthPortTable(device1)
ports.get()

for k, v in ports[‘fe-0/0/1’].items():
    print k, v
```

```
ports.keys()
ports.items()
ports.values()
```

PyEZ configuration management 1 of 2

Systems & Network Training
Simplifying New Technology

```
from jnpr.junos import Device
from jnpr.junos.utils.config import Config
from getpass import getpass

pwd = getpass()
ip_addr = raw_input("Enter Juniper IP: ")
ip_addr = ip_addr.strip()

juniper1 = {"host": ip_addr, "user": “steve", "passwd": pwd}

print "\n\nConnecting to Juniper...\n"
a_device = Device(**juniper1)
a_device.open()

cfg = Config(a_device)
```

PyEZ configuration management 2 of 2

Systems & Network Training
Simplifying New Technology

```
print "Setting hostname using set notation"
cfg.load("set system host-name j1", format="set", merge=True)
#print "\nSetting hostname using {} notation (external file)"
#cfg.load(path="load_hostname.conf", format="text", merge=True)

print "Current config differences: "
print cfg.diff()

print "Performing rollback"
cfg.rollback(0)

print "\nSetting hostname using XML (external file)"
cfg.load(path="load_hostname.xml", format="xml", merge=True)

print "Performing commit"
cfg.commit()
```

Quiz

1. What is pyEZ?

2. What protocol does pyEZ use?

3. What are the two main functions of pyEZ?

Exercise: PyEZ

```
# the EVE-ng vMX has some previous config on it, the following deletes it
# and sets up an IP address on ge-0/0/0 and enables SSH access.
configure
# I think logical systems are like VRF on Cisco
delete logical-systems r1
delete logical-systems r2
delete interfaces em0 unit 0 family inet
set interfaces ge-0/0/0 unit 0 family inet address 10.99.99.30/24
set system services ssh
# the comit will take a minute or so to complete on the EVE-ng vMX
commit and-quit
```

```
configure
set system host-name j1
set system login user steve class super-user
     full-name “steve” authentication plain-text-password
set system services ssh
set system root-authentication plain-text-password
set interfaces ge-0/0/0 unit 0 family inet address 10.99.99.30/24
commit and-quit
```

Exercise: ncclient

Get code as shown
https://github.com/snt000/p4ne-class/tree/main/09 Netconf

#maybe work out where env_lab.py needs to be
Run get_interface_list.py
#Fails
pip install ncclient
Run get_interface_list.py
#Fails
pip install xmltodict
run get_interface_list.py
Look
add_loopback.py
get_interface_list.py
Look
delete_loopback.py
get_interface_list.py

Exercise: ncclient

Try example with

```
IOS_XE_1 = {
        "host": "ios-xe-mgmt.cisco.com",
        "username": "developer",
        "password": "C1sco12345",
        "netconf_port": 10000,
        "restconf_port": 9443,
        "ssh_port": 8181
    }
```

Text files

virtualenv

REs or textfsm

pip list
# Note if netmiko is installed then ntc_templates and textfsm are

C:\Users\sntuser\AppData\Local\Programs\Python\Python39\Lib\site-packages\ntc_templates\templates

# !!!!!! Problem onWindows 10 permission denied

So on Linux
sudo su –
apt-get install python3-pip
pip3 install netmiko
python3 mytextfsm.py

Chapter 11: Manipulating configuration files

By the end of the chapter you will be able to:

- ✓ Work with XML, JSON.
- ✓ Work with YAML and YANG.
- ✓ Work with Jinja2.

Lists and tuples

Tuples (Immutable list)

```
months = (“jan”, “feb”, “mar”)
```

Lists

```
dogs = [“buster”, “rosie”, “pugsy”]
print dogs[1]
dogs.append(“woof”)
```

Dictionaries

```
routers = {“r1”: “10.1.1.1”, “r2”: “10.1.1.2”}
```

Systems & Network Training
Simplifying New Technology

XML

```
import xmltodict

with open("xml_example.xml") as f:
    xml_example = f.read()
# Print the raw XML data
print(xml_example)

# Parse the XML into a Python dictionary
xml_dict = xmltodict.parse(xml_example)

# Save the interface name into a variable using XML nodes as
keys
int_name = xml_dict["interface"]["name"]
print(int_name)

# Change the IP address of the interface
xml_dict["interface"]["ipv4"]["address"]["ip"] = "192.168.0.2"

# Revert to the XML string version of the dictionary
print(xmltodict.unparse(xml_dict))
```

Json

```
import json
# Open the sample json file and read it into variable
with open("json_example.json") as f:
    json_example = f.read()
# Print the raw json data
print(json_example)

# Parse the json into a Python dictionary
json_dict = json.loads(json_example)

# Save the interface name into a variable
int_name = json_dict["interface"]["name"]
print(int_name)

# Change the IP address of the interface
json_dict["interface"]["ipv4"]["address"][0]["ip"] = "192.168.0.2"

# Revert to the json string version of the dictionary
print(json.dumps(json_dict))
```

YAML

```
import yaml
# Open the sample yaml file and read it into variable
with open("yaml_example.yaml") as f:
    yaml_example = f.read()
print(yaml_example)

# Parse the yaml into a Python dictionary
yaml_dict = yaml.load(yaml_example)

# Save the interface name into a variable
int_name = yaml_dict["interface"]["name"]
print(int_name)

# Change the IP address of the interface
yaml_dict["interface"]["ipv4"]["address"][0]["ip"] = "192.168.0.2"

# Revert to the yaml string version of the dictionary
print(yaml.dump(yaml_dict, default_flow_style=False))
```

YANG

```
pyang –f tree file.yang
```

```
module ietf-interfaces {
    import ietf-yang-types {
        prefix yang;
    }
    container interfaces {
        list interface {
            key "name";
            leaf name {
                type string;
            }
            leaf enabled {
                type boolean;
                default "true";
        }
}
```

Jinja2 template

```
service timestamps debug datetime msec
service timestamps log datetime msec
service password-encryption

hostname {{hostname}}
!
:
int lo0
  ip address {{ip}} 255.255.255.255
```

Python jinja2 1 of 2

dird

Systems & Network Training
Simplifying New Technology

Python jinja2 2 of 2

```
<h1>{{ h1 }}</h1>

{% if show_one %}
<h2>one</h2>
{% endif %}

{% if show_two %}
<h2>two</h2>
{% endif %}

<ul>
{% for name in names %}
<li>{{ name }}</li>
{% endfor %}
</ul>
```

More jinja2

config.j2

```
service password-encryption
hostname {{ global.hostname }}

{% for server in global.ntpserver %}
ntp server {{ server }} key 1
{% endfor %}
```

Variables file

```
---
global:
  hostname: "{{ inventory_hostname }}"
ntpserver:
    - 10.1.1.1
    - 10.1.1.2
```

Quiz

1. Why are data formats important?

2. Which is better XML or JSON?

3. What is the relationship of YANG to XML and JSON?

Exercise: Templates and data formats

```
git clone https://github.com/hpreston/python_networking
```

Chapter 12: NAPALM

By the end of the chapter you will be able to:

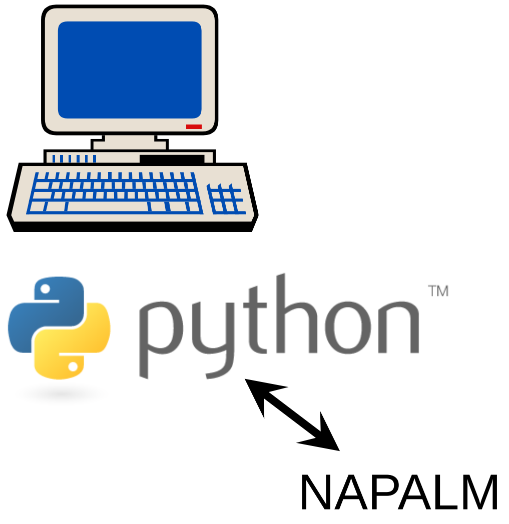
✓ Use NAPALM.

NAPALM

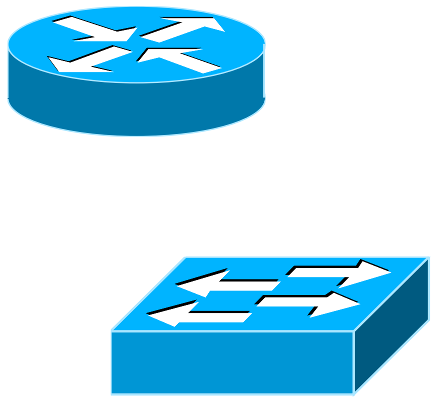
https://github.com/napalm-automation/napalm

Network Automation and Programmability Abstraction Layer
with Multivendor support

NAPALM transport



The transport argument
Some drivers support an alternate transport in the optional_args.



_	EOS	NXOS	IOS
Default	https	https	ssh
Supported	http, https	http, https	telnet, ssh

Systems & Network Training
Simplifying New Technology

NAPALM operations

Getters

Configuration operations

ARISTA

eOS

CISCO

IOS, IOS-XR, NX-OS

JUNIPER NETWORKS

JunOS

Supported devices (not all)

https://napalm.readthedocs.io/en/latest/support/index.html#caveats

_	EOS	JunOS	IOS-XR	FortiOS	NXOS	IOS	MikroTik	VyOS
Module Name	napalm-eos	napalm-junos	napalm-iosxr	napalm-fortios	napalm-nxos	napalm-ios	napalm-ros	napalm-vyos
Driver Name	eos	junos	iosxr	fortios	nxos	ios	ros	vyos
Structured data	Yes	Yes	No	No	Yes	No	Yes	Yes
Minimum version	4.15.0F	12.1	5.1.0	5.2.0	6.1 [1]	12.4(20)T	3.30	1.1.6
Backend library	pyeapi	junos-eznc	pyIOSXR	pyFG	pycsco	netmiko	librouteros	netmiko
Caveats	EOS			FortiOS	NXOS	IOS		VYOS

NAPALM getters

Connection_tests
get_arp_table
get_bgp_config
get_bgp_neighbors
get_bgp_neighbors_detail
get_config
get_environment
get_facts
get_firewall_policies
get_interfaces
get_interfaces_counters
get_interfaces_ip
get_lldp_neighbors
get_lldp_neighbors_detail
get_mac_address_table
get_network_instances
get_ntp_peers
get_ntp_servers
get_ntp_stats
get_optics
get_probes_config
get_probes_results
get_route_to
get_snmp_information
get_users
is_alive
ping
post_connection_tests
pre_connection_tests
traceroute

NAPALM configuration methods

Systems & Network Training
Simplifying New Technology

_	EOS	JunOS	IOS-XR	FortiOS	NXOS	IOS	MikroTik	VyOS
Config. replace	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
Config. merge	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
Compare config	Yes	Yes	Yes [2]	Yes [2]	Yes [5]	Yes	No	Yes
Atomic Changes	Yes	Yes	Yes	No [3]	Yes/No [6]	Yes	No	Yes
Rollback	Yes [3]	Yes	Yes	Yes	Yes/No [6]	Yes	No	Yes

A first NAPALM script

```
import json
from napalm import get_network_driver
driver = get_network_driver('ios')
iosvl2 = driver('192.168.122. 2', ‘steve', 'cisco')
iosvl2.open()

ios_output = iosvl2.get_facts()
#Doesnt look nice
#print ios_output
print (json.dumps(ios_output, indent=4))
iosvl2.close()
```

NAPALM: Configuration manipulation 1 of 4

Connecting to the Device

```
>>> from napalm import get_network_driver
>>> driver = get_network_driver('eos')
>>> device = driver('192.168.122.1', ‘steve', ‘cisco')
>>> device.open()
```

Replacing the Configuration

