
Transitioning from 'write mem' to 'git commit'

....and a story of failed automation

A Story of Failed Automation

- Turnaround Time - Aviation term for how long it takes to get a plane unloaded then loaded and prepared for flight
- With growing competition turnaround time is critical to both airlines and airports success
- A key component of turnaround is getting the baggage routed to the proper destination quickly and accurately

A Story of Failed Automation

- With the construction of the new airport under way, in 1990 Denver International Airport (DIA) began exploring an automated baggage system
- The leader in baggage systems was consulted to investigate the feasibility of such a system
- The report came back that such a system was not feasible

Based on the industry ~~report~~ ~~that it was not feasible~~ ~~to build~~ ~~an automated baggage system~~ ~~at that time~~ ~~the project~~

The City of Denver contracted them to build the first automated baggage system!

A Story of Failed Automation

- The DIA automated baggage project suffered from numerous delays and setbacks
- Scope was constantly being redefined as new stakeholders were identified
- The opening of DIA was delayed by 16 months by the baggage system, this cost the City of Denver roughly \$1.1M per day

System at a glance:

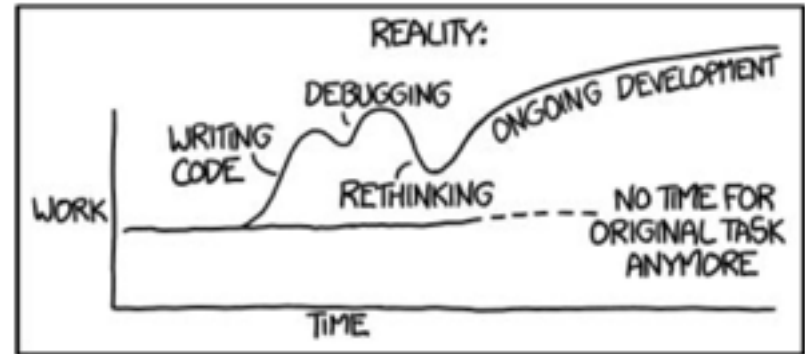
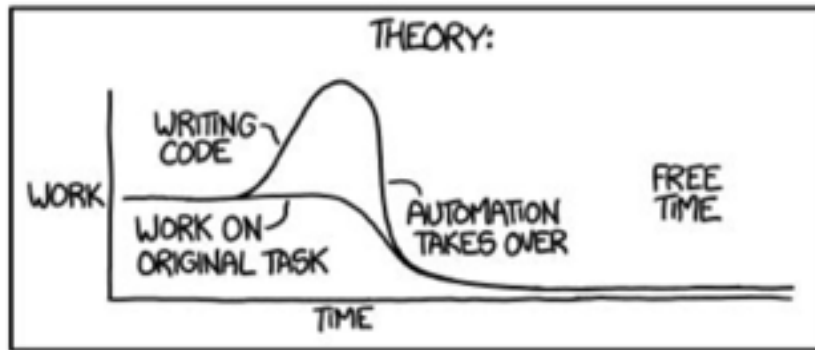
1. 88 airport gates in 3 concourses
2. 17 miles of track and 5 miles of conveyor belts
3. 3,100 standard carts + 450 oversized carts
4. 14 million feet of wiring
5. Network of more than 100 PC's to control flow of carts
6. 5,000 electric motors
7. 2,700 photo cells, 400 radio receivers and 59 laser arrays



A Story of Failed Automation

- The automated baggage system failed - the final product only served outbound luggage for a single airline in a single concourse
- In 2005, the remaining components of the system were removed due to the \$1M per month operating cost exceeding the cost of a conventional system

"I SPEND A LOT OF TIME ON THIS TASK.
I SHOULD WRITE A PROGRAM AUTOMATING IT!"



How does this affect transitioning to 'git commit'?

