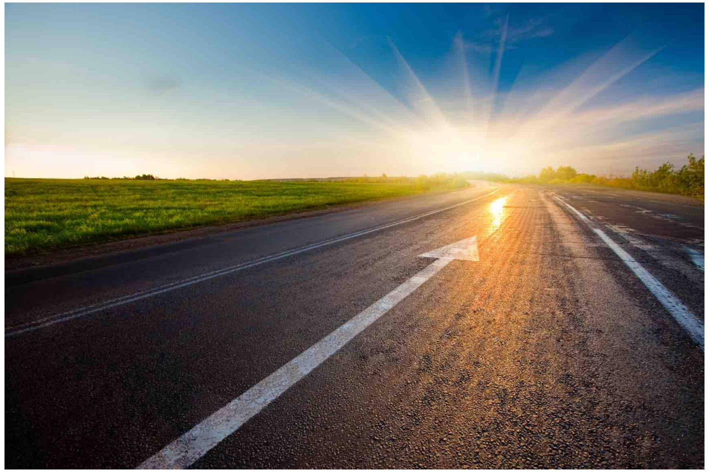
Ansible Tower

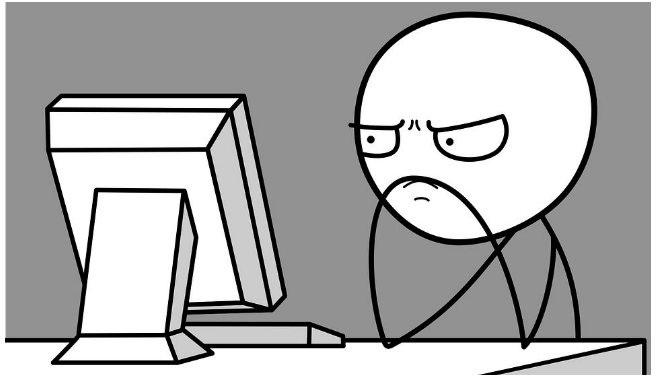
Ansible Journey @ General Mills



- First used Ansible core to automate server patching
- Linux team started using it for more automation tasks
- Network and Enterprise App teams caught on
- We started encouraging other teams to deploy applications using Ansible
  - Separate application from OS config
- Windows web hosting team got involved
- App Dev CoE team...
- Automation team...
- DBA team...

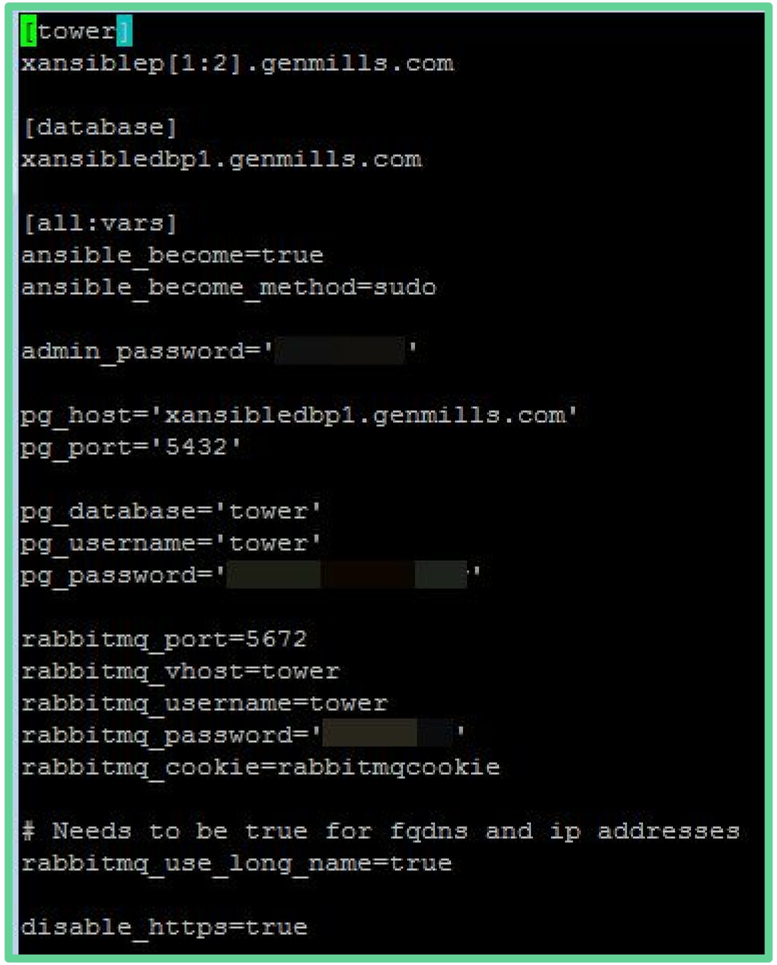
What led to Ansible Tower?

