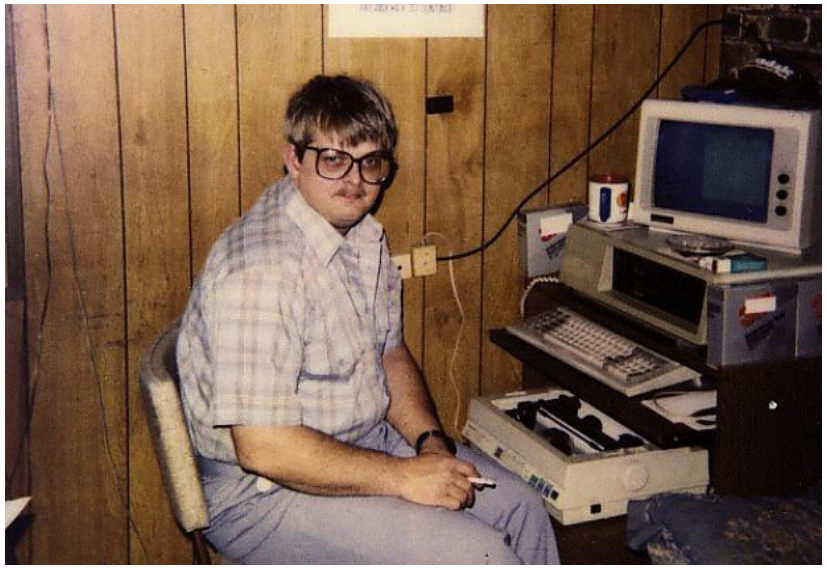
Cumulus Meetup 2019.03.26 - Lausanne

Network automation at Kudelski – Romain Aviolat

Agenda

- Network automation
- DevOps
- Return of experience (>3y)

About ME



- (social) Nerd
  - Open-source advocate
  - Linux / Network / Sec
  - Hardware hacker (µC, Drones, toasters, …)
  - Endurance-sports enthusiast
  - Don’t like products
- Team lead infrastructure team (IT structure)
  - Small cross-functional team
- Joined the group 5 years ago

Kudelski Group

- +60 years
- +3K employees on 5 continents
- 200M+ annual R&D investment

- DigitalTV (Content protection)
- Public Access
- Cyber Security
- IoT

en.wikipedia.org/wiki/Kudelski_Group

The team

(thanks guys)

"Provide multi-tenant and as-a-service infrastructure for the whole **K** group, based on new foundations. Inside Tier3+ Datacentres, close to our users (/clients)"

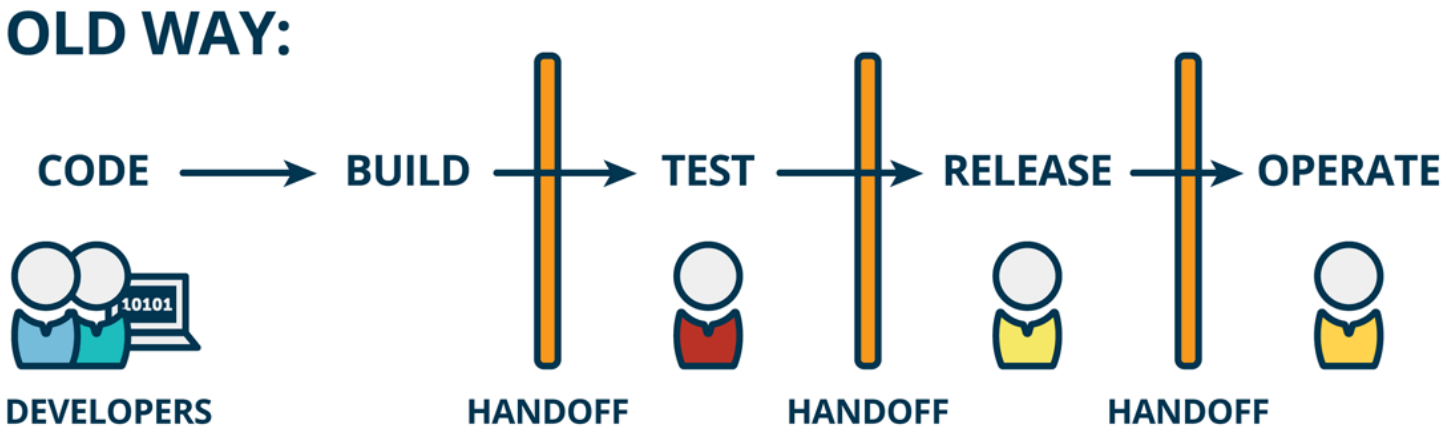
- Infrastructure (IT)
- DataCenters
  - hosting / collocation
  - IP-transit (LIR)
- Private-Cloud
  - OpenStack + Vmware
  - Baremetal / DataScience
- Services
  - DevOps tooling