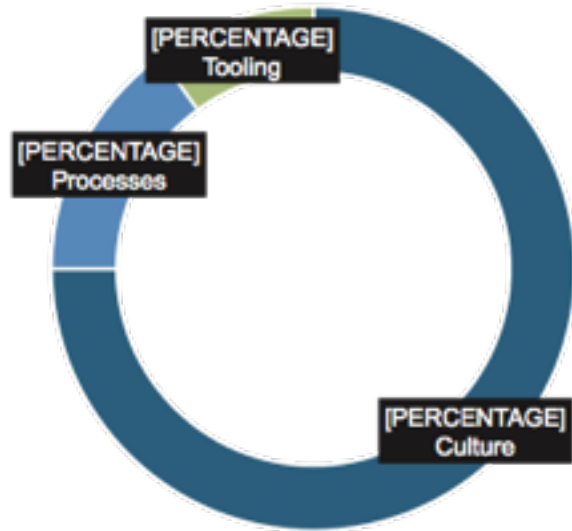
- Automation can be very powerful - when implemented properly
- This is also more than just a technological change
- Like most things, greater agility does come with a cost
- Remember that this is also a huge cultural change that requires buy in from everyone – top down...and bottom up
- So how do we get there?



What is DevOps?

DevOps isn't primarily about toolchains or processes - it's more about culture than anything

Remove silos. Everyone works together towards the greater good



The Right Tools for the Job



- **Revision Control** for backups, auditing, peer review, and access-control



- **Continuous Integration (CI)** and **Continuous Deployment (CD)** for automated testing and workflow



- **Configuration Management** tools for continuous and consistent auditing

Git – It's not just for developers!



- Git is the most popular version control system
- Similar to SVN (Subversion) or CVS but more modern and distributed architecture
- Git keeps your source code/configurations in a repository (repo) and maintains a record of what changed and by whom
- *Note - GitHub *could* be used but for security and compliance it is highly recommended to setup a local/on-prem Git repository for this type of use case

Jenkins – Your new best friend



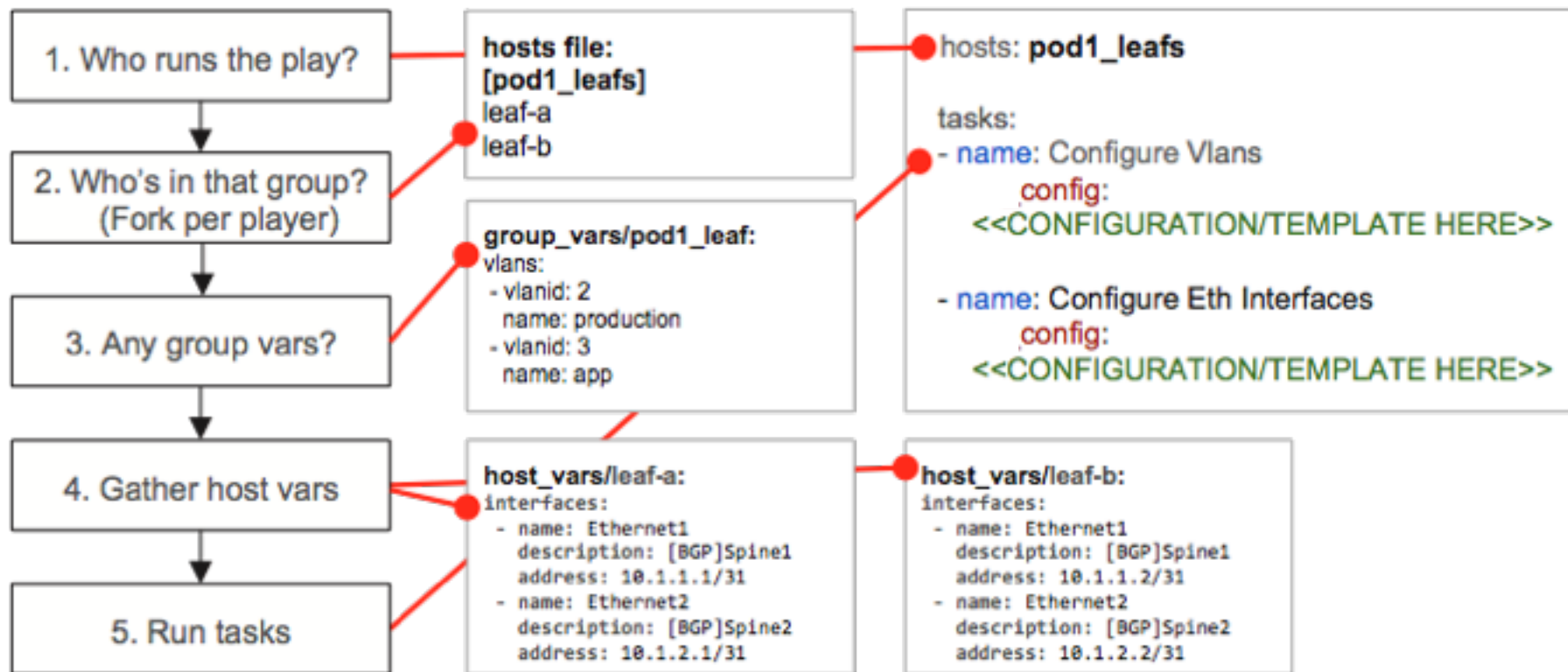
- Jenkins is one of the most popular open source automation tools
- Allows you to create pipelines for building, testing, and deploying software (or configuration) – AKA Continuous Integration/Continuous Delivery (CI/CD)
- It has hundreds of plugins for building/deploying/automating pretty much anything.
 - Git
 - Ansible
 - Docker
 - AWS
 - Many more: <https://plugins.jenkins.io/>