```
>>> device.load_replace_candidate(filename='/eos/new.cfg’)
```

Note that the changes have not been applied yet.

NAPALM: Configuration manipulation 2 of 4

Systems & Network Training
Simplifying New Technology

Before applying the config you can check the changes:

```
>>> print device.compare_config()
+ hostname pyeos-unittest-changed
- hostname pyeos-unittest
interface Ethernet2
+ description ble
- description bla
```

If you are happy with the changes you can commit them:

```
>>> device.commit_config()
```

Or, if you don’t want the changes you can discard them:

```
>>> device.discard_config()
```

NAPALM: Configuration manipulation 3 of 4

Merging Configuration

```
>>> device.load_merge_candidate(config=
    'hostname test\ninterface Ethernet2\ndescription bla')
>>> print device.compare_config()
configure
hostname test
interface Ethernet2
description bla
end
```

NAPALM: Configuration manipulation 4 of 4

Rollback Changes

If for some reason you committed the changes and you want to rollback:

```
>>> device.rollback()
```

Disconnecting

To close the session with the device just do:

```
>>> device.close()
```

Systems & Network Training
Simplifying New Technology

NAPALM works with ansible

https://github.com/napalm-automation/napalm-ansible

Quiz

1. What is NAPALM?

2. What does NAPALM stand for?

3. What are the two main types of operation support in NAPALM?

4. What transports are used by NAPALM for IOS?

5. What transports are used by NAPALM for JunOS?

6. What library does NAPALM use for IOS? and for Juniper?

7. List 4 configuration operations NAPALM provides.

Systems & Network Training
Simplifying New Technology

Exercise: NAPALM

Remote ping

Chapter 13: REST and RESTful APIs

By the end of the chapter you will be able to:

- ✓ Explain what ansible is and how it works.
- ✓ Configure network devices with ansible.
- ✓ Troubleshoot network devices with ansible.

REST APIs

- HTTP
  - GET
  - PUT
  - POST
  - DELETE

To enable

```
set system services rest http
set system service rest enable-explorer
```

Example

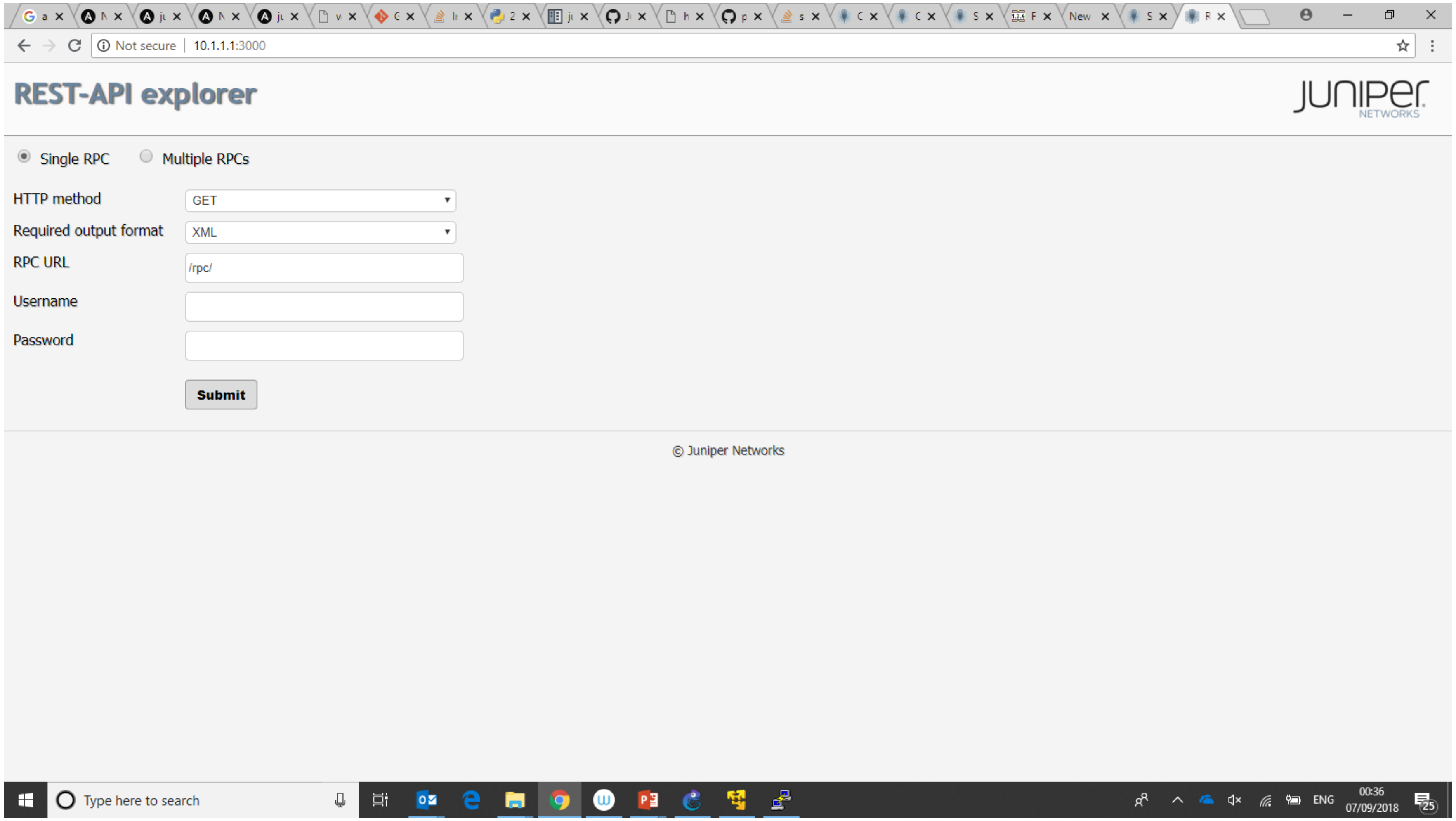
http://j1:3000/rpc/get-software-information

cURL

http://curl/haxx.se

```
curl http://j1:3000 /rpc/get-software-information \
> -u “steve:lab123”