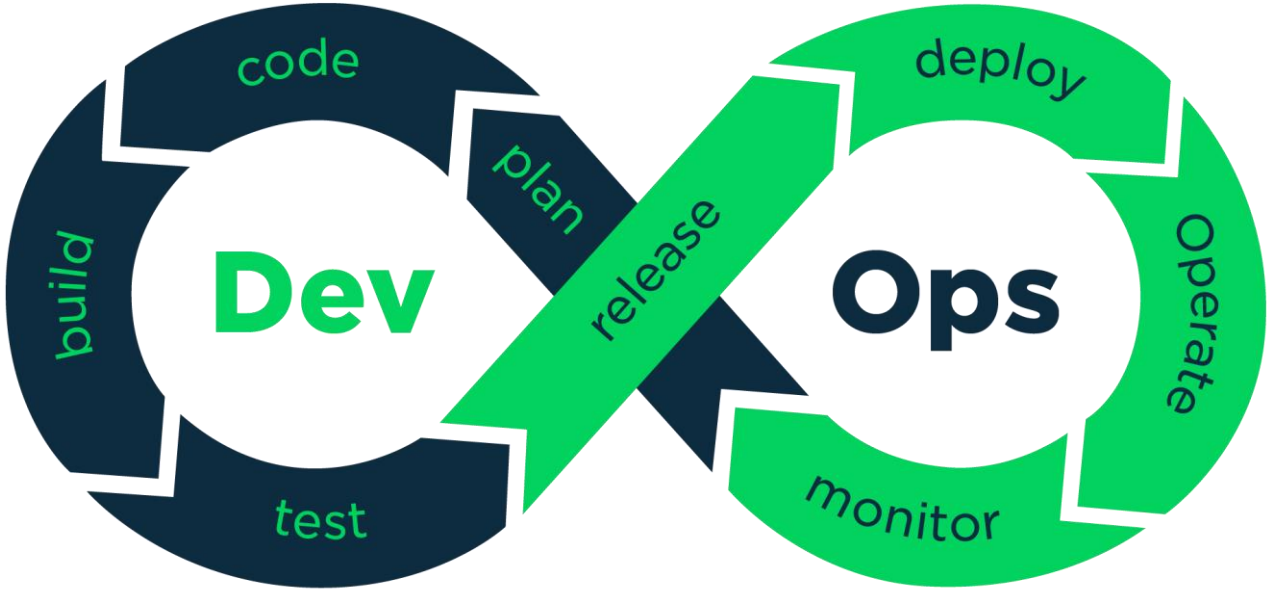
+ Cross functional
+ Everybody's hands-on ! / no golden-fingers
+ Generalize as-code for everything
+ Commit often, perfect later, push once

Team mojo – “Do it as-code or don’t do it”

- We automate everything (as in everything as possible), using:
  - Config-mgmt tools (Ansible, SaltStack)
  - APIs (custom tools if doesn’t exist)
  - DNSes, RIPE object, IaaS, …
- It’s a strong criteria when selecting hardware or software
- Starting to do that also to lower operational tasks
- We of course also apply this principle for network appliances

Refresher: Traditional model vs DevOps





+ Follow software dev best practices
+ Try to apply the same practices for infra
+ Never ending story

Techno stack

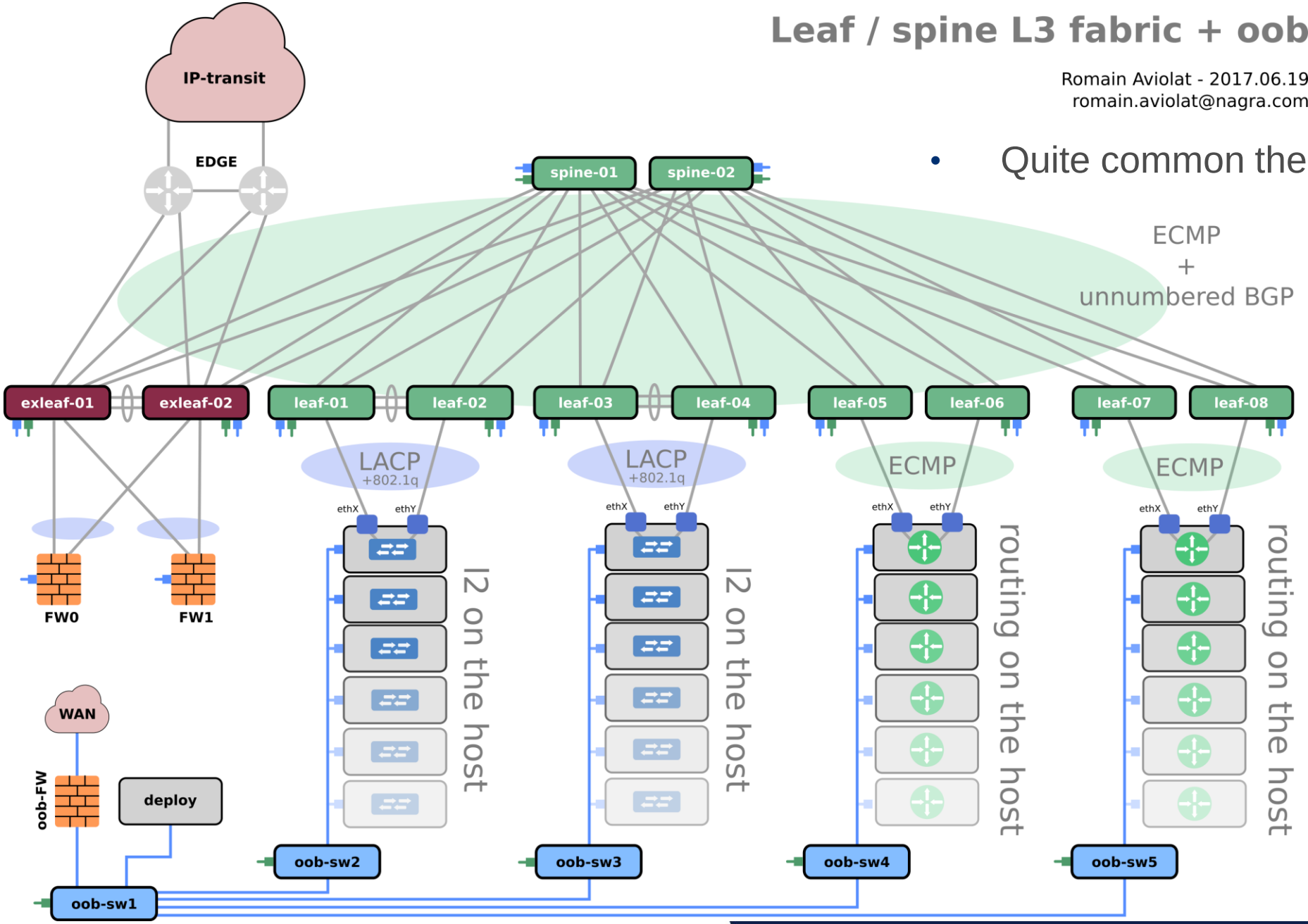
- IaaS: OpenStack
- Storage (+object): CEPH
- VMware cloud: NSX / vRA
- Baremetal deployment: MaaS (Canonical)
- Network: Whitebox + cumuluslinux
- Containers: K8s, docker
- IaaC: Ansible, Saltstack
- CI/CD + versioning: GitLab
- Monitoring: Prometheus, Grafana, Kibana, Grafana