Ansible – Somebody has to do it



- Low barrier for entry
- Uses SSH or API as transport
- Not just for network devices – servers, cloud providers, VMware, whatever
- Python based, so easily extended
- YAML driven, making it extremely easy
- Works with JSON
- 1200+ built-in modules including:
 - apt, yum, copy, command, cron, dns, docker, easy_install, ec2 (amazon modules), file, filesystem, find, git, known_hosts, mysql, mongodb, nagios, npm, openstack, rax (rackspace), pip, shell, snmp_facts...

Ansible 101 - Running a Playbook



Ansible + Jinja = Awesomeness

- Writing Ansible playbooks that issue command after command to configure all 48+ interfaces on a switch is not what most people would call fun
- Jinja is a Python based templating engine that works with Ansible out of the box.
- Jinja templates take variables from Ansible and then output text. To make an analogy, it's a mail merge for configuration.
- Jinja is also present in other parts of Ansible, such as filters and tests

```
{%for intf in interfaces%}  
interface {{ intf.name }}  
    description {{ intf.description }}  
{%endfor%}
```

Truly Making the Transition

Step 1: Define a group variable `leaves` with all of our leaf switches in

`~/lab/hosts`

```
[leaves]
192.168.0.14
192.168.0.15
192.168.0.16
192.168.0.17
```

Step 2: Setup the playbook, in this example we're using a predefined role from Ansible Galaxy and saving it as `~/lab/vlan.yml`

```
- hosts: leaves
  connection: local
  roles:
    - arista.eos-bridging
```

Arista: <https://galaxy.ansible.com/arista/>

Cisco: <https://galaxy.ansible.com/ios-xr/> -
Role: iosxr-ansible

Juniper: <https://galaxy.ansible.com/Juniper>
Role: juniper_junos_config

Truly Making the Transition

Step 3: Setup the group variables for the playbook in Step 2 and save this as `~/lab/group_vars/leafs.yml`

```
provider:
  host: "{{ inventory_hostname }}"
  username: arista
  password: arista
  authorize: yes
  transport: eapi
  validate_certs: no

eos_purge_vlans: true

vlans:
  - vlanid: 1001
    name: default
```