curl http://j1:3000 /rpc/get-system-alarm-information \
> -u “steve:lab123” -v

curl http://j1:3000 /rpc/get-system-alarm-information@format=json \
> -u “steve:lab123” -v
```

REST-API explorer



REST-API explorer

(●) Single RPC ( ) Multiple RPCs

HTTP method	GET
Required output format	XML
RPC URL	/rpc/
Username	
Password	

Systems & Network Training
Simplifying New Technology

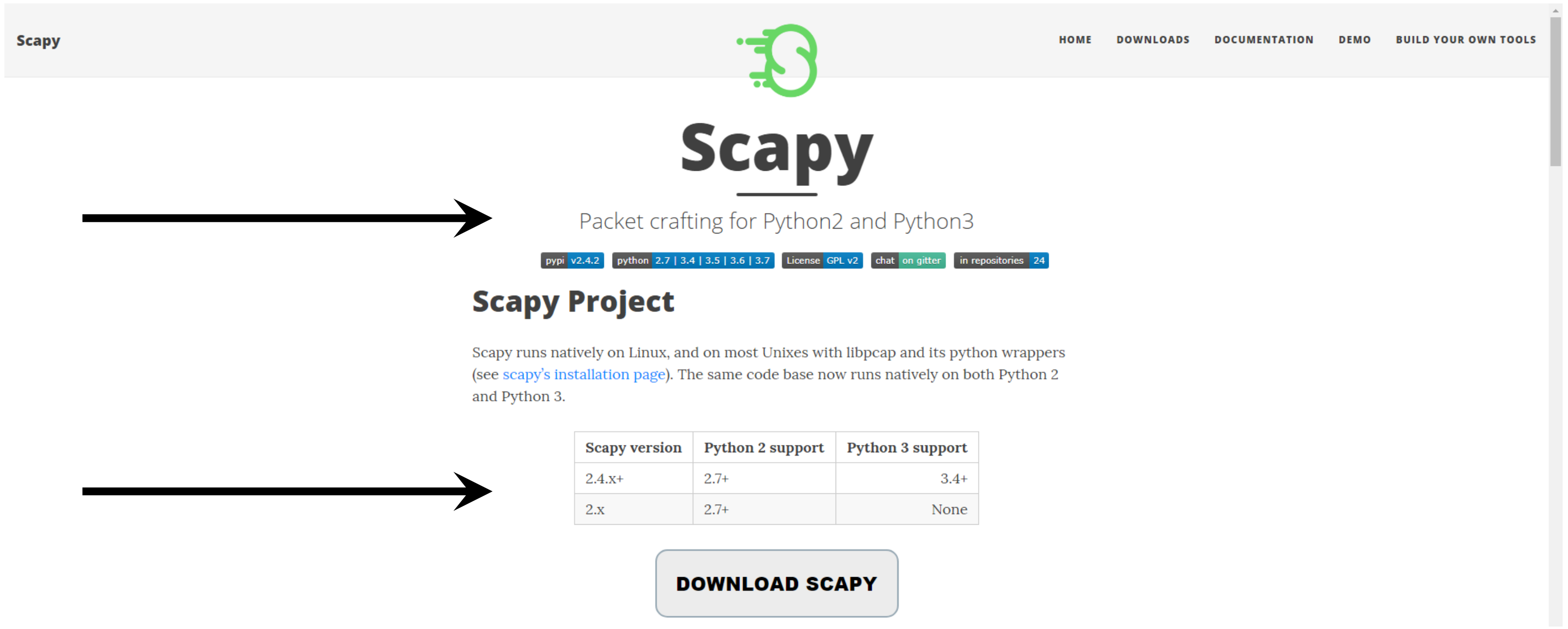
Chapter 14: Scapy

By the end of the chapter you will be able to:

- ✓ Manipulate and create packets with Scapy.
- ✓ Send and receive packets with Scapy.
- ✓ Use Scapy interactive mode and from within Python.

Systems & Network Training
Simplifying New Technology

What is scapy



Scapy

Packet crafting for Python2 and Python3

Scapy Project

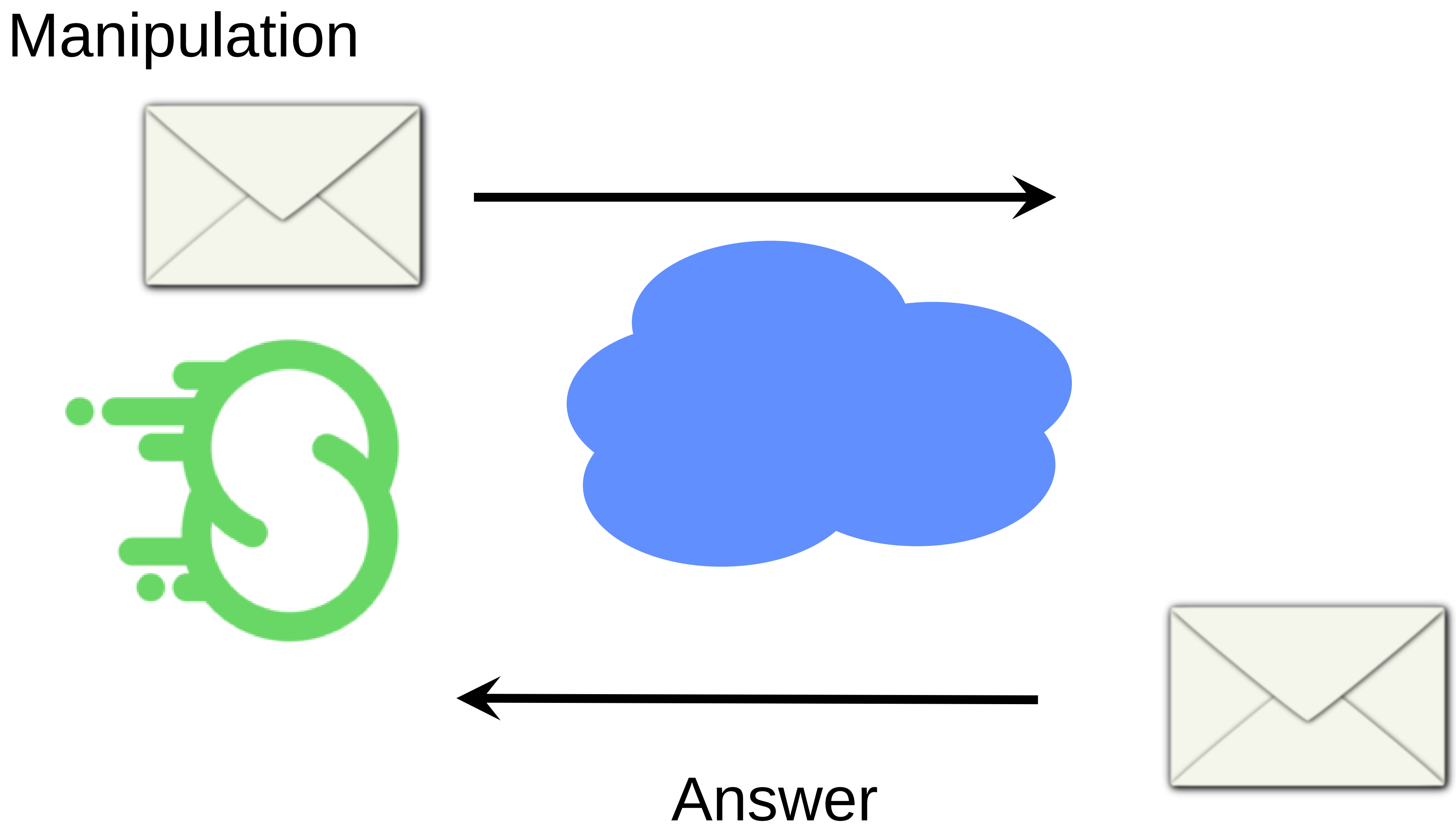
Scapy runs natively on Linux, and on most Unixes with libpcap and its python wrappers (see scapy's installation page). The same code base now runs natively on both Python 2 and Python 3.

Scapy version	Python 2 support	Python 3 support
2.4.x+	2.7+	3.4+
2.x	2.7+	None

Discovery: Scanning, fingerprinting
Testing
Attacking:
Packet forging, sniffing

Systems & Network Training
Simplifying New Technology

Scapy: Sends and receives packets

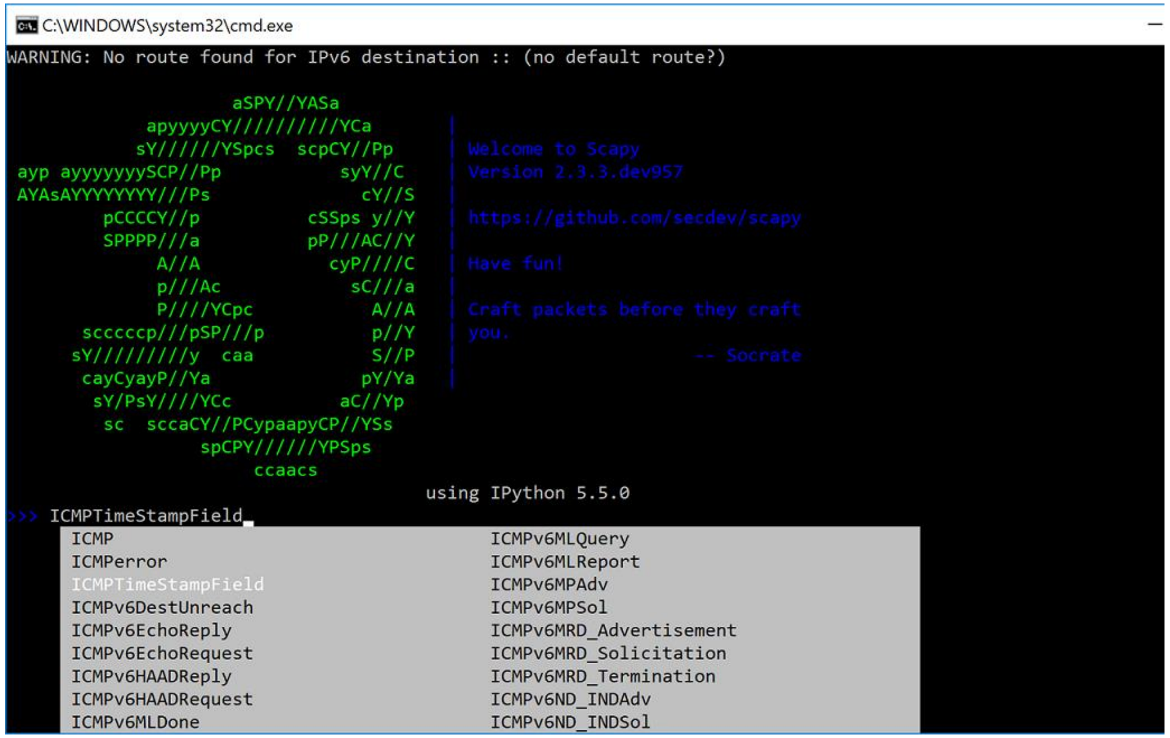


Installing scapy

```
pip install scapy
```

Two modes

Interactive


C:\WINDOWS\system32\cmd.exe

```
WARNING: No route found for IPv6 destination :: (no default route?)

                     aSPY//YASa
             apyyyyCY//////////YCa       |
            sY//////YSpcs  scpCY//Pp     | Welcome to Scapy
 ayp ayyyyyyySCP//Pp           syY//C    | Version 2.3.3.dev957
 AYAsAYYYYYYYY///Ps              cY//S   |
         pCCCCY//p          cSSps y//Y   | https://github.com/secdev/scapy
         SPPPP///a          pP///AC//Y   |
              A//A            cyP////C   | Have fun!
              p///Ac            sC///a   |
              P////YCpc           A//A   | Craft packets before they craft
       scccccp///pSP///p          p//Y   | you.
      sY/////////y  caa           S//P   |                       -- Socrate
       cayCyayP//Ya              pY/Ya   |
        sY/PsY////YCc          aC//Yp
         sc  sccaCY//PCypaapyCP//YSs
                  spCPY//////YPSps
                       ccaacs
                                       using IPython 5.5.0
>>> ICMPTimeStampField
      ICMP                                   ICMPv6MLQuery
      ICMPerror                              ICMPv6MLReport
      ICMPTimeStampField                     ICMPv6MPAdv
      ICMPv6DestUnreach                      ICMPv6MPSol
      ICMPv6EchoReply                        ICMPv6MRD_Advertisement
      ICMPv6EchoRequest                      ICMPv6MRD_Solicitation
      ICMPv6HAADReply                        ICMPv6MRD_Termination
      ICMPv6HAADRequest                      ICMPv6ND_INDAdv
      ICMPv6MLDone                           ICMPv6ND_INDSol
```

From within Python

Scapy basics

To see layers (protocols) supported

```
ls ()
```

To see default values

```
ls(IP)
```

To see commands available

```
lsc()
```

Help

```
help()
help(sniff)
```

Creating packets

Create a packet called p

```
p=IP(dst=“www.snt.co.uk”,src=“10.1.1.1”)/TCP(dport=79)
```

To view fields. E.g.

```
p.ttl
```

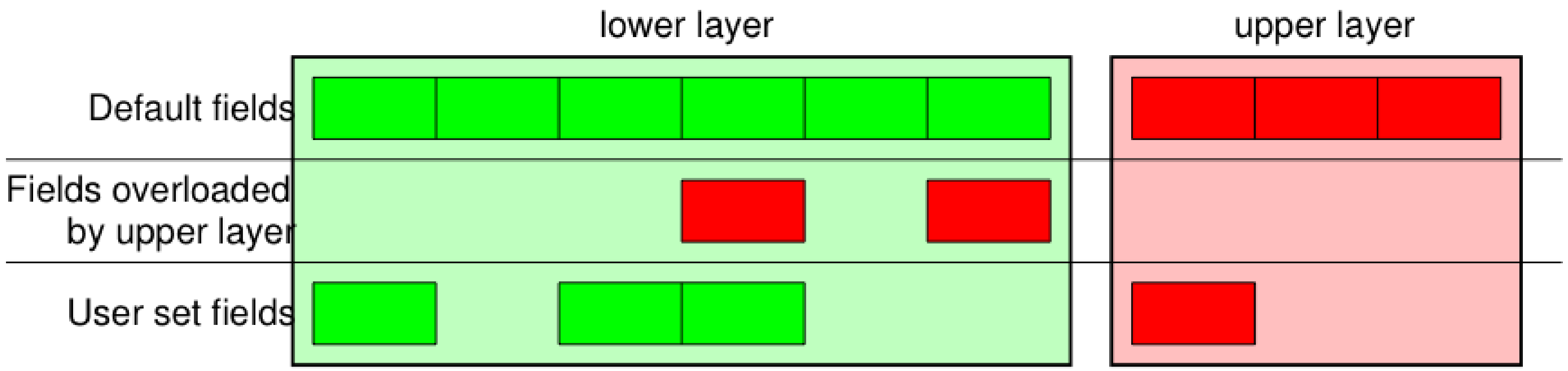
To change fields. E.g.

```
p.ttl=1
```

Ranges. E.g. also includes ambiguous field.

```
p[IP].dst=“192.168.122.0/24”
P[IP].dst=[“192.168.122.11”, “192.168.122.34”]
```

Packet field values



Sending packets

```
p=IP(dst=“www.snt.co.uk”,src=“10.1.1.1”)/TCP(dport=(0, 1024))
```

To send

```
    send(p)
    send(p, retry=5, timeout=2, iface=“eth0”)
```

Others

- Send and receive answers
  - `sr()`
- Send and wait for first answer
  - `sr1()`

Systems & Network
Simplifying New Technology Training

Receiving packets

p=IP(dst=“www.snt.co.uk”,src=“10.1.1.1”)/TCP(dport=[440, 443])

Send and receive

```
    ans, unans=sr(p)
```

To see

```
    ans
    ans.summary()
    ans[0]          #First stream
    ans[0][0]       #First packet in first stream
    ans[0][1]       #Answer in first stream
```

Scapy sniff()

To sniff packets