Infrastructure big picture

Leaf / spine L3 fabric + oob

Romain Aviolat - 2017.06.19
romain.aviolat@nagra.com

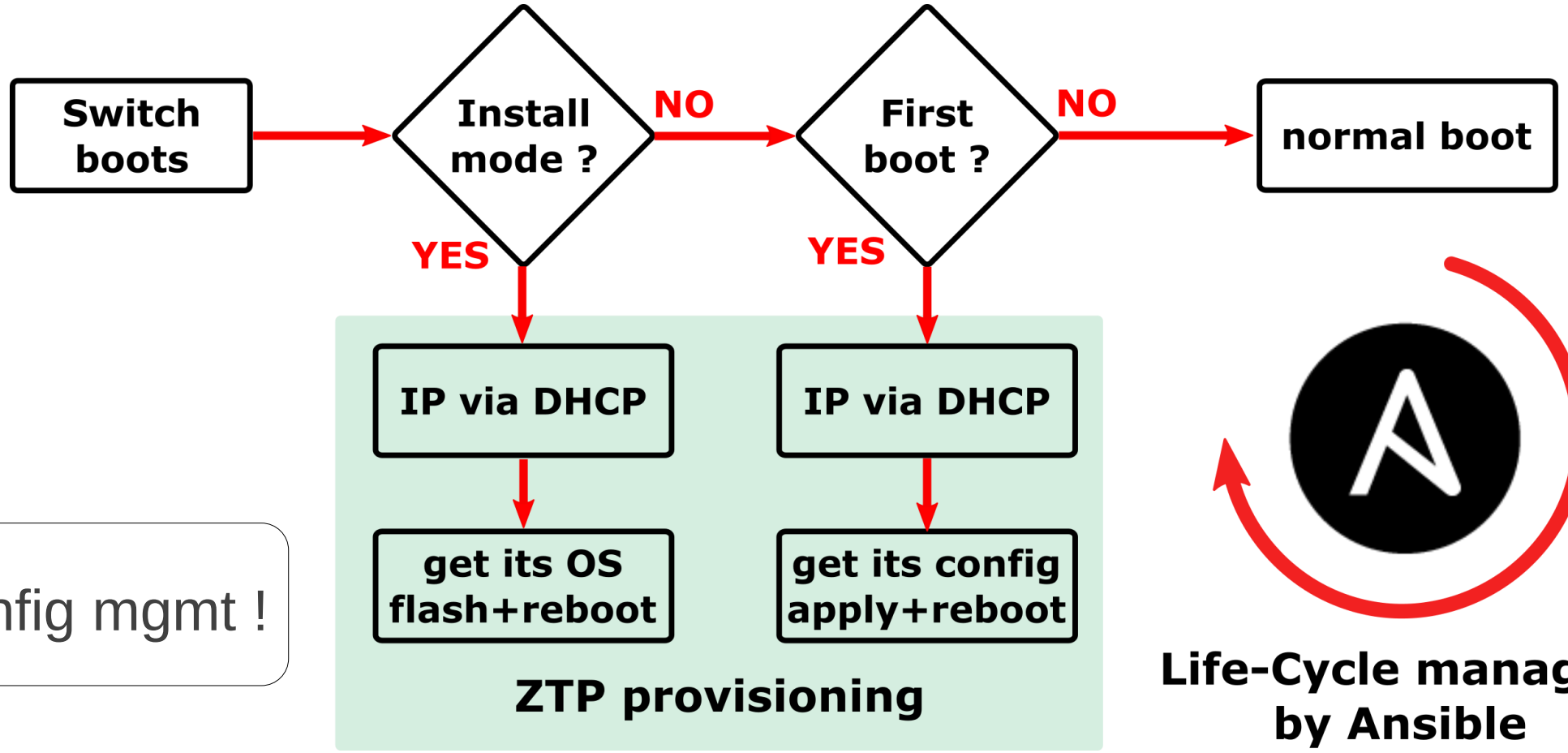
- Quite common these days



Network features

- 1 / 10 / 40GbE
- Layer 2 virtualization using eVPN
- L3 down to the host (Bare-metal infrastructure)
  - 1 IP (v4/v6) per host and per service
  - Anycast v4 + v6 for services HA
  - FRR
- VRFs for data / control plane

Devices life-cycle



- ZTP != config mgmt !