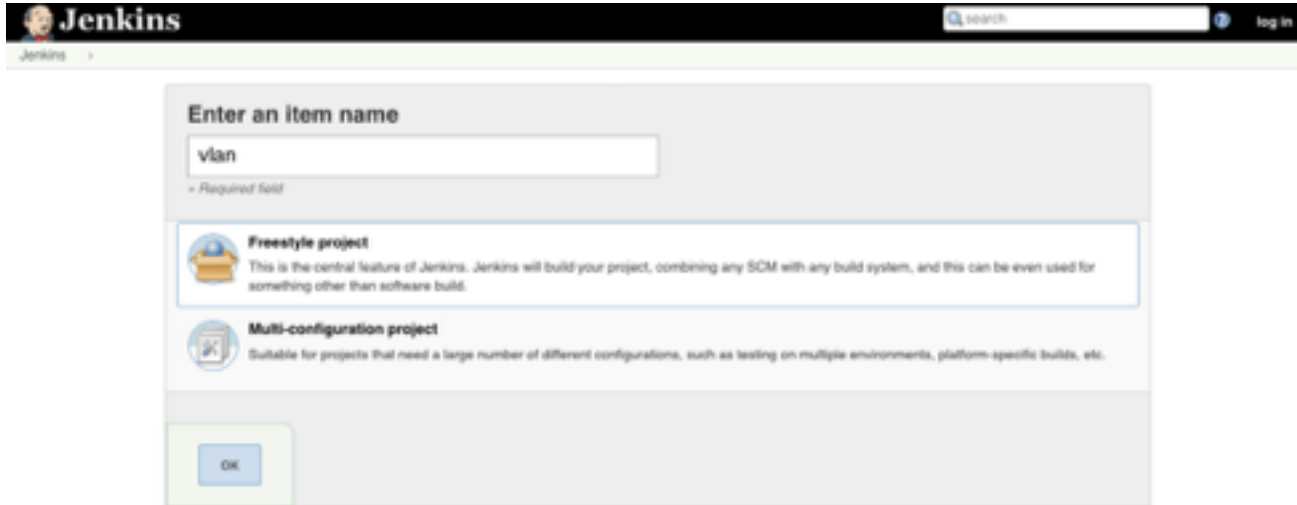

Truly Making the Transition

Step 4: Setup the Git repo and perform the initial commit, this example is using a local repo

```
cd ~/lab
git init
git add .
git commit -m "Initial commit"
git remote add origin ~/lab/repo
git push origin master
```

Truly Making the Transition

Step 5: Login to Jenkins and create a new job and select **Freestyle Project** and name the project `vlan` and click **OK**



The screenshot shows the Jenkins web interface. At the top, there is a black header with the Jenkins logo and name on the left, a search bar in the center, and a 'log in' link on the right. Below the header, the main content area is titled 'Enter an Item name'. It features a text input field containing the text 'vlan'. Below the input field, there is a small red asterisk followed by the text '* Required field'. Below the input field, there are two project type options. The first option is 'Freestyle project', which is highlighted with a blue border. It includes a blue folder icon and the text: 'This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.' The second option is 'Multi-configuration project', which includes a blue folder icon and the text: 'Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.' At the bottom left of the dialog, there is a blue button labeled 'OK'.

Truly Making the Transition

Step 5: Under **Source Code Management** select **Git** and the absolute path to `~/lab` in the **Repository URL** field

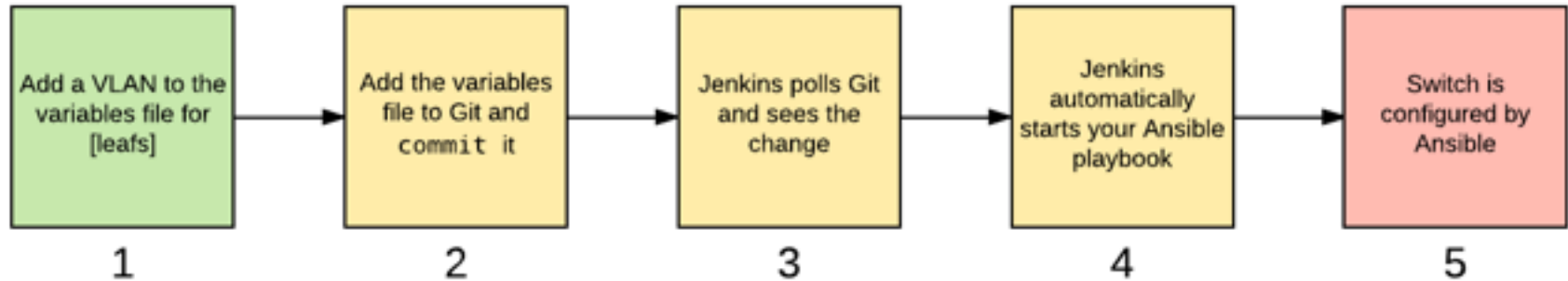
Scroll down to **Build Triggers**, and check **Poll SCM**. Poll SCM will poll for changes in Git and trigger a build from it

In the **Schedule** field enter this to poll every minute: `* * * * *`

Scroll down to **Build** and click on **Add build step** and select **Invoke Ansible Playbook**

For **Playbook Path** enter `vlan.yml` and select **File or Host List** and enter `hosts` and save

A Recap Of What Just Happened



Easy cowboy, where's the approval process?

- Jenkins has a plugin called Pipeline (formally Workflow)
- Pipeline allows you to customize the orchestration flow and add steps such as an approval process
- More info at: <https://plugins.jenkins.io/workflow-aggregator>



Q&A

Contact me at teren@arista.com
Or @terensapp on twitter

Special thank you to Alex Feigenson

ARISTA