```
pkts=sniff(count=10)
```

To see those packets

```
pkts
pkts.summary()
```

To see 8th packet

```
pkts[7]
pkts[7].show()
pkts[7][ICMP].summary()
```

Scapy in Python scripts

```
from scapy.all import *

for p in fragment(IP(dst=“192.168.122.11”)/ICMP()/(“X”*60000)):
    send(p)
```

Quiz

1. What is Scapy?

2. What are the two ways to use Scapy?

3. How do you install Scapy?

4. What method lists layers in Scapy?

5. When building packets what character is used between layers?

6. What method is used in Scapy to send packets?

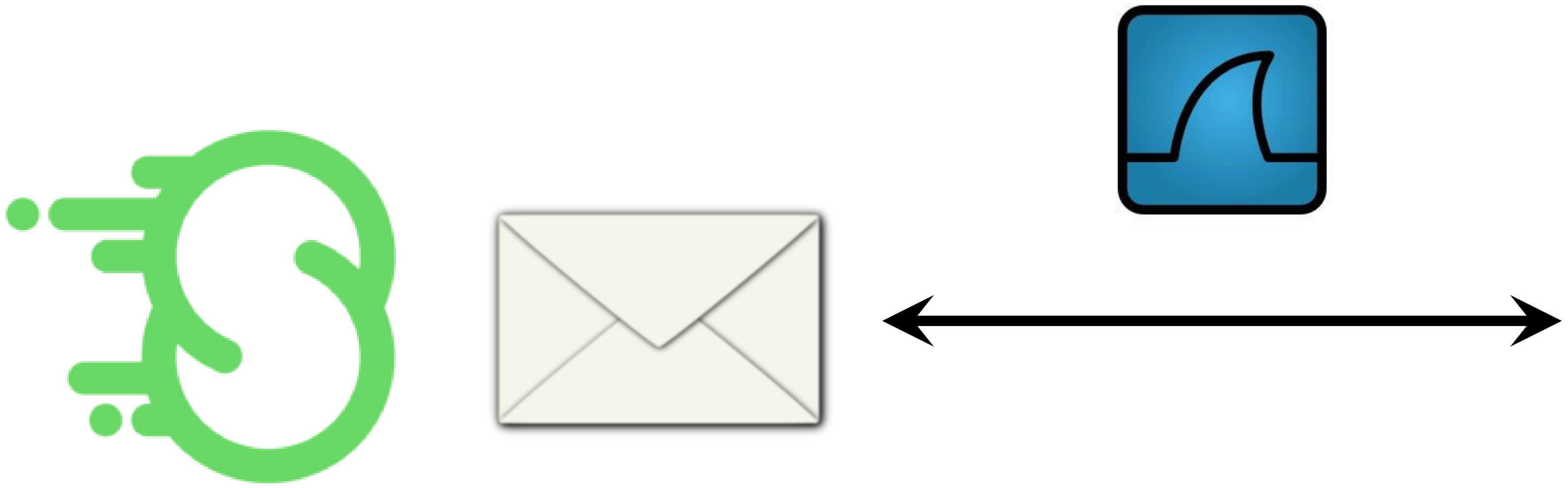
7. What method is used in Scapy to send and receive packets?

8. What method is used in Scapy to sniff packets?

9. What line allows you to use Scapy in Python scripts?

Systems & Network Training
Simplifying New Technology

Exercise: Scapy



Chapter 15: Warning

By the end of the chapter you will be able to:

✓ Use try and finally.

✓ Use with.

Systems & Network Training
Simplifying New Technology

Error checking is essential

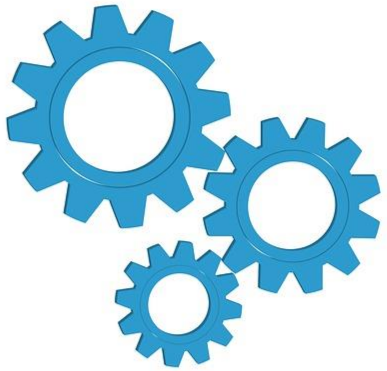
Most code in this course is to simplify code

“Industrial” level code needs error checking throughout

And also test test test

Manually test then automate

```
enable
conf t
…
```



Beware when using in band management

Systems & Network Training
Simplifying New Technology

Reading a traceback

```
Traceback (most recent call last):
  File "./cvers2.py", line 18, in <module>
    tn = telnetlib.Telnet(host)
  File "/usr/lib/python2.7/telnetlib.py", line 211, in __init__
    self.open(host, port, timeout)
  File "/usr/lib/python2.7/telnetlib.py", line 227, in open
    self.sock = socket.create_connection((host, port), timeout)
  File "/usr/lib/python2.7/socket.py", line 575, in create_connection
    raise err
socket.error: [Errno 101] Network is unreachable
```

Start at bottom and work back

Try finally

```
f = open("hello.txt")
try:
    for line in f:
        print line
finally:
    f.close()
```

with

```
with open(“foo.txt”, “w”) as f:
    f.write(“hello world!”)
```

Chapter 16: Optional - Writing your own functions and classes

By the end of the chapter you will be able to:

- ✓ Write your own functions.
- ✓ Write your own classes.

Systems & Network Training
Simplifying New Technology

What are you?

Your role will affect your code

Sys admin

Programmer

Network admin

Module structure and layout

Startup line (UNIX)
Module documentation
Module imports
Variable declarations
Class declarations
Function declarations
“main” body

Systems & Network Training
Simplifying New Technology

Writing your own functions

```
def function_name(arguments):
    “Doc string”
    Function body
```

Could be many args
x, y, z

```
def hello (x):
        “A simple function”
        text = "Hello, " + x + "!"
        print text
```

```
>>> hello ("World")
Hello, World!
```

Returning values

```
def function_name(arguments):
        “Doc string”
        Function body
```

```
def hello (x):
        “A simple function”
        text = "Hello, " + x + "!"
        return text
```

```
>>> hello ("World")
>>> str = hello(“World”)
```

Default arguments

```
def netcon (host="localhost", port=80):
        "A function with two default args"
        …

netcon () # Connects to localhost on port 80
netcon(port=8080) #Changes second arg
```

Variable arguments: Non keyword tuples

Systems & Network Training
Simplifying New Technology

```
def tupf(arg1, *therest):
“””One normal arg then any number of args”””
print arg1
for arg in therest:
    print arg
```

```
>>> tupf (1)
…
>>> tupf(1, 2, 3, 4)
```

Systems & Network Training
Simplifying New Technology

Variable arguments: Keyword dictionary

```
def tupf(arg1, **therest):
“One normal arg then any number of args”
print arg1
for arg in therest.keys():
    print “Arg %s: %s” % (arg, str(therest[arg]))
```

```
>>> tupf (1)
…
>>> tupf(1, b=2, c=3, d=4)
```

main()

Contains module name if imported

```
if __name__ == “__main__”:
    main()
```

Classes

```
>>> class Student:
...     def __init__ (self, name, age, gender):
...         self.name   = name
...         self.age    = age
...         self.gender = gender
...
```

Quiz

1. What are the 7 parts (in order) of a Python program?

2. What is the keyword used for a function?

3. What is main()?

4. What is the first method used in a class?

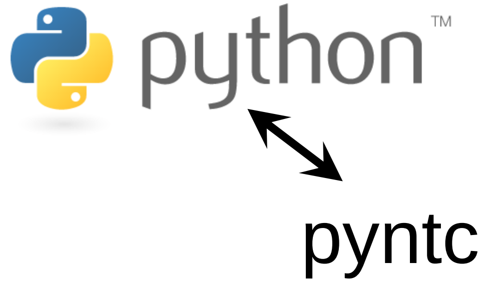
5. What is the first argument in every (almost) method?

Chapter 17: pyntc

By the end of the chapter you will be able to:

✓ Use pyntc.

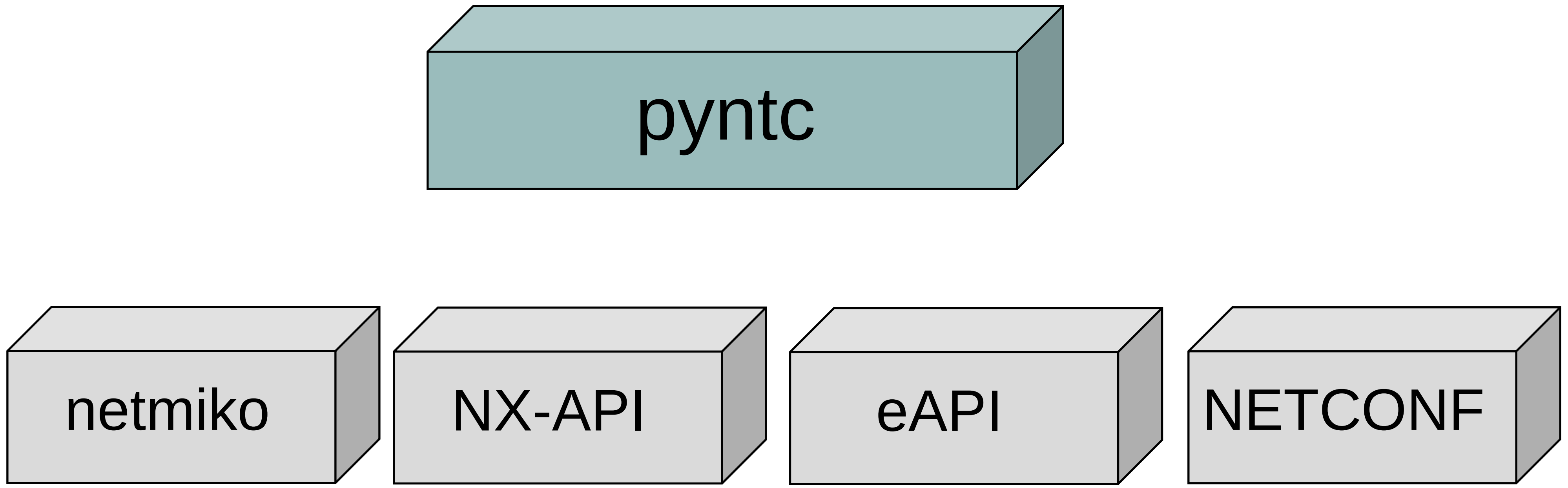
What is pyntc



https://github.com/networktocode/pyntc

pyntc is multi platform

cisco_ios_ssh
cisco_nxos_nxapi
arista_eos_eapi
juniper_junos_netconf



pyntc is for system tasks

Executing commands

Copying files

Upgrading devices

Rebooting devices

Saving / Backing Up Configs

Installing pyntc

```
pip install pyntc
```

A first pyntc program

```
import json
from pyntc import ntc_device as NTC
l2 = NTC(host='s1', username='steve', password='cisco’,
                 device_type='cisco_ios_ssh')
l2.open()

f = l2.facts
print (json.dumps(f, indent=4))

l2.close()
```

A second pyntc program

```
import json
from pyntc import ntc_device as NTC
l2 = NTC(host='s1', username='steve', password='cisco’,
                   device_type='cisco_ios_ssh')
l2.open()

l2.config_list(['hostname s1',
                    'router ospf 1',
                    'network 0.0.0.0 255.255.255.255 area 0'])
l2.close()
```

Creating instances

```
from pyntc import ntc_device as NTC
l2 = NTC(host='s1', username='steve', password='cisco',
                 device_type='cisco_ios_ssh')
```

Or .ntc.conf file:

```
[cisco_nxos_nxapi:nxos-spine1]
host: 10.22.1.1
username: steve
password: cisco
transport: http

[cisco_ios_ssh:csr1]
host: 172.16.1.1
username: steve
password: cisco
port: 22
```

Or

```
csr1 = NTCNAME('csr1')
nxs1 = NTCNAME('nxos-spine1')
```

pyntc methods: facts and show

Gathering Facts

```
>>> csr1 = NTCNAME('csr1')
>>>
>>> print json.dumps(csr1.facts, indent=4)
```

show method

Note: API enabled devices return JSON by default

```
>>> nxs1.show('show hostname')
{'hostname': 'nxos-spine1'}
```

Use raw_text=True to get unstructured data from the device

```
>>> nxs1.show('show hostname', raw_text=True)
'nxos-spine1 \n'
```

pyntc methods: show_list

Systems & Network Training
Simplifying New Technology

show_list method (with multiple commands)

```
>>> cmds = ['show hostname', 'show run int Eth2/1']

>>> data = nxs1.show_list(cmds, raw_text=True)
>>> for d in data:
...   print d
...
```

pyntc methods: Config commands

config and config_list