Config management using Ansible

```
interfaces:
  swp1:
    alias: storage-dc1r02n01
    vlans:
    - ipmi
    pvid: 2048
  swp2:
    alias: storage-dc1r02n02
    vlans:
    - ipmi
    pvid: 2048
  swp3:
    alias: storage-dc1r02n03
    vlans:
    - ipmi
    pvid: 2048
  swp4:
    alias: storage-dc1r02n04
    vlans:
    - ipmi
    pvid: 2048
```

```
{% if interfaces is defined %}
{% for port, value in interfaces.items() %}
auto {{ port }}
iface {{ port }} {% if value and 'address'

{% if value and 'mtu' in value %}
    mtu {{ value['mtu'] }}
{% endif %}
{% if value and 'link-speed' in value %}
    link-speed {{ value['link-speed'] }}
{% endif %}
```

```
auto swp2
iface swp2
    bridge-vids 1536
    bridge-pvid 2048
    alias storage-dc1r02n02

auto swp3
iface swp3
    bridge-vids 1536
    bridge-pvid 2048
    alias storage-dc1r02n03

auto swp1
iface swp1
    bridge-vids 1536
    bridge-pvid 2048
    alias storage-dc1r02n01
```

- Routing engine, interfaces, …
- Use variable groups to maintain consistency
- We don’t use custom modules
- Repeatability is keys (really)

Git for versioning

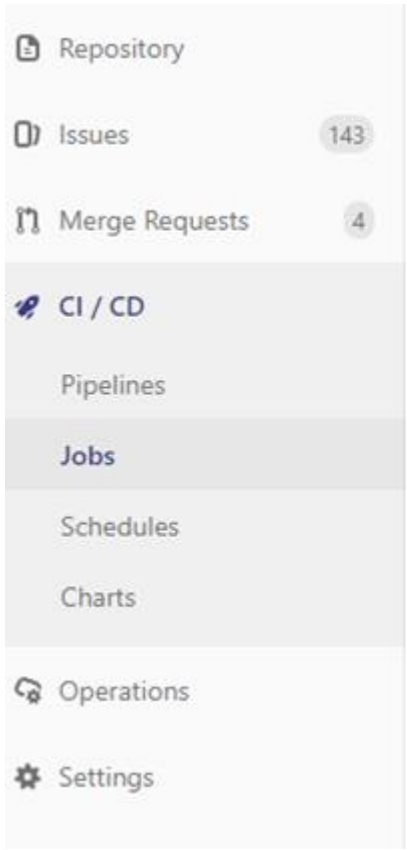
- Collaboration
- Track changes

```
commit bc889ac62847d03f38ad6db26edc47cfd5e584fe (origin/
Author: Aviolat Romain <romain.aviolat@nagra.com>
Date:   Thu Oct 4 08:37:18 2018 +0200

    fabric:: [redacted] add configs for Ipanema

commit 5b2731598a40329e4063cd7a9c5a2d3c09818821
Author: Aviolat Romain <romain.aviolat@nagra.com>
Date:   Thu Oct 4 08:36:40 2018 +0200

    doc: vlan: update to reflect last [redacted] changes
```

```
    swp5:
      link-autoneg: true
      link-speed: 1000
+   swp8:
+     alias: ipanema-lan
+     access: 1538
+     link-speed: 1000
+     link-autoneg: false
    swp35:
      alias: service-dc1r01n01
      l3host: true
```

GitLab / Github / CI/CD platform

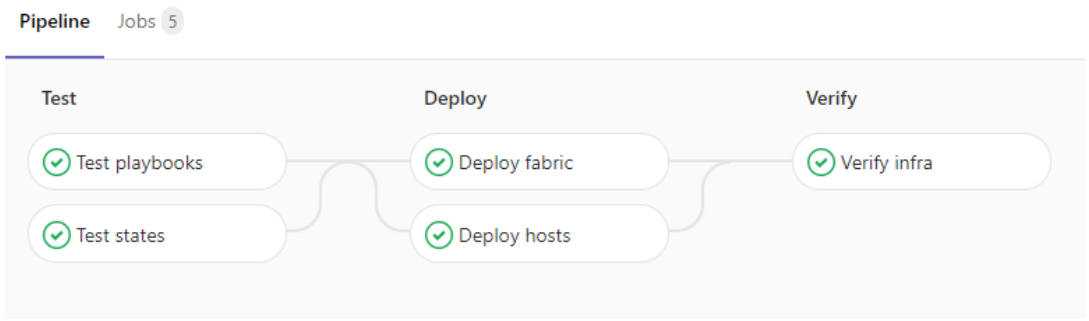
- Don’t run your code locally on your machine
- Describe your work inside issues
- Ansible code / Pipelines are trigerred by commits
- Merge / Pull requests to push in production
- History of all Ansible runs



