Wi-Fi automation lab

Day 1 presentations

Andreas Koksrud / Telenor Bedrift

Co-authors:

Kjetil Teigen Hansen & François Vergès

telenor

References / inspiration

https://github.com/CiscoDevNet/yangsuite/
https://postman.com
https://docs.ansible.com/ansible/latest/collections/cisco/ios/ios_facts_module.html
https://www.wifireference.com/2020/01/14/viewing-network-telemetry-from-the-catalyst-9800-with-grafana/
https://grafana.com/grafana/dashboards/13462-device-health-monitoring/
https://grafana.com/grafana/dashboards/12468-catalyst-9800-client-stats/
https://wirelessisfun.wordpress.com/2020/12/10/network-telemetry-data-and-grafana-part-1-the-advanced-netconf-explorer/
https://python.org
https://codeium.com
https://canonical.com/multipass
https://blog.apnic.net/

Copyright

Any slides marked Kjetil Teigen Hansen or the Conscia logo will also have full or co-ownership by Kjetil

Any slides marked François Vergès or the SemFio logo will also have full or co-ownership by François

Prerequisites

- Cisco Meraki account (https://dashboard.meraki.com)
- Juniper MIST account (https://manage.mist.com)
- Postman account (https://postman.com)
- Complete the pre-lab exercises (this document) before the deep dive labs
- Bring an Ethernet dongle if you don't have built-in port
- (optional) Bring an extra screen to show lab guide (or prepare to Alt-Tab)

Communications

- WebEx space: Wi-Fi automation lab
- Please help each other
- Sharing is caring ☺

Agenda

Pre-lab

- Choose your hypervisor
- Install Ubuntu Server w/Docker
- Install Postman
- Install VS Code
- (optional) install 9800-CL

Day 1

- Sort out pre-lab task problems
- Get to know the lab environment
- Connect VS Code to Ubuntu
- Install and explore Ansible
- Explore Python automation
- Install and explore YANG Suite
- Explore Postman
- Install and explore Grafana

Day 2

- In-depth explore a topic of choice
  - Grafana / TIG-stack
  - Grafana Cloud
  - Ansible
  - Python
  - Cloud vendor automation (MIST or Meraki)

Agenda - About Day 2 choices...

- Choose a track

  Horiontal - A taste of everything (try to cut at 45min for each)