```
>>> csr1.config('hostname testname')
>>>
>>> csr1.config_list(['interface Gi3', 'shutdown'])
>>>
```

Viewing Running/Startup Configs

```
>>> run = csr1.running_config
>>> print run
```

file_copy method

```
>>> devices = [csr1, nxs1]
>>> for device in devices:
...   device.file_copy('newconfig.cfg')
```

pyntc methods: saving and backups

save method

copy run start for Cisco/Arista and commit for Juniper

```
>>> csr1.save()
True
```

You can also do the equivalent of copy running-config <filename> by specifying a filename:

```
>>> csr1.save('mynewconfig.cfg')
True
```

Backup current running configuration and store it locally

```
>>> csr1.backup_running_config('csr1.cfg')
>>>
```

pyntc methods: Reboot and install OS

Systems & Network Training
Simplifying New Technology

Reboot

Parameters:
  timer=0 by default
  confirm=False by default

```
>>> csr1.reboot(confirm=True)
>>>
```

Installing Operating Systems

```
>>> device.install_os('nxos.7.0.3.I2.1.bin')
>>>
```

Full workflow example:

```
>>> device.file_copy('nxos.7.0.3.I2.1.bin')
>>> device.install_os('nxos.7.0.3.I2.1.bin')
>>> device.save()
>>> device.reboot()          # IF NEEDED, NXOS automatically reboots
>>>
```

Quiz

1. What is pyntc?
2. How does pyntc access Cisco devices?
3. How does pyntc access Juniper devices?
4. How does pyntc differ from NAPALM?
5. How do you install pyntc?
6. List three main tasks pyntc can be used for.

Chapter 18: Nornir

By the end of the chapter you will be able to:

- ✓ Recognise when to use Nornir.
- ✓ Use Nornir.

What is Nornir?

Network automation framework

Inventory

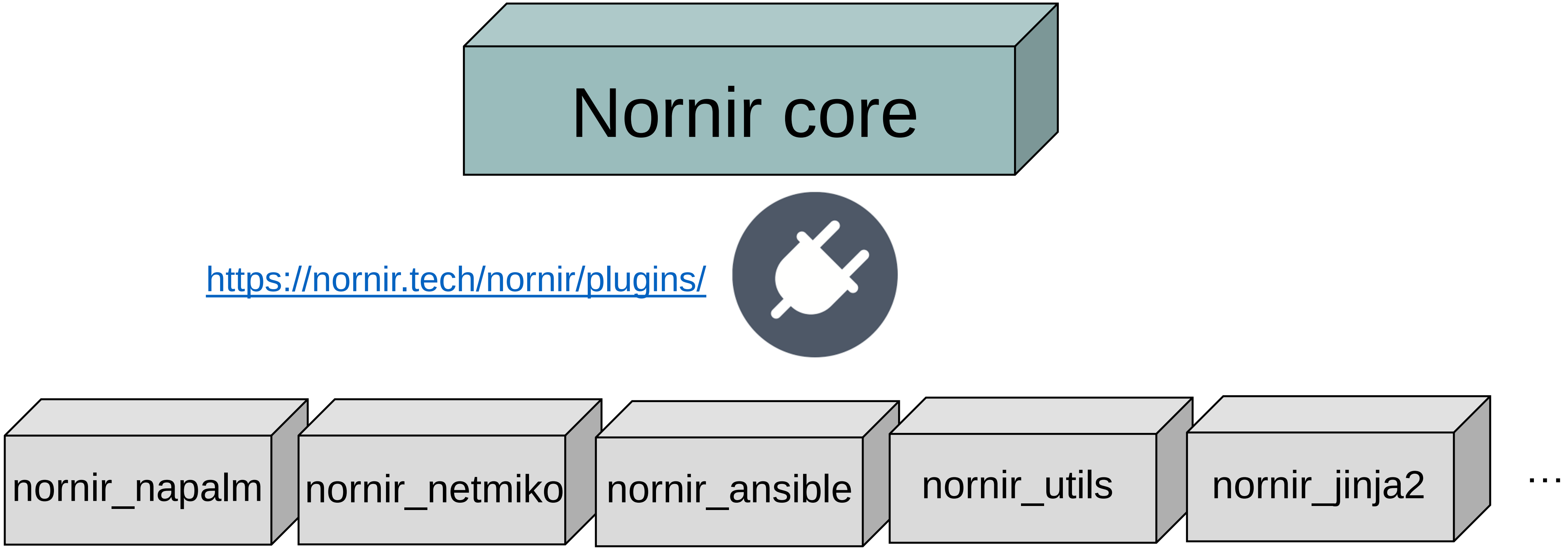
```
for rtr in [“r10”, “r11”, “r12”]:
      print (“Connecting to: ”, rtr)
```

Connection management

Parallelization

Nornir architecture

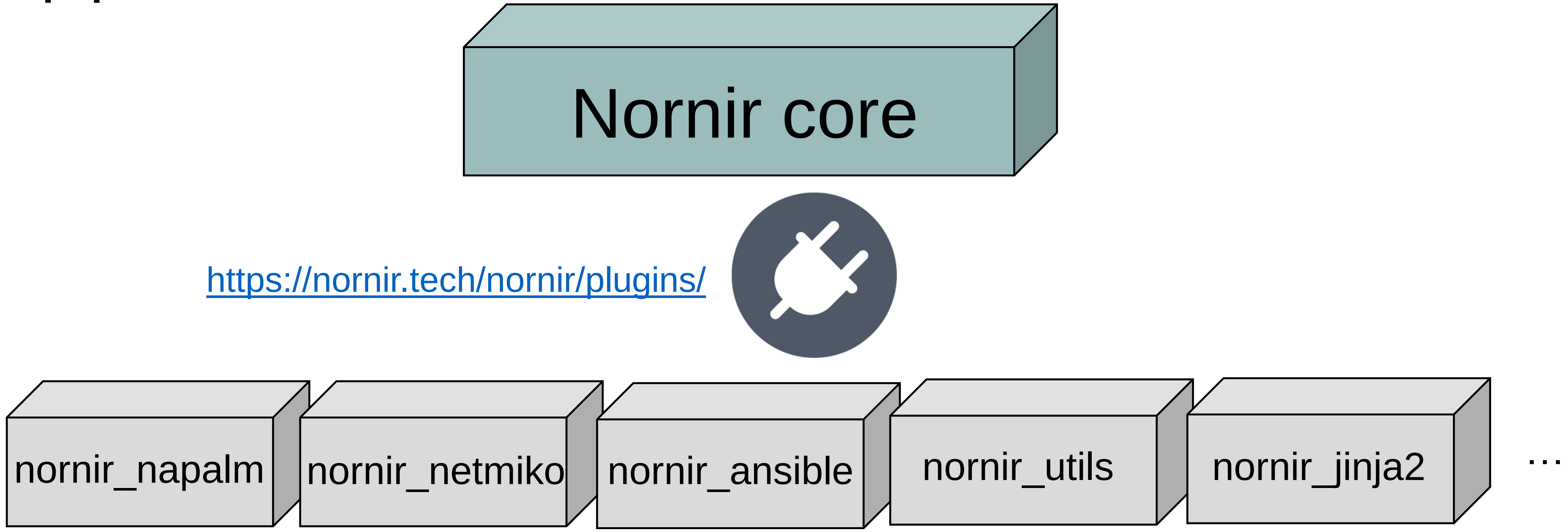
https://nornir.tech/nornir/plugins/



Installing Nornir

With Nornir 3 you also need to install plugins

```
pip install nornir
```



https://nornir.tech/nornir/plugins/

```
pip install nornir_napalm
pip install nornir_netmiko
pip install nornir_utils
:
```

Nornir setup

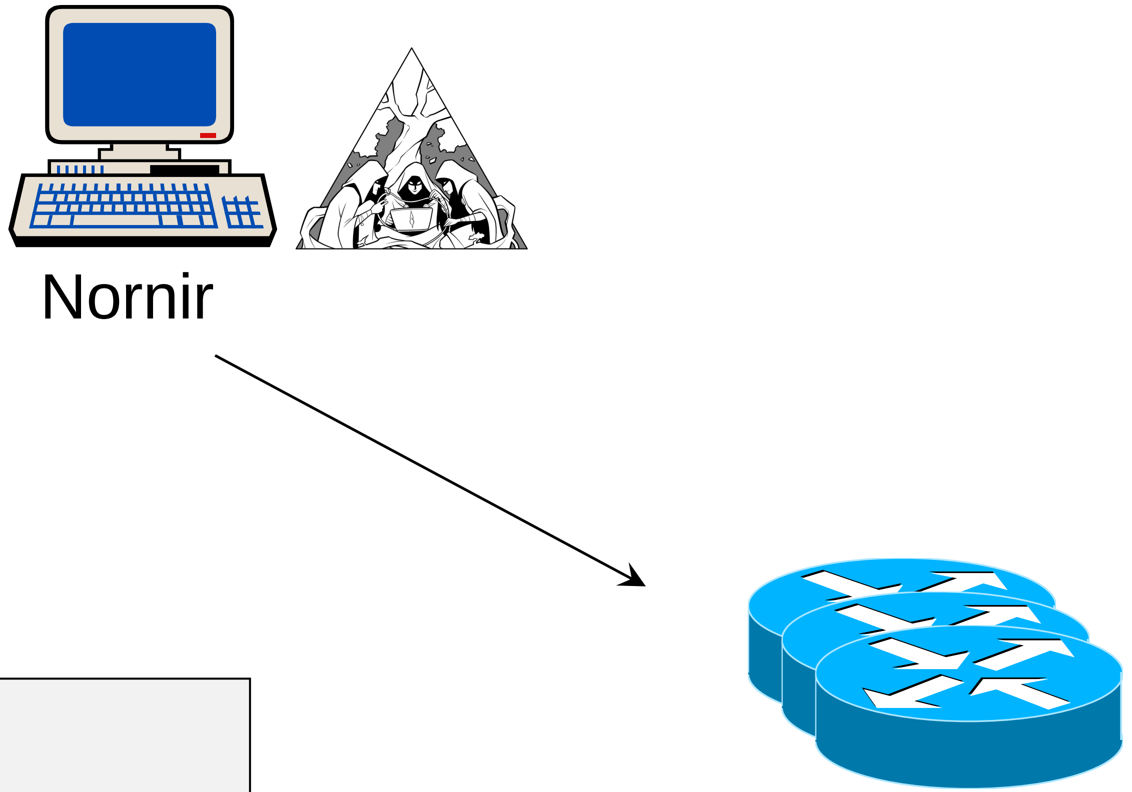
config.yml ↗ YAML format

inventory

hosts.yml

defaults.yml

groups.yml



3 rules of YAML

1. Indentation
   Represents relationships

2. Colons
   Dictionaries (key: value)

3. Dashes
   A list of items

Systems & Network Training
Simplifying New Technology

Example nornir script