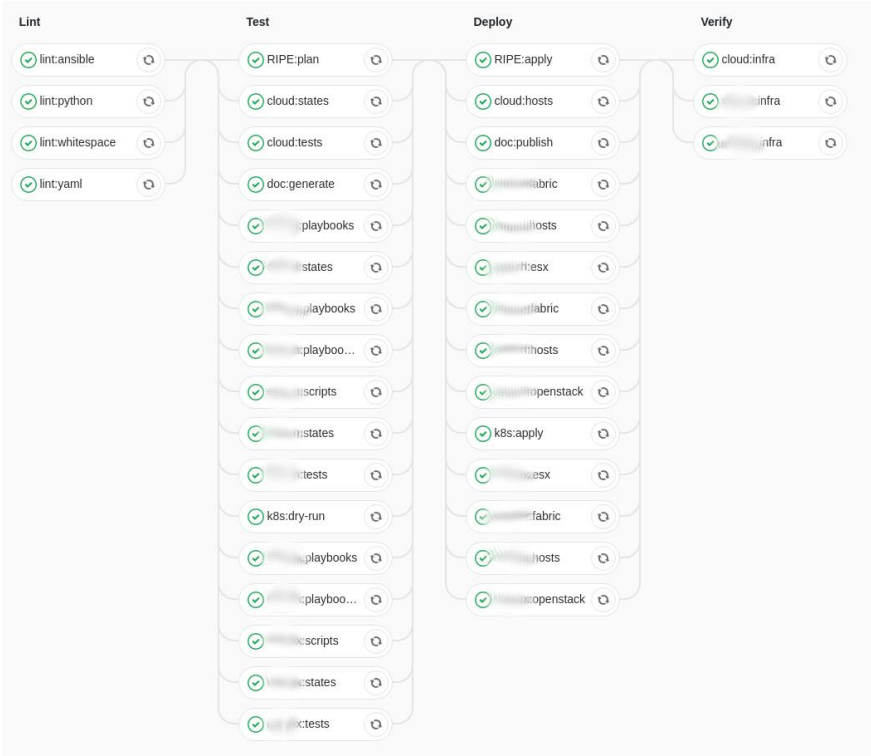

```
TASK [dhcp_server : define static dhcp mappings] ********************************
ok: [adminsw-dc1r01n01 ]
Wednesday 10 October 2018  13:45:06 +0200 (0:00:03.539)       0:03:58.566 *****

TASK [dhcp_server : set dhcp options] ********************************************
ok: [adminsw-dc1r01n01. ]
Wednesday 10 October 2018  13:45:09 +0200 (0:00:03.399)       0:04:01.966 *****

TASK [dhcp_server : configure dnsmasq] *******************************************
ok: [adminsw-dc1r01n01. ]

PLAY RECAP ***********************************************************************
adminsw-dc1r01n01.    : ok=33   changed=0    unreachable=0    failed=0
adminsw-dc1r02n01.    : ok=26   changed=0    unreachable=0    failed=0
adminsw-dc1r03n01.    : ok=26   changed=0    unreachable=0    failed=0
exleaf-dc1r01n01.     : ok=32   changed=1    unreachable=0    failed=0
exleaf-dc1r01n02.     : ok=32   changed=1    unreachable=0    failed=0
leaf-dc1r02n01.       : ok=32   changed=1    unreachable=0    failed=0
leaf-dc1r02n02        : ok=32   changed=1    unreachable=0    failed=0
leaf-dc1r03n01        : ok=32   changed=1    unreachable=0    failed=0
leaf-dc1r03n02.       : ok=32   changed=1    unreachable=0    failed=0
spine-dc1r01n01       : ok=32   changed=1    unreachable=0    failed=0
spine-dc1r01n02.      : ok=32   changed=1    unreachable=0    failed=0
```

Start simple !