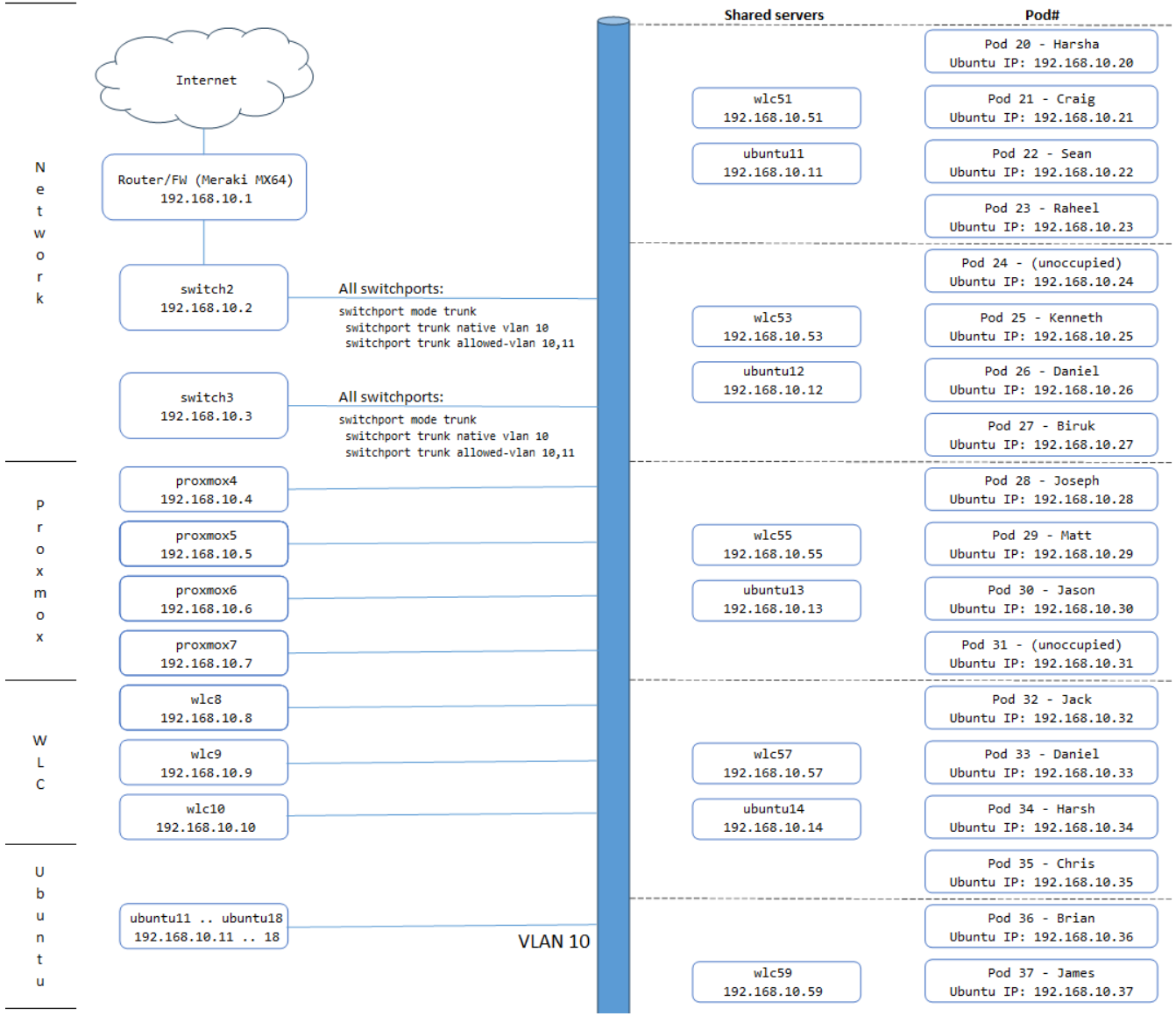
  Vertical - 3h deep dive into one of the topics

Ansible	Python	Grafana	Cloud APs
• Run a CLI command • CLI configuration • Using Jinja2 templates • Using RESTCONF • Write your own module • RESTCONF + own module • Organizing projects • … more to come	• Using Netmiko • Using RESTCONF • Get metrics, draw a graph • Working with AI companions • Using .env files • Nornir automation framework • More Nornir • … more to come	• Working with syslog • Finish building your own TIG stack from day 1 • Extended C9800 dashboard • … building more • … and improving even more	• MIST or Meraki • Use APIs with Postman • Use APIs with Python • Grafana Dashboard for Cloud APs

Scope

- In scope
  - Getting started with various systems/languages/solutions for lab purposes
  - Set up your own Ubuntu Linux server on your own laptop. It will be possible to use a shared server if you do not want or have possibility to install an Ubuntu VM on your laptop
  - Some nice examples to try various aspects of automation
  - Inspiring you to explore deeper on your own
- Out of scope topics
  - Git
  - Learning the languages (Ansible, Python, InfluxQL, etc)
  - Learning Linux
  - Deploying the systems for production use
  - Troubleshooting WLC/AP connection

Wi-Fi Automation deep dive - Lab topology



- Each student have an assigned Pod number. When using static IP, their Ubuntu Server should have the same last octet as the Pod#
- 2 students share a preconfigured WLC, as assigned in the topology map
- Everyone is connected to VLAN 10
- All switchports on all switches are trunk ports with VLAN 10 as native, and allowed vlan as VLAN 10 and 11
- For Wi-Fi clients on your SSIDs you can use VLAN 11 to not fill up VLAN 10

Login to shared devices

User: devnet-adm
Pass: ChangeMe2025!