```
from nornir import InitNornir
from nornir_utils.plugins.functions import print_result
from nornir_napalm.plugins.tasks import napalm_get

nr = InitNornir(
    config_file="nornir.yaml", dry_run=True
)

results = nr.run(
    task=napalm_get, getters=["facts", "interfaces"]
)
print_result(results)
```

Quiz

1. What is Nornir?
2. How does Nornir3 mainly differ from Nornir2?
3. What are the two main things Nornir provides?
4. Which should you use ansible or Nornir?
5. What are the 4 configuration files of Nornir?
6. How does Nornir connect to devices?

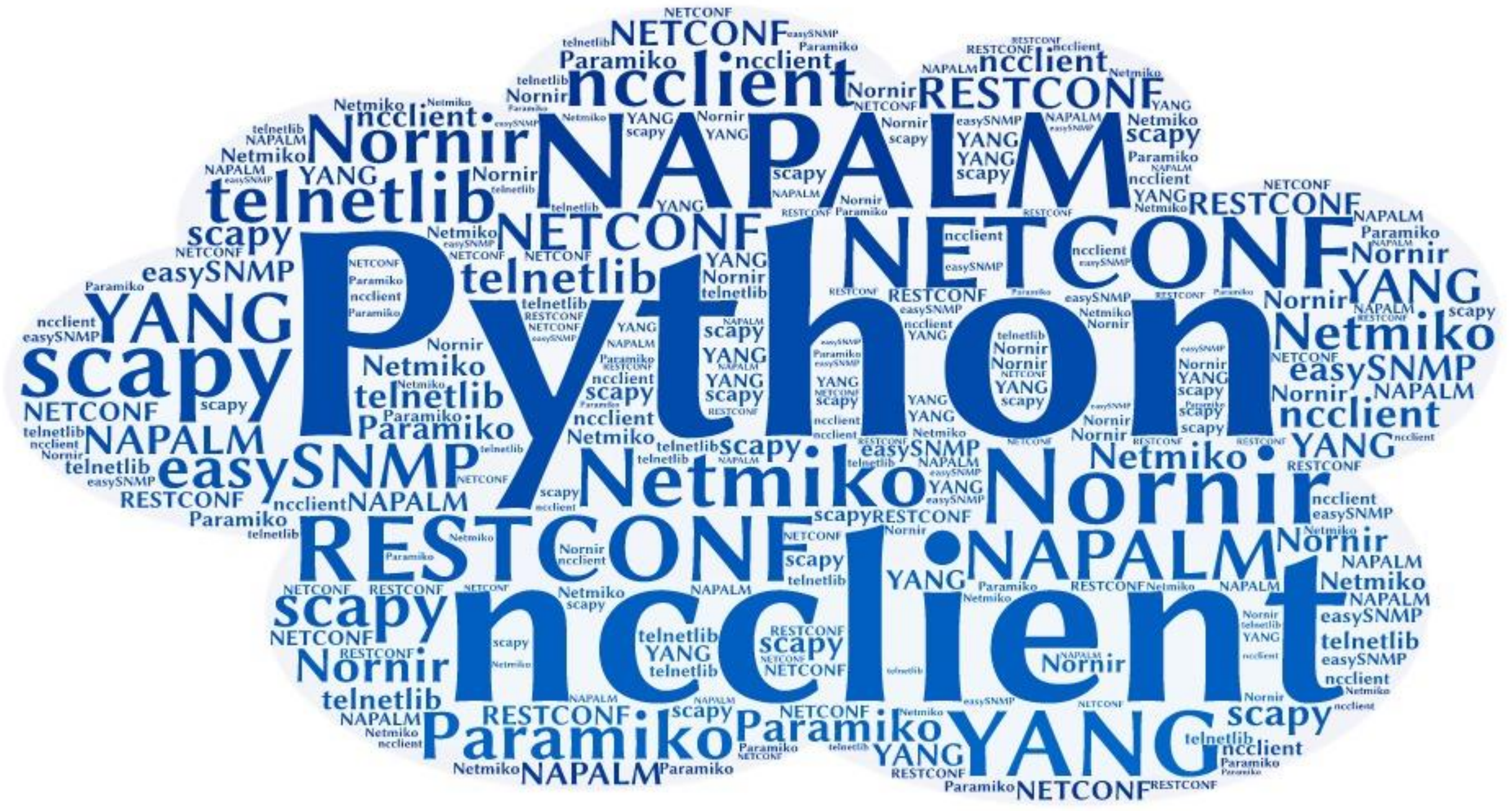
Exercise: Nornir

Chapter 19: Summary

By the end of the chapter you will be able to:

✓ Go home.

Summary


Python
ncclient
NAPALM
NETCONF
Nornir
RESTCONF
Netmiko
telnetlib
scapy
YANG
easySNMP
Paramiko

Just do it

Start small, progress from there.

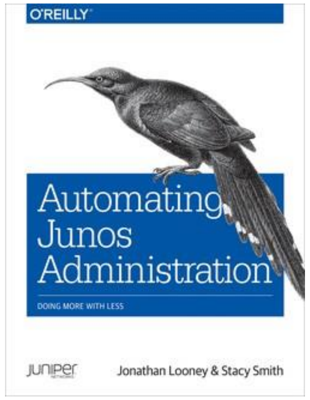
E.g.
Step 1: Read only
Step 2: Automate labs
Step 3: git

Why procrastinate today when you can do that tomorrow.

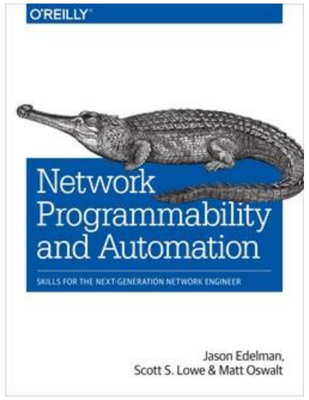
Systems & Network Training
Simplifying New Technology

Books – but they get dated

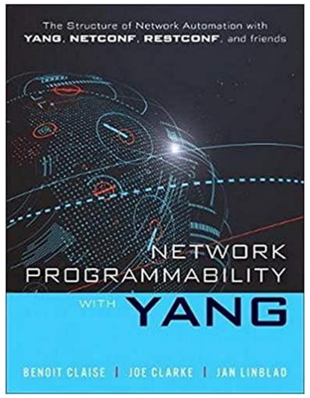
2016


O'REILLY
Automating Junos Administration
DOING MORE WITH LESS
JUNIPER
Jonathan Looney & Stacy Smith

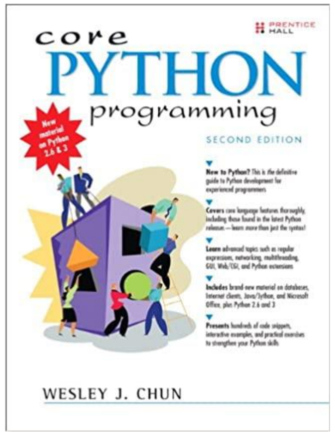
2018


O'REILLY
Network Programmability and Automation
SKILLS FOR THE NEXT-GENERATION NETWORK ENGINEER
Jason Edelman, Scott S. Lowe & Matt Oswalt

2019


The Structure of Network Automation with YANG, NETCONF, RESTCONF, and friends
NETWORK PROGRAMMABILITY WITH YANG
BENOIT CLAISE | JOE CLARKE | JAN LINDBLAD

2006

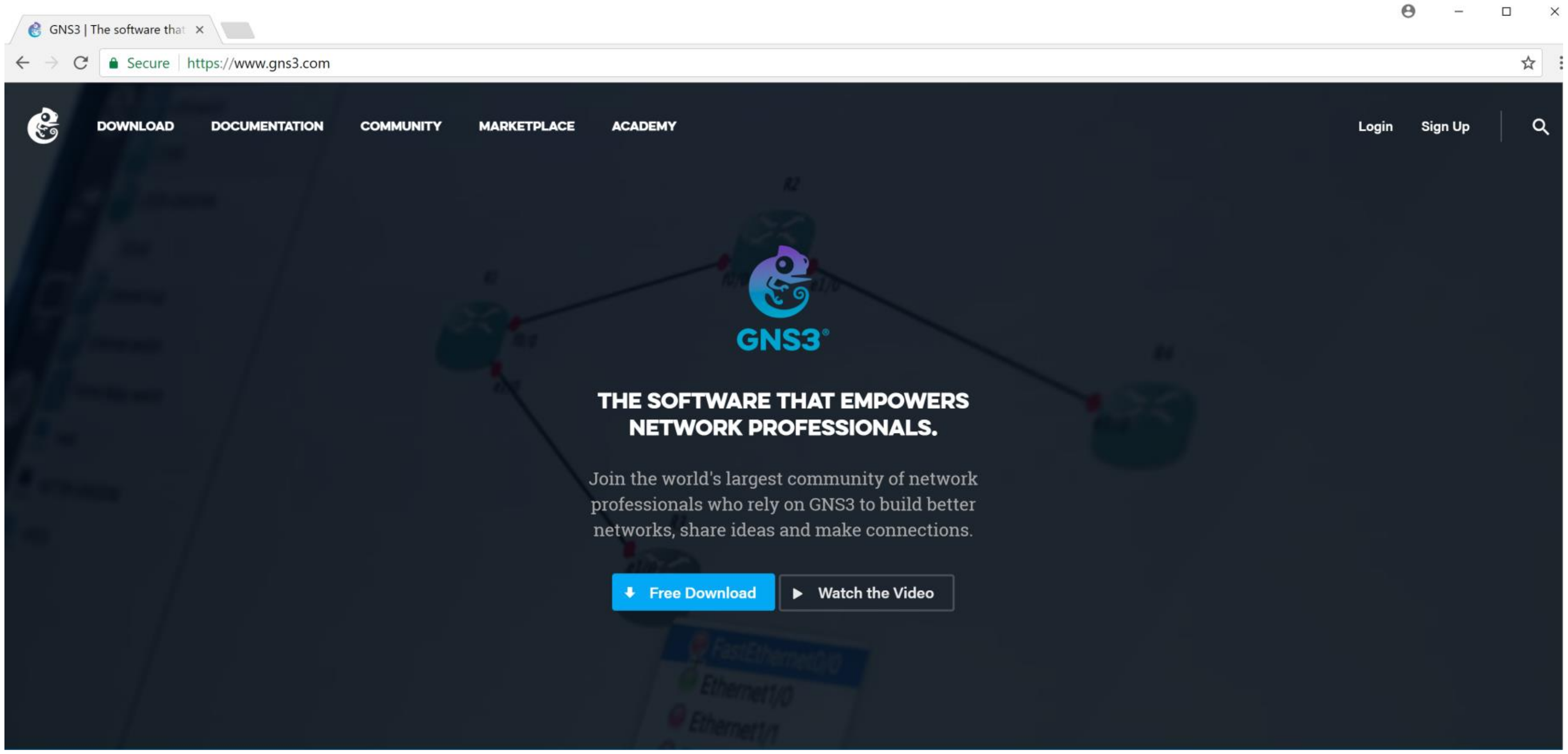