2016



2019

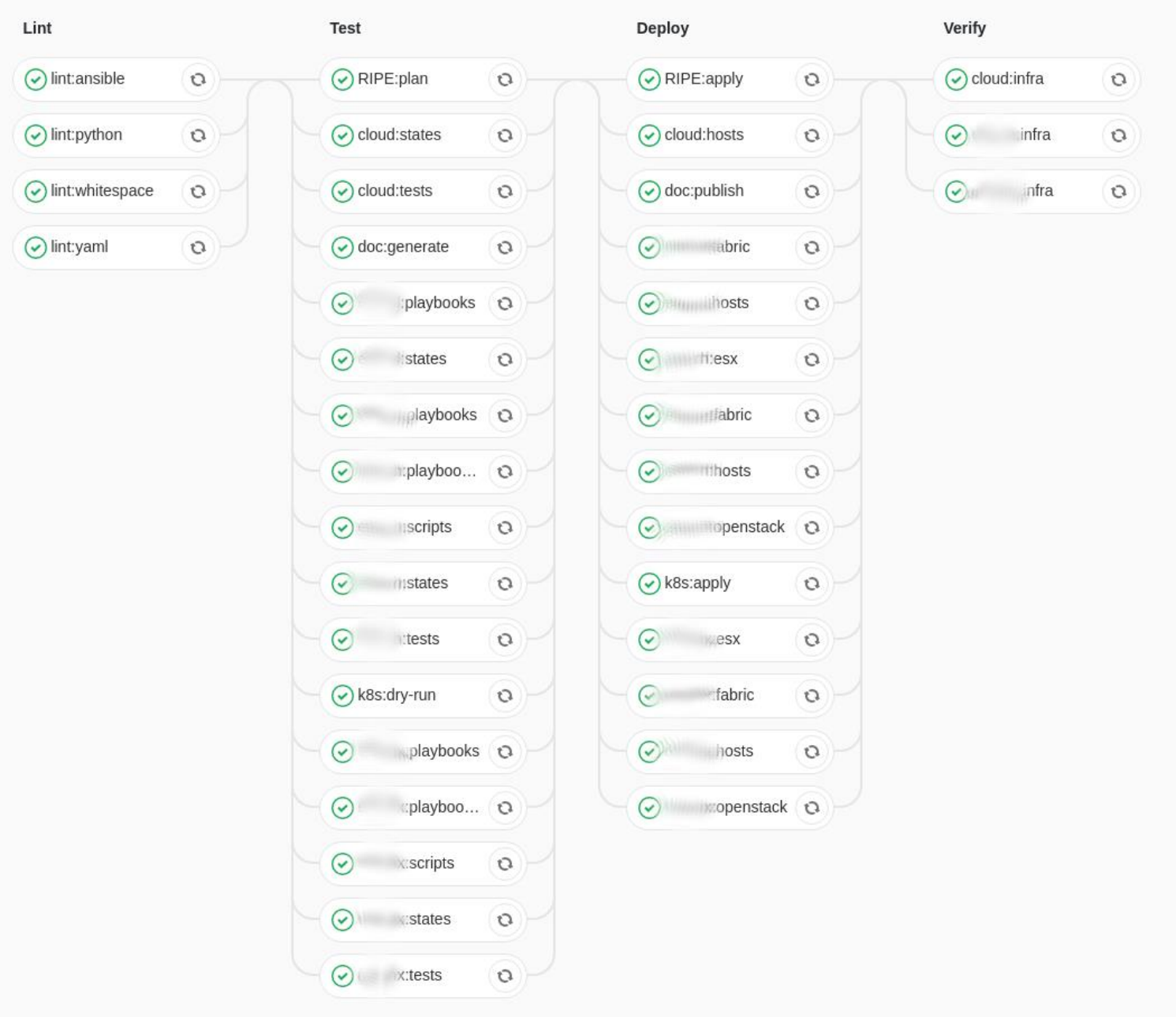


- Start with low-hanging fruits first

GitLab CI/CD

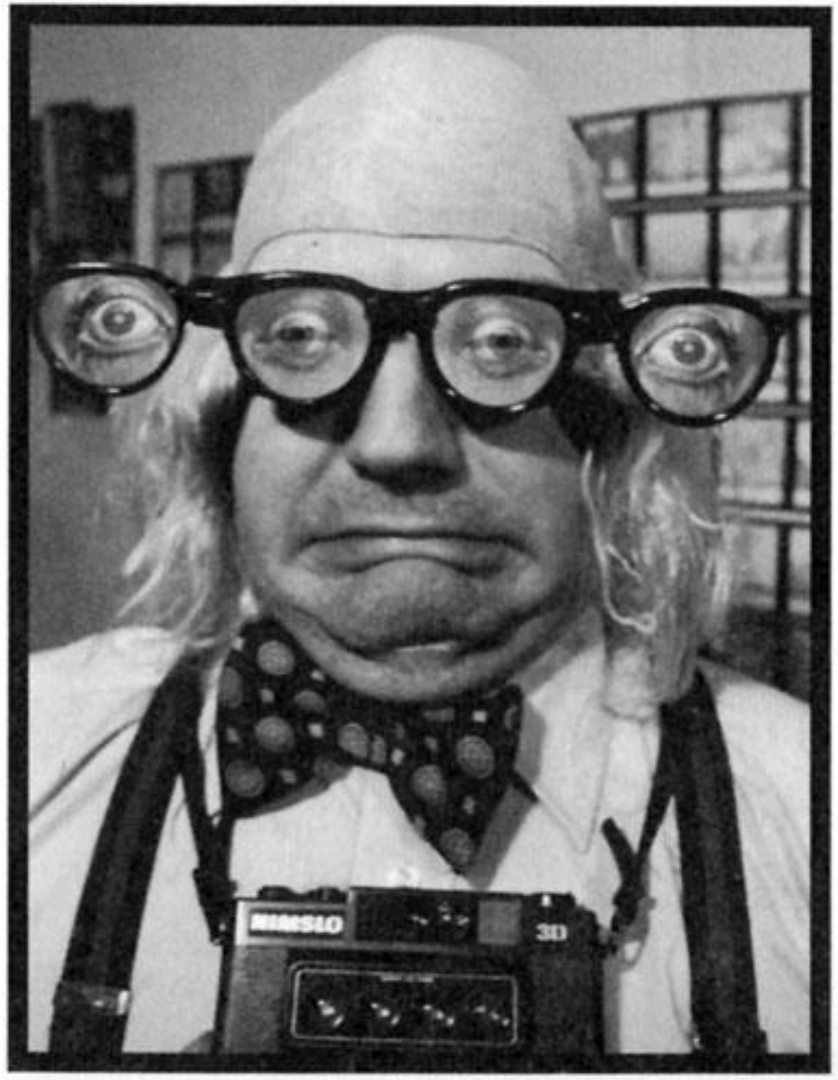
- Multiple stages:
  - Lint (very important)
  - Test
  - Deploy
  - Verify