IP Plan (VLAN 10)

Static adm IPs: 192.168.10.1 - 19
Ubuntus (per pod): 192.168.10.20 - 50
WLCs (shared): 192.168.10.51 - 70
DHCP range: 192.168.10.71 - 250

IP Plan (VLAN 11)

Static adm IPs: 192.168.11.1 - 10
DHCP range: 192.168.10.11 - 250

Automation in the Wi-Fi lifecycle

- Some examples of automation in different phases
- Prepare
  - Python scipt for information gathering / network mapping / device walking
- Plan
  - Python script for large-number analysis of devices (and/or maybe Excel is better for parts of it)
- Design
  - Prepare data to use in the Implement phase
- Implement
  - Python script for creating IP pools, or creating site hierarcy etc in DNAC
  - Ansible and/or some vendor specific Zero-Touch provisioning variants for device config based on some source of truth
- Operate
  - Write Python scripts for changes that involve large number of devices
  - Ansible for repeating tasks
  - Grafana for real-time troubleshooting
- Optimize
  - Grafana for monitoring/graphing/analysis
  - Python scripts or Ansible playbooks to analyze possible optimizations (APs on 100Mbit, hostnames of neighbors that differ, etc)

VirtualBox - Type 1 vs Type 2 hypervisors

Type 1

- Bare-metal (run directly on hardware)
- Superior performance
- For production use

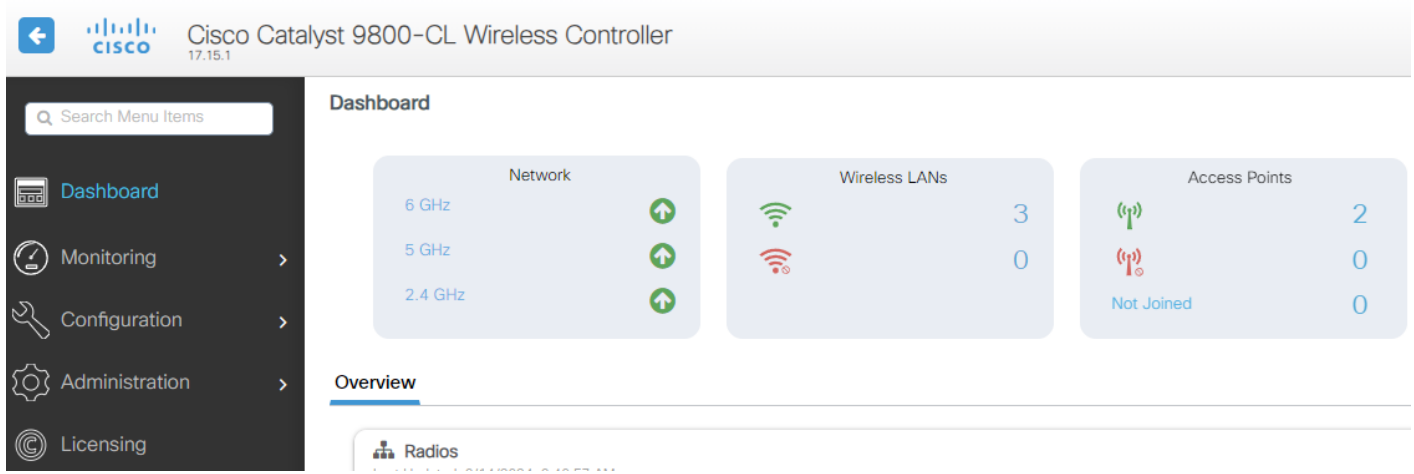
Examples
- VMWare ESXi
- Proxmox
- KVM
- Xen
- Nutanix

Type 2

- On top of another OS
- Flexible, can run on your laptop
- For lab purposes or at least non-24/7

Examples
- VirtualBox
- VMWare Workstation
- Hyper-V
- Parallels

WLAN Controllers



- This lab use Cisco 9800-CL WLC
- Project the knowledge to your platform of choice
  - Other vendors WLAN Controllers or similar concepts
  - Cloud-managed solutions often have good APIs
  - Ansible, Python, etc have lots of collections