Appendix: GNS3

By the end of the chapter you will be able to:

- ✓ Install GNS3.
- ✓ Configure GNS3.
- ✓ Recognise the role of GNS3 in network DevOps.

What is GNS3?

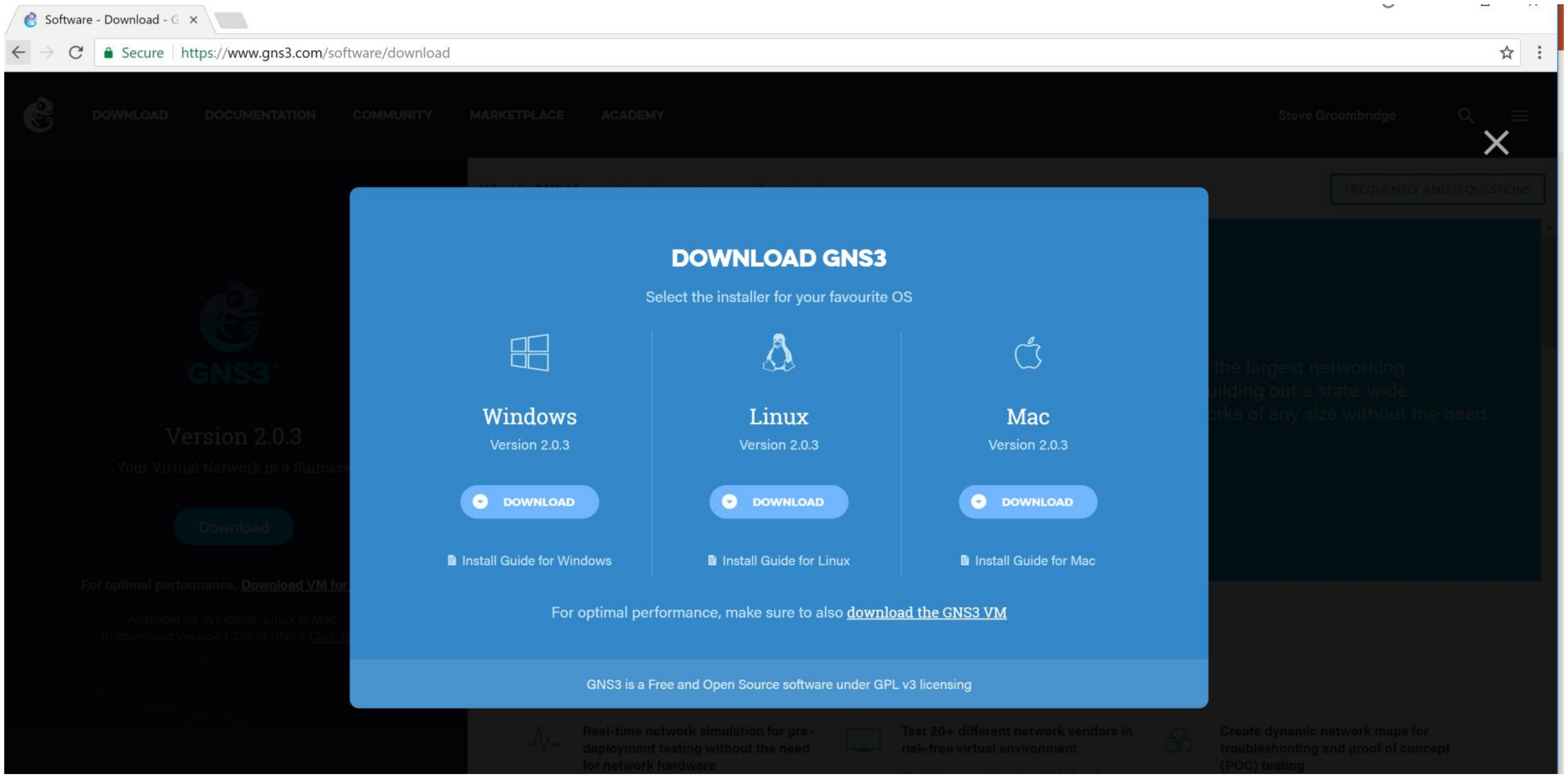


GNS3 | The software that
Secure https://www.gns3.com

THE SOFTWARE THAT EMPOWERS NETWORK PROFESSIONALS.

Join the world's largest community of network professionals who rely on GNS3 to build better networks, share ideas and make connections.

Installing GNS3



Software - Download - G

https://www.gns3.com/software/download

DOWNLOAD GNS3

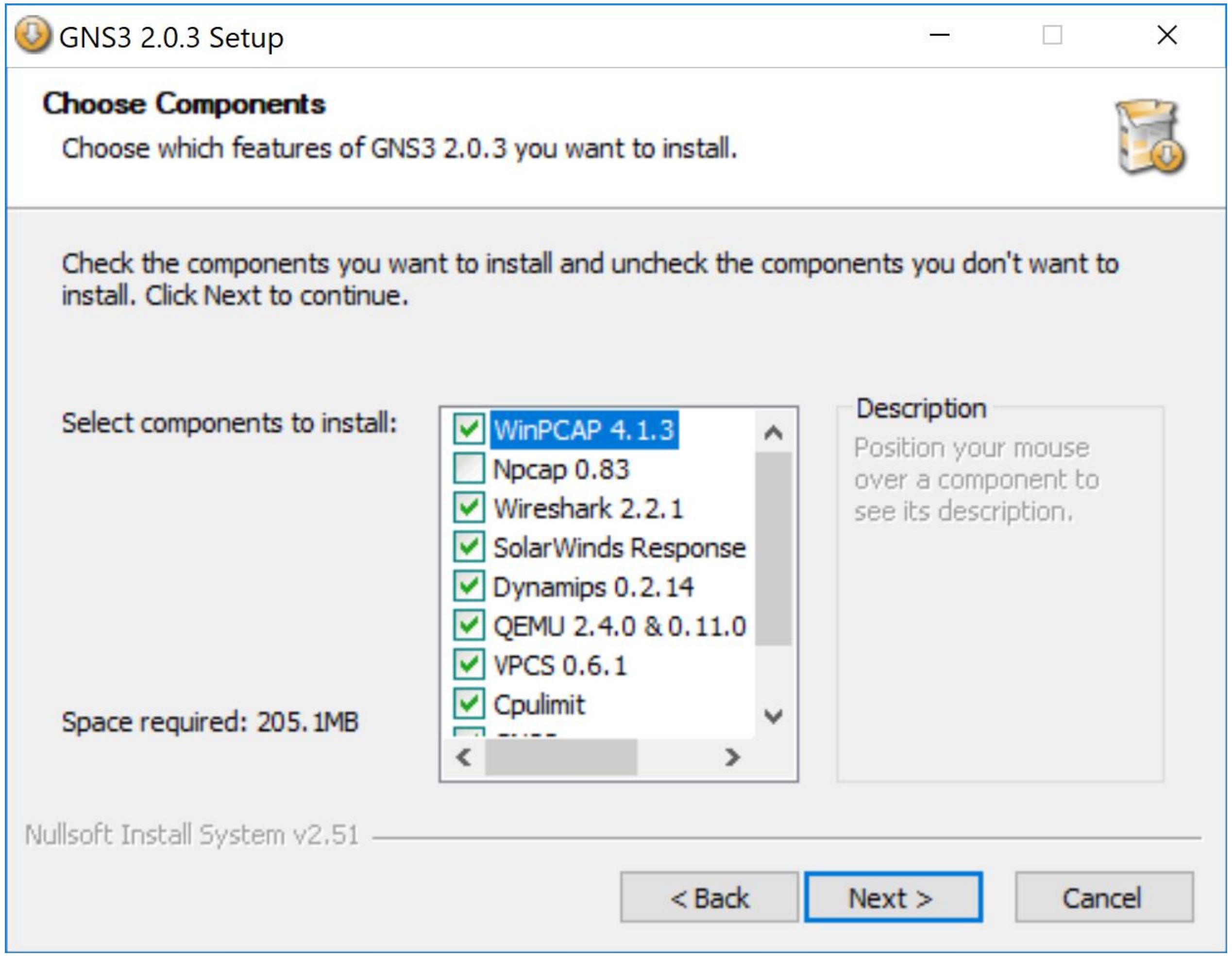
Select the installer for your favourite OS

For optimal performance, make sure to also download the GNS3 VM

GNS3 is a Free and Open Source software under GPL v3 licensing

Systems & Network Training
Simplifying New Technology

GNS3 components


GNS3 2.0.3 Setup

Choose Components

Choose which features of GNS3 2.0.3 you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select components to install:

- [✓] WinPCAP 4.1.3
- [ ] Npcap 0.83
- [✓] Wireshark 2.2.1
- [✓] SolarWinds Response
- [✓] Dynamips 0.2.14
- [✓] QEMU 2.4.0 & 0.11.0
- [✓] VPCS 0.6.1
- [✓] Cpulimit

Space required: 205.1MB

Description

Position your mouse over a component to see its description.

Nullsoft Install System v2.51

< Back | Next > | Cancel

Systems & Network Training
Simplifying New Technology

Two ways to use

GNS3 GUI

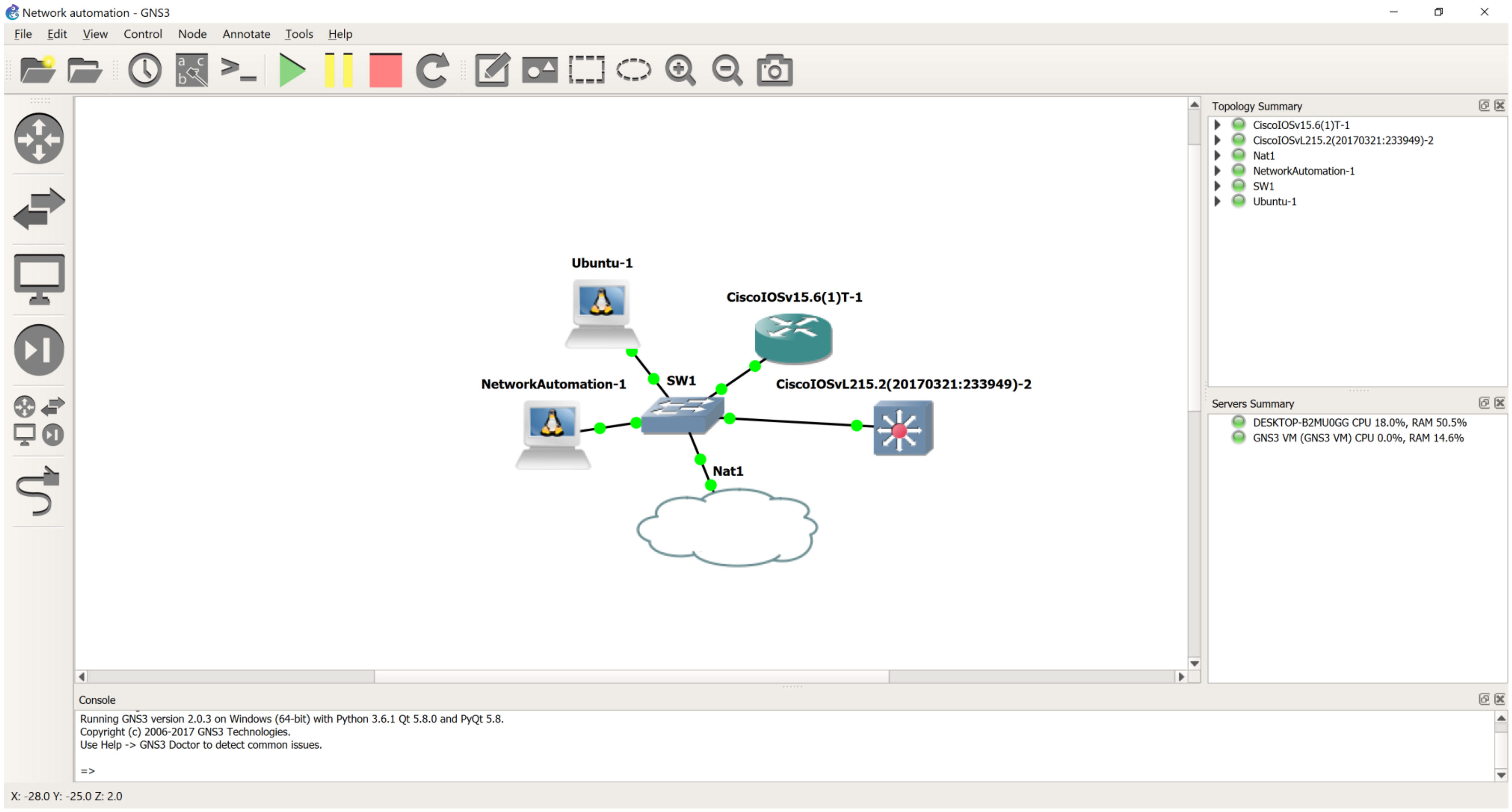
GNS3 VM

Needs VMWare workstation or player

Can use others but…

Also need device OS’s

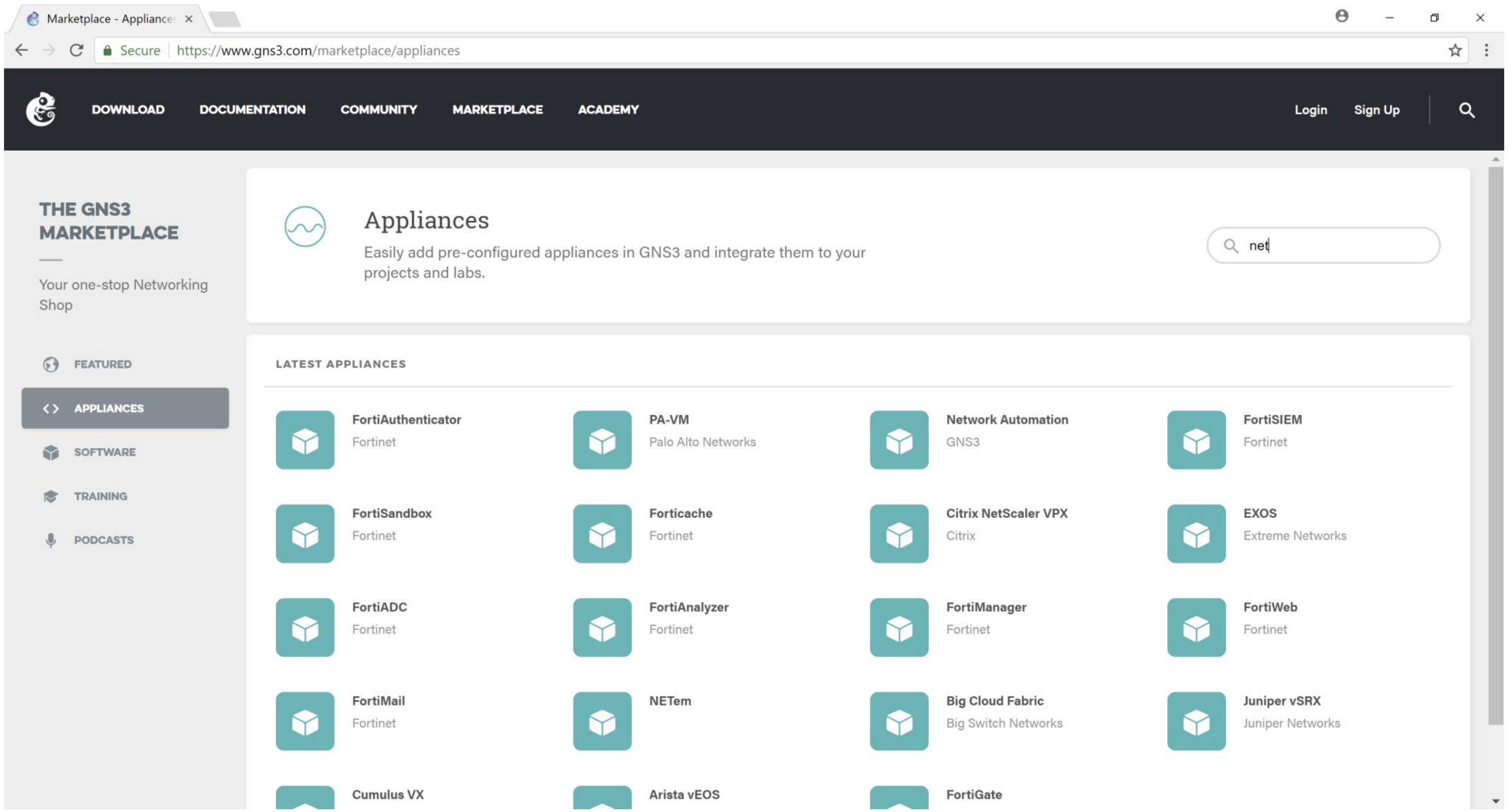
Using GNS3



DESKTOP-B2MU0GG CPU 18.0%, RAM 50.5%
GNS3 VM (GNS3 VM) CPU 0.0%, RAM 14.6%