- Ops people spending a lot of time running playbooks for other people
- Cron filling up with ansible jobs
  - No easy way of notifying of failure
- Lack of Linux expertise on Windows side
- Need for integration with other tools (API)
- Want to hide playbook contents while still giving people ability to run them
- Desire for complete inventory of systems
  - Physical and virtual
  - Regularly updating



Tower Installation

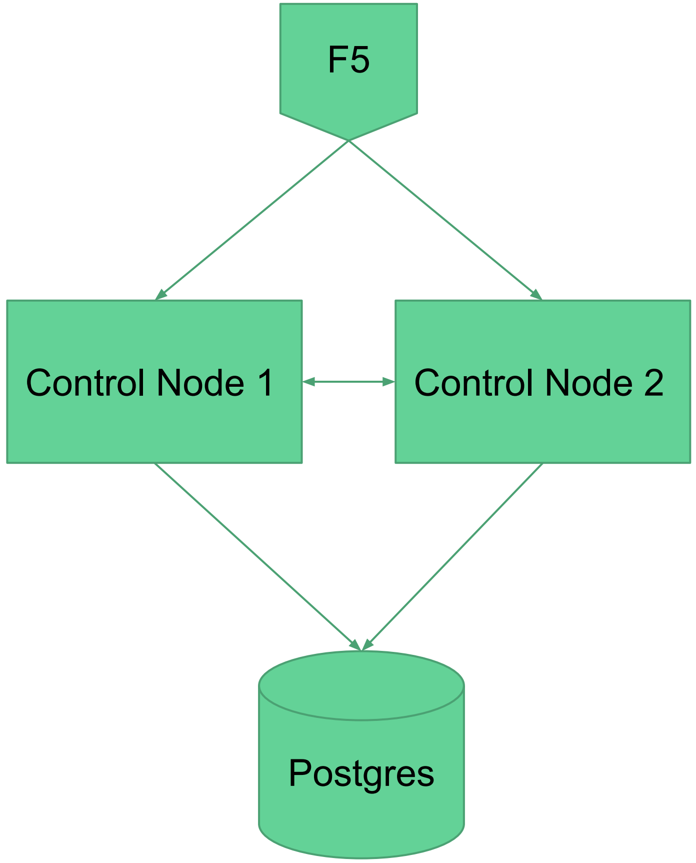
- Download latest tarball
- Installation script that calls playbook
  - Also comes with config/database backup and restore functionality
- Postgres database
- Services
  - RabbitMQ
  - Nginx
  - Supervisord
- Install python dependencies in Ansible virtual environment
  - Separate from Tower virtualenv



```
[tower]
xansiblep[1:2].genmills.com

[database]
xansibledbp1.genmills.com

[all:vars]
ansible_become=true
ansible_become_method=sudo

admin_password='[redacted]'

pg_host='xansibledbp1.genmills.com'
pg_port='5432'

pg_database='tower'
pg_username='tower'
pg_password='[redacted]'

rabbitmq_port=5672
rabbitmq_vhost=tower
rabbitmq_username=tower
rabbitmq_password='[redacted]'
rabbitmq_cookie=rabbitmqcookie

# Needs to be true for fqdns and ip addresses
rabbitmq_use_long_name=true

disable_https=true
```

Our Environment

- Clustered setup
  - Two control nodes
  - External Postgres database server
- Load balancing via F5 across both control nodes
- Nodes are RHEL 7.3 virtual machines
- Each team has own Ansible core server
  - Set up to push to TFS Git repos
- Tower logs exported to Splunk



Tower Demo

- Goal: Provision a new server in Digital Ocean and deploy an Nginx container
- Create Project from GitHub playbook repo
- Create Inventory to use for Digital Ocean servers
- Create three Job Templates
  - Push SSH key and provision new server
  - Add new server to inventory
  - Deploy Docker and Nginx container
- Create Workflow Job Template to chain templates
- Execute workflow via UI

Advice

- Playbook compatibility with Tower
  - Minimize local actions - use delegate_to instead
  - Remember Tower is running as “awx” user
- Don’t turn off job isolation to get Kerberos working
  - Other Tower users can access credential cache
- Write playbooks for Tower control node installation
  - Some configurations are local to nodes in /etc/tower/conf.d
  - Python dependencies for modules
- PyCharm with Git integration is great for editing roles
- Don’t set “Update on Launch” if you want concurrent job templates
- http://docs.ansible.com/ansible-tower/latest/html/userguide/job_templates.html#utilizing-cloud-credentials