Ubuntu Server

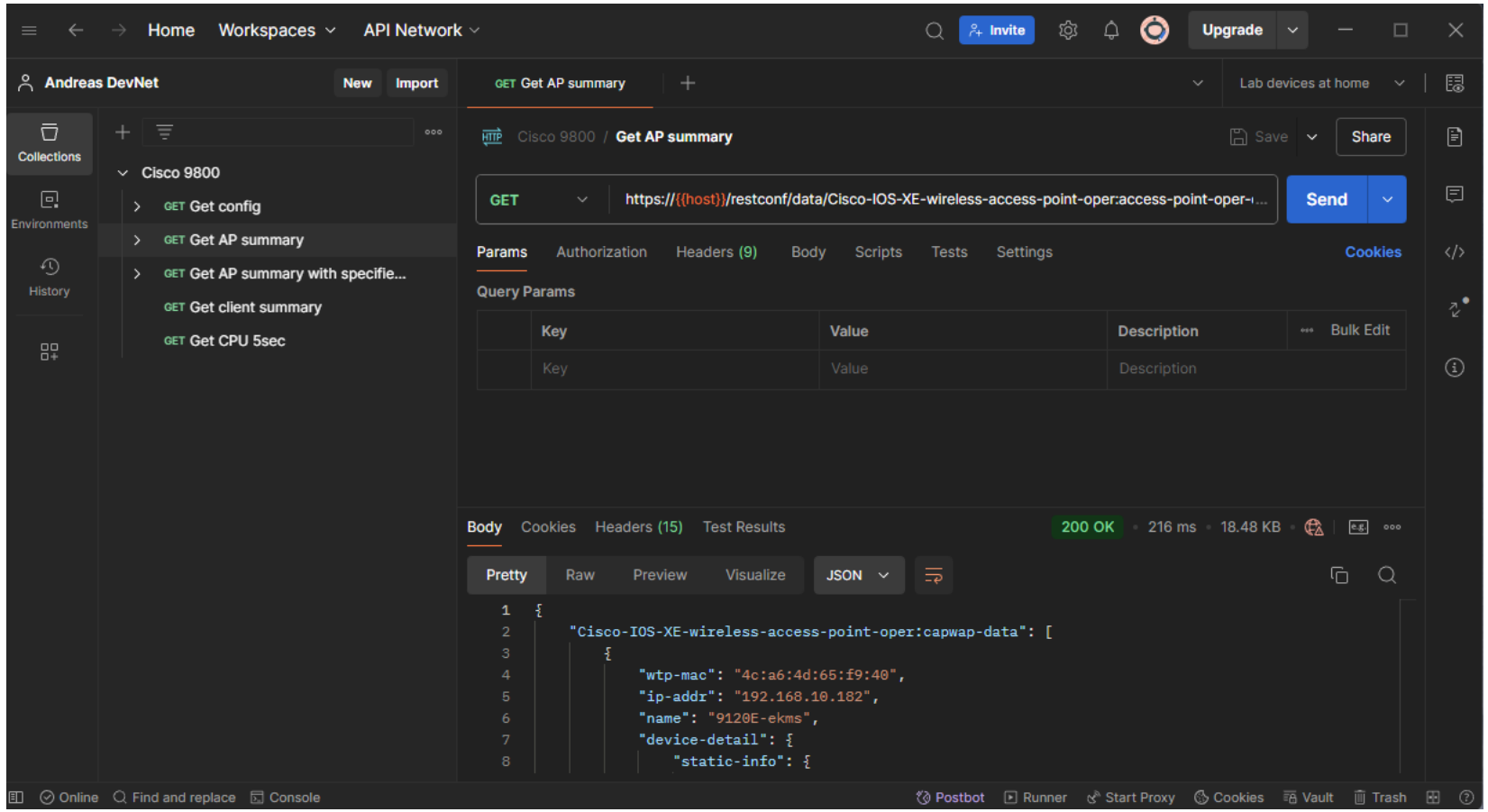
- This lab use Ubuntu Server
- Most stuff can be done directly on your own laptop
  - Currently Ansible can not run directly on Windows
  - TIG stack ... maybe(?)
- I like to play with stuff using a server, I find it easier to transfer to live systems
- WSL (Windows Subsystem for Linux) is a very good alternative for Python and Ansible. For Grafana it might be more tinkering

Docker

- This lab use Docker as the container platform
  - Pre-made packages for TIG stack and YANG Suite
  - Could just as well be using Kubernetes etc