```
Running GNS3 version 2.0.3 on Windows (64-bit) with Python 3.6.1 Qt 5.8.0 and PyQt 5.8.
Copyright (c) 2006-2017 GNS3 Technologies.
Use Help -> GNS3 Doctor to detect common issues.

=>
```

GNS3 marketplace



THE GNS3 MARKETPLACE

Your one-stop Networking Shop

Appliances

Easily add pre-configured appliances in GNS3 and integrate them to your projects and labs.

LATEST APPLIANCES

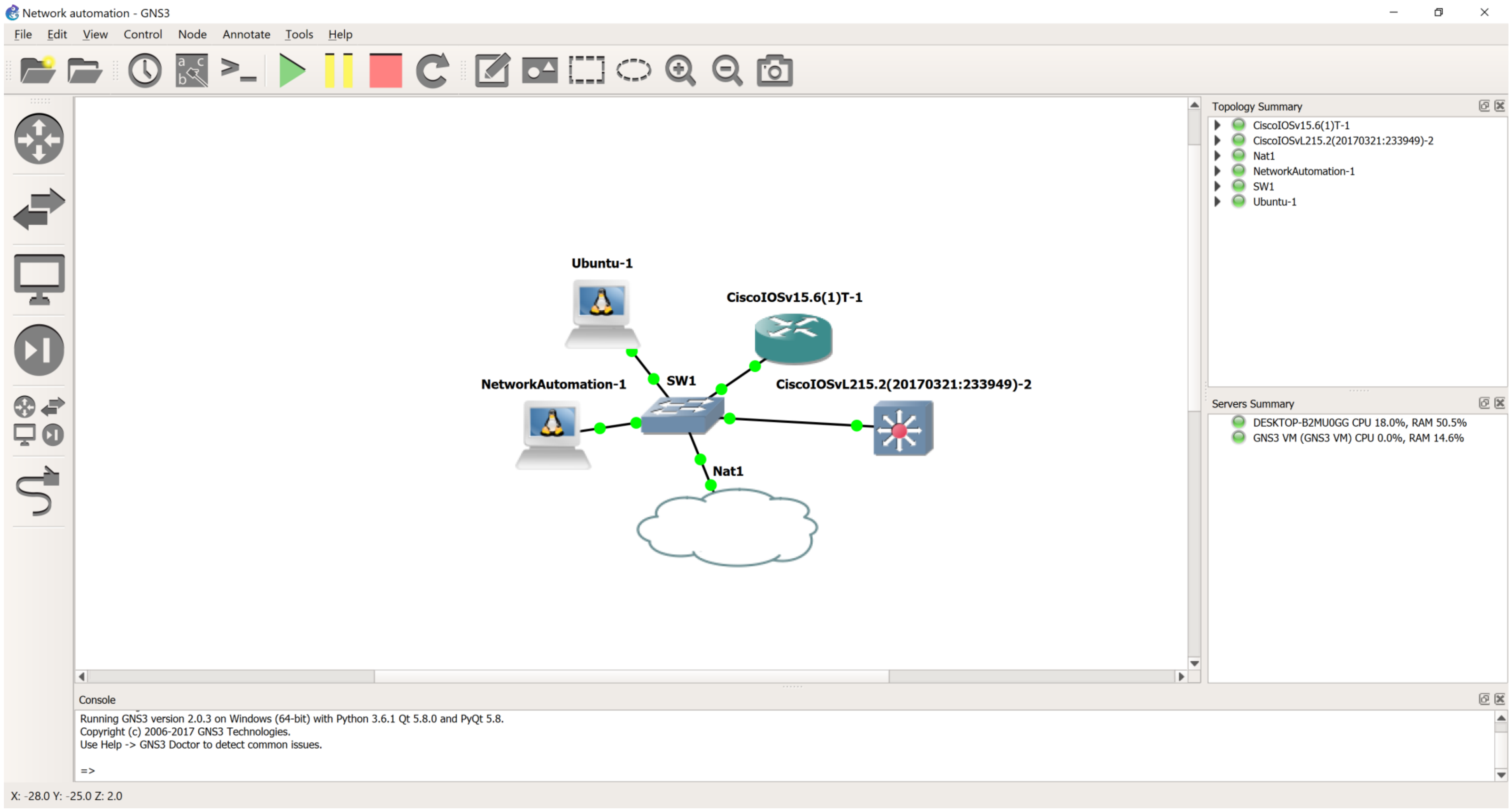
FortiAuthenticator Fortinet	PA-VM Palo Alto Networks	Network Automation GNS3	FortiSIEM Fortinet
FortiSandbox Fortinet	Forticache Fortinet	Citrix NetScaler VPX Citrix	EXOS Extreme Networks
FortiADC Fortinet	FortiAnalyzer Fortinet	FortiManager Fortinet	FortiWeb Fortinet
FortiMail Fortinet	NETem	Big Cloud Fabric Big Switch Networks	Juniper vSRX Juniper Networks
Cumulus VX	Arista vEOS	FortiGate	

Systems & Network Training
Simplifying New Technology

Quiz

1. What is GNS3?
2. What are the two modes of using GNS3 and which is recommended?
3. What does the NAT appliance do?
4. What is the network automation appliance?
5. What operating system does the network appliance use?

Exercise



Network automation - GNS3
File Edit View Control Node Annotate Tools Help
Ubuntu-1
CiscoIOSv15.6(1)T-1
NetworkAutomation-1
SW1
CiscoIOSvL215.2(20170321:233949)-2
Nat1
Topology Summary
Servers Summary
Console

Topology Summary
- CiscoIOSv15.6(1)T-1
- CiscoIOSvL215.2(20170321:233949)-2
- Nat1
- NetworkAutomation-1
- SW1
- Ubuntu-1

Servers Summary
- DESKTOP-B2MU0GG CPU 18.0%, RAM 50.5%
- GNS3 VM (GNS3 VM) CPU 0.0%, RAM 14.6%

```
Running GNS3 version 2.0.3 on Windows (64-bit) with Python 3.6.1 Qt 5.8.0 and PyQt 5.8.
Copyright (c) 2006-2017 GNS3 Technologies.
Use Help -> GNS3 Doctor to detect common issues.

=>
```

X: -28.0 Y: -25.0 Z: 2.0

Exercise: Set IP address in Linux

```
NetworkAutomation-1 console is now available... Press RETURN to get started.
root@NetworkAutomation-1:~# cat /etc/network/interfaces
#
# This is a sample network config uncomment lines to configure the network
#

# Static config for eth0
#auto eth0
#iface eth0 inet static
#       address 192.168.0.2
#       netmask 255.255.255.0
#       gateway 192.168.0.1
#       up echo nameserver 192.168.0.1 > /etc/resolv.conf

# DHCP config for eth0
 auto eth0
 iface eth0 inet dhcp
```

Exercise: Enable ssh on Cisco devices

Systems & Network Training
Simplifying New Technology

```
hostname r11
username steve password cisco
username steve privilege 15
line vty 0 4
      login local
      transport input all
      exit
ip domain-name snt.co.uk
crypto key generate rsa
int gi 0/0
ip add 192.168.122.11
no shut
end
copy run start
```

Exercise: Enable ssh on Juniper devices

Systems & Network Training
Simplifying New Technology

```
configure
set system host-name j1
set system login user steve class super-user
    full-name “steve” authentication plain-text-password
set system services ssh
set system root-authentication plain-text-password
set interfaces ge-0/0/0 unit 0 family inet address 10.1.1.1/8
commit and-quit
```

Systems & Network Training
Simplifying New Technology

Some more Juniper

```
delete security
set security forwarding-options family
                 mpls mode packet-based
commit and-quit
```