- Multiple Job:
  - Fabric
  - DNSes
  - Hosts
  - …

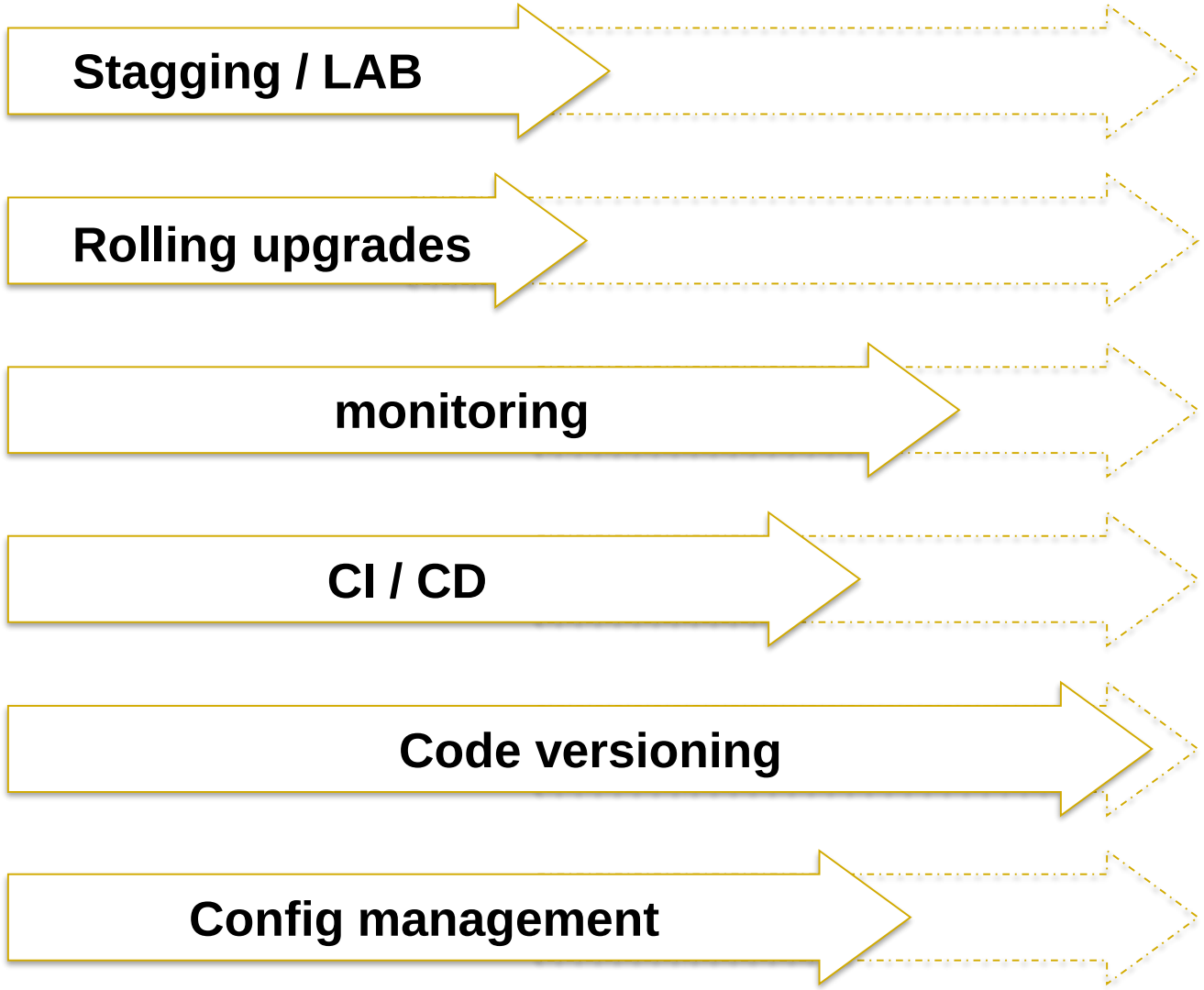


Follow sw-dev. best-practices

- Guys been dong that for years (at scale)
  - Unit tests
- Multiple environments (hard)
  - Dev, Staging, Production
- Code review / Four-eyes review
  - Don’t push in production what you coded
  - Ask someone to do it (he’ll become responsible)
- CI / CD pipeline



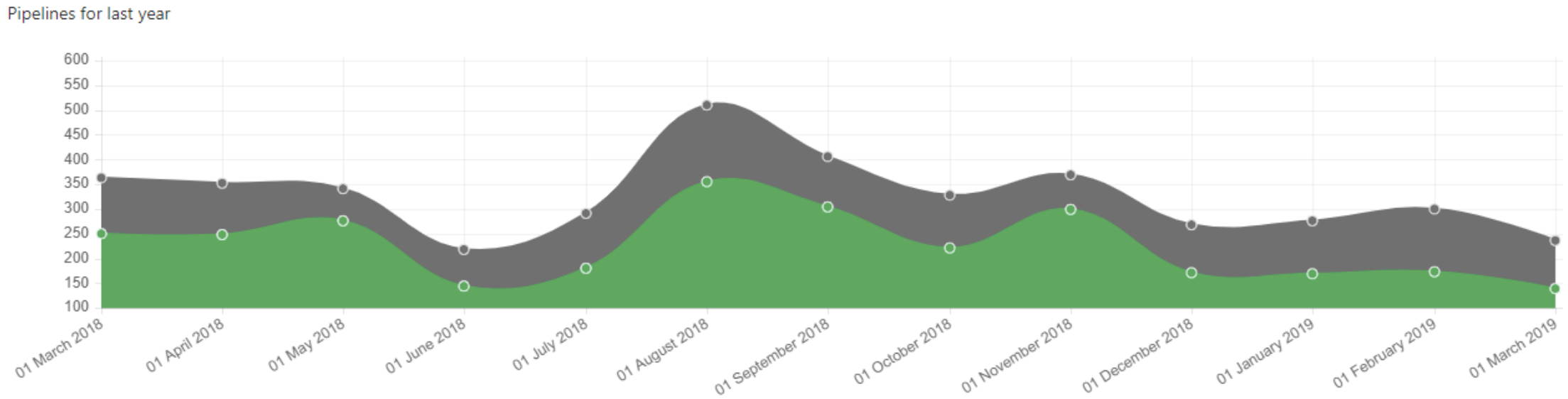
DevOpsness –> never ending story



+ Document technical debt
+ Be honest (backlog)
+ Issue tracking
+ Direct communication

Numbers

- We merged 3000 time in production since August 2016



- Managing 3 DataCenters this way
- ~60 network appliances

Automating operational tasks

- Reducing operational load
- Decrease human errors
  - 20 appliances to upgrade it’s very likely that you’ll do mistakes
- Appliances auto-upgrade

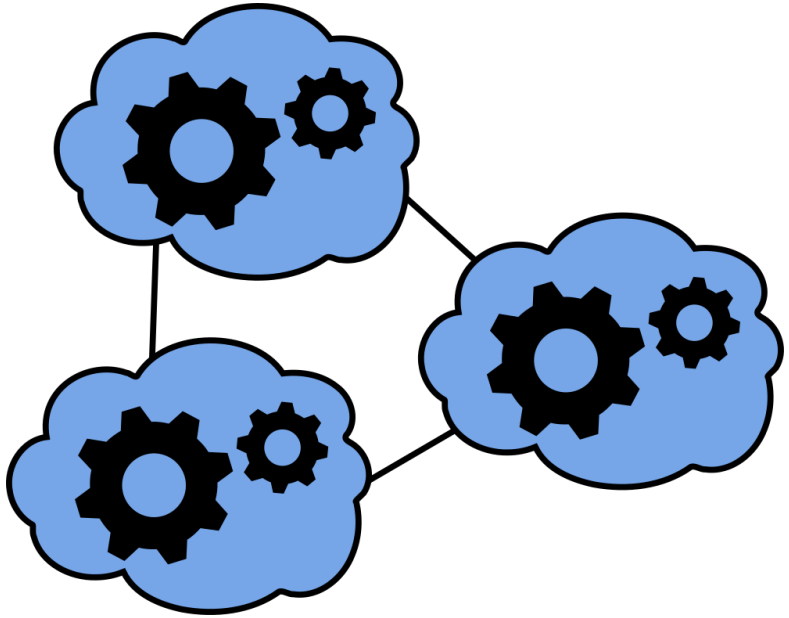
Upgrade my DataCenter ! (scary part)