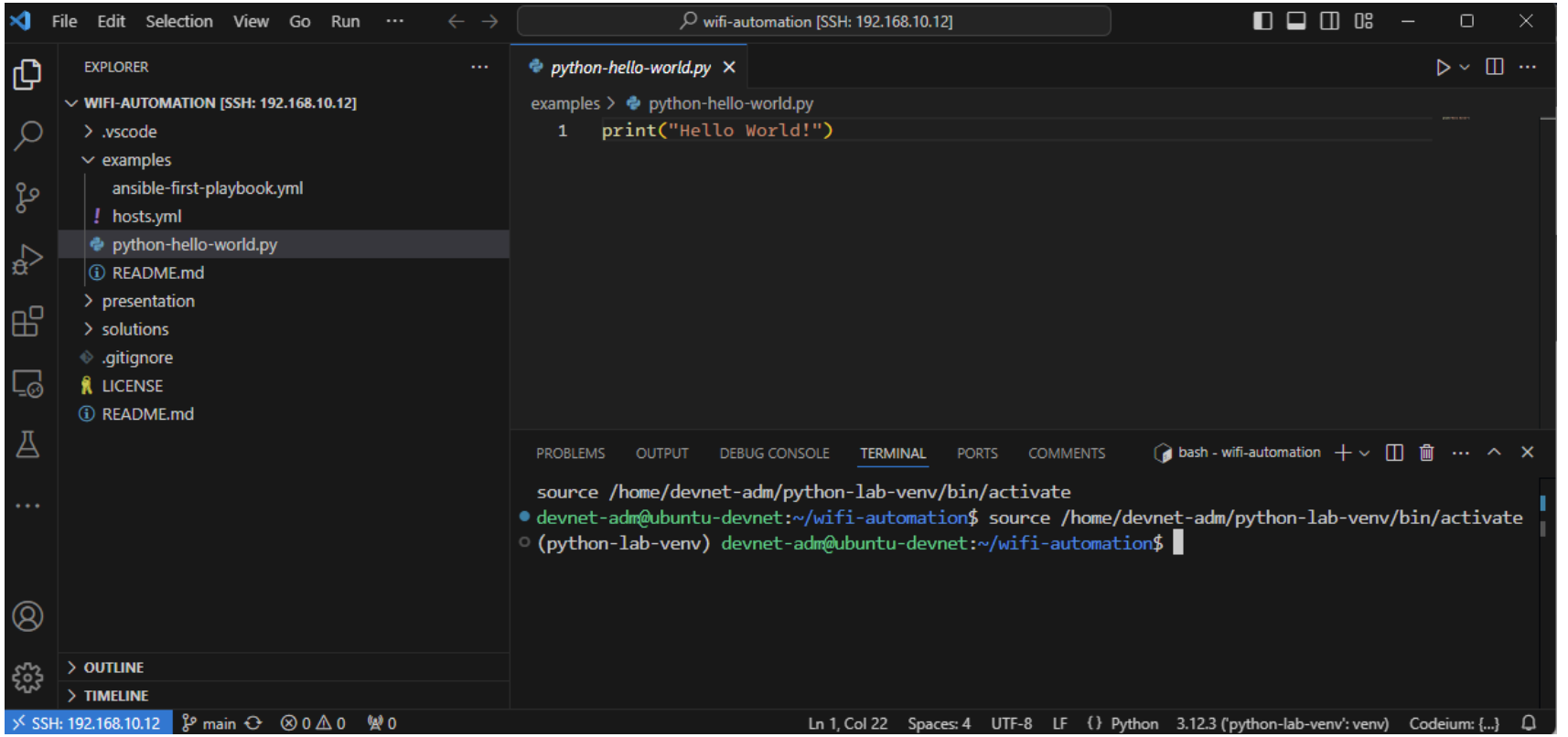
Postman

- Postman is a "go-to" tool for RESTCONF APIs. Some examples for automation are
  - Exploring RESTCONF calls
  - Checking the datastructure you get in return
  - Validating the calls to devices before implementing in Python, Ansible, etc
  - Get example code in your preferred language for that specific RESTCONF call



VS Code

- This lab use VS Code as text editor / development environment
  - Some other popular alternatives for all or parts of the process
    - VS Codium
    - Atom
    - Notepad++
    - Jupyter Notebook
    - Anaconda ecosystem



YANG Suite

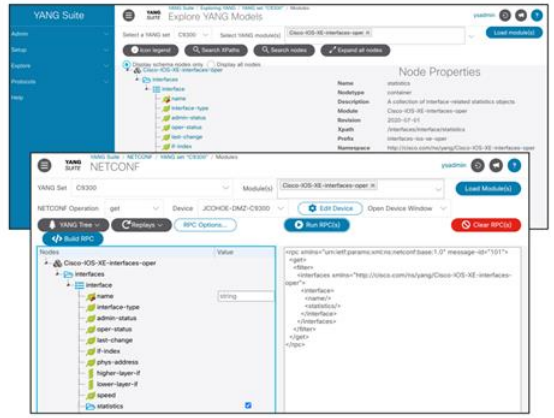
- Testing and validation environment for YANG related tasks
- Install as a docker container
- We will use this to explore the world of YANG models present in the Cisco 9800 WLC

Cisco YANG Suite

YANG API Testing and Validation Environment

Construct and test YANG based APIs over NETCONF, RESTCONF, gRPC and gNMI

IOS XE / IOS XR / NX OS platforms



Now Available !

