Ansible, immutable servers & automated deploys on AWS

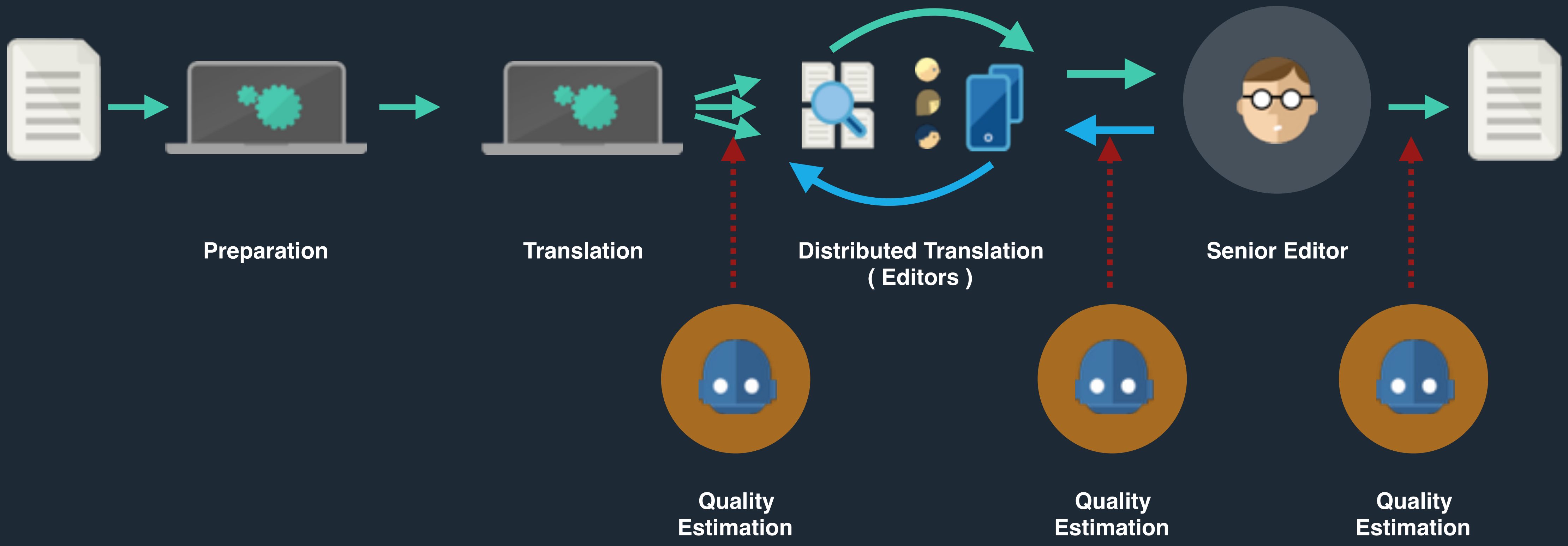
whoami

@vascogpinho