- Custom set of playbooks / scripts developed internally to upgrade a whole DC
- APT upgrades only for now (no binary upgrades)

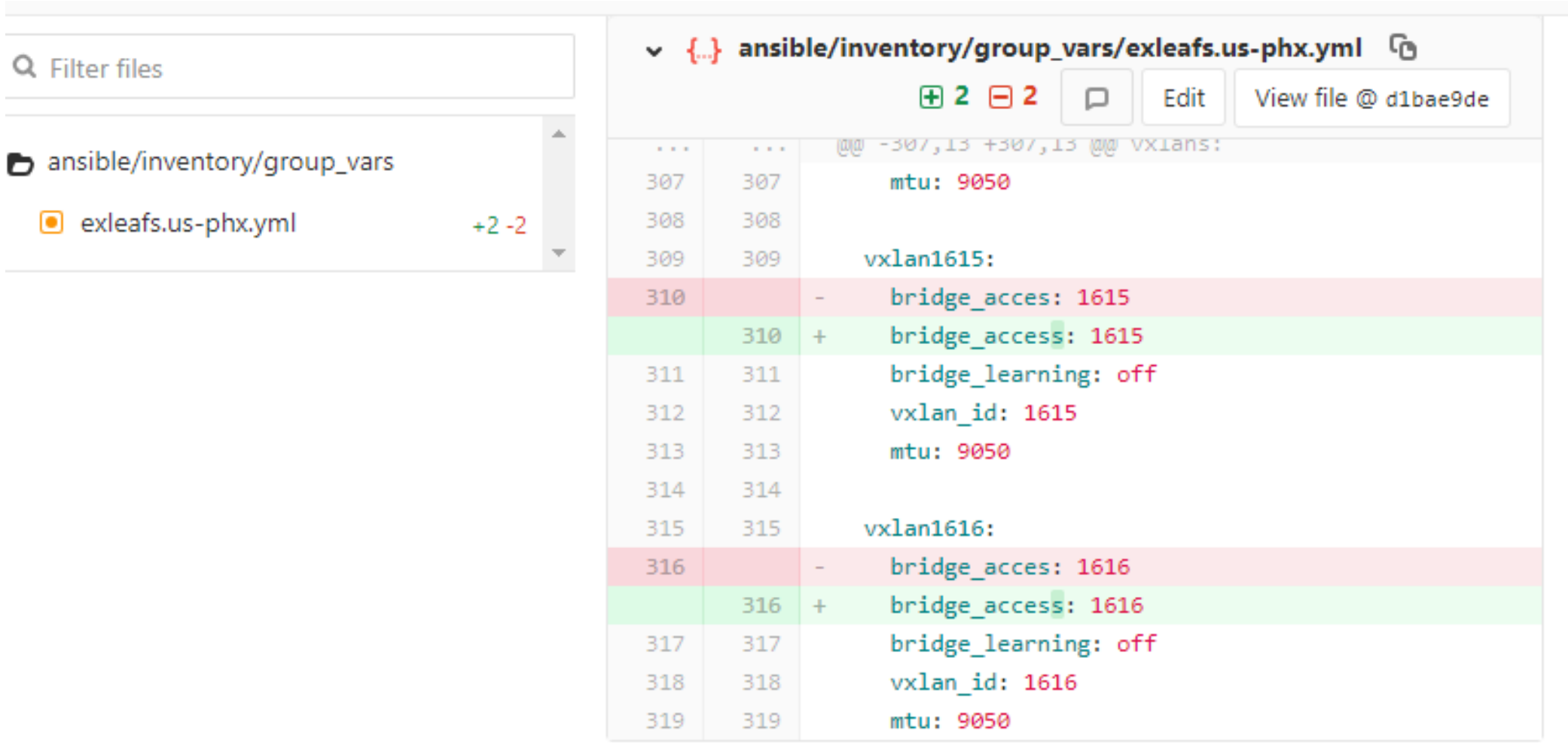
```
- hosts: switches
  user: cumulus
  serial: 1

  tasks:

  #########
  #### check if the switch needs to be upgraded or not

  - name: Register the OS version
    shell: grep "VERSION_ID=" /etc/os-release | cut -d "=" -f 2
    register: current_version
    tags: register

  - name: Check if switch needs to be upgraded or not
    block:
      - debug:
          msg: "Switch needs to be upgraded from {{ current_version.stdout }}

      #########
      #### pre-flight checks applied to all the fabric switches

      # Extecute the consistency checks on all the switches
      - include_tasks: fabric_consistency.yml

      - pause:
          prompt: "Make sure that the peer has become master and hit enter"
```

Staging / LABs

- Complicated to achieve with physical hardware
- With virtualization it’s now “easy” to simulate a complete network environment
- KVM, Virtualbox / Vagrant
- Some vendors directly provide a VM for their OS



As-code != ultimate perfection



Challenges

- Optimize the Ansible code to make it fast (but not too fast…)
- i40e Linux drivers
  - broken on Ubuntu 14.04, had to hack it a bit
  - Post-spectre memory leak
- Non x86/amd64 platform (oob)
  - Binaries (in)compatibility