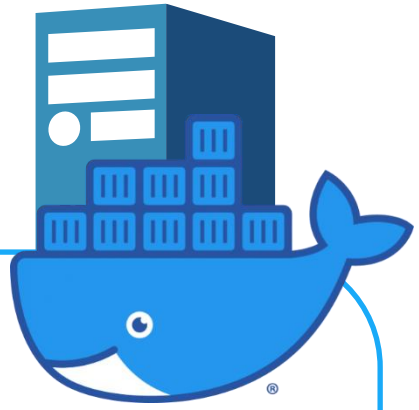
developer.cisco.com/yangsuite

github.com/CiscoDevNet/yangsuite

TIG Stack

- Bla bla about what is TIG stack

gRPC / TCP 57000



telegraf™

metrics goes to the database

influxdb TCP 8086

Grafana to visualize

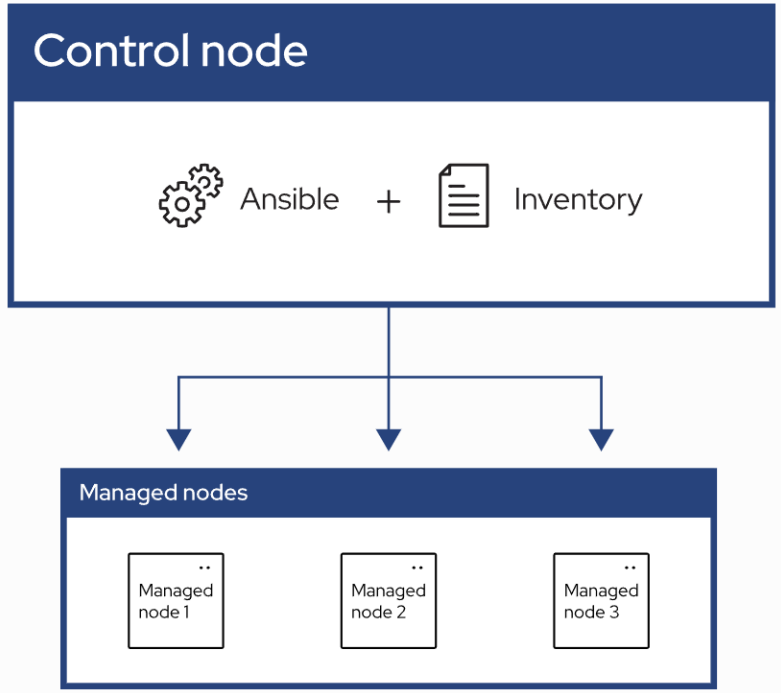
Grafana TCP 3000

WebUI – server:3000

Ansible

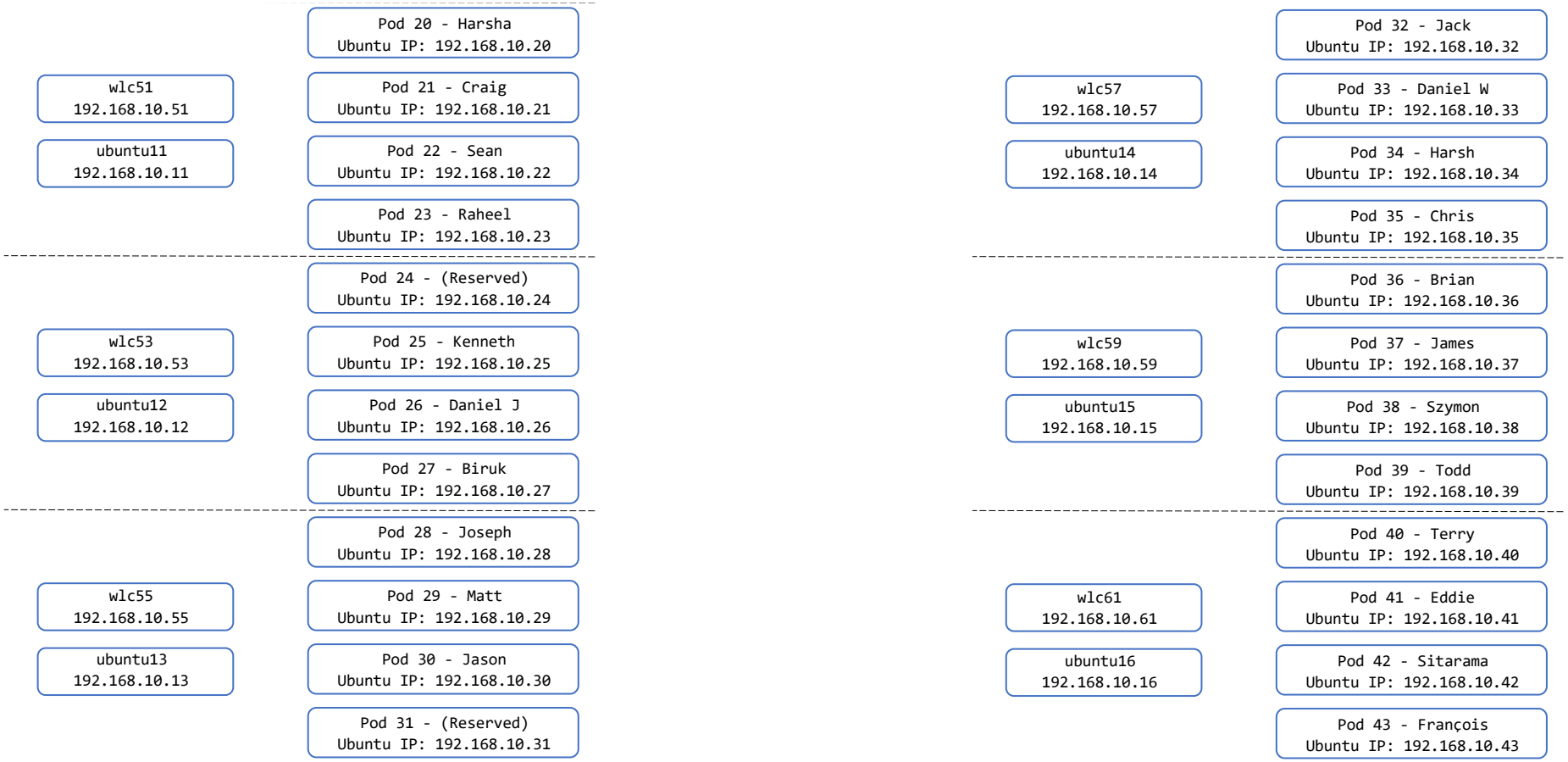
- Automation using "playbooks" written in YAML format
- Agent-less architecture
  - No installation on the managed nodes
- Idempotency
  - Does not change/write if target is already in state described by playbook

- Control node
  - System where Ansible is installed and runs the playbooks
- Inventory
  - List of managed nodes where the playbook will affect
- Managed nodes
  - Remote systems that are the targets of your playbooks



Pod numbers

- Students get pod numbers ranging from 20 and up
- 4 pods share 1 WLC and 1 preinstalled Ubuntu server



Material on github

- Everything can be downloaded from
- https://github.com/akoksrud/wifi-automation
- Some examples used in Day 1 tasks
- All presentations (PPT and PDF)
- Solutions for all exercises


```
PS C:\Users\akoksrud> git clone https://github.com/akoksrud/wifi-automation
Cloning into 'wifi-automation'...
remote: Enumerating objects: 311, done.
remote: Counting objects: 100% (72/72), done.
remote: Compressing objects: 100% (55/55), done.
remote: Total 311 (delta 34), reused 53 (delta 15), pack-reused 239 (from 1)
Receiving objects: 100% (311/311), 43.05 MiB | 34.74 MiB/s, done.
Resolving deltas: 100% (140/140), done.
PS C:\Users\akoksrud> cd wifi-automation
PS C:\Users\akoksrud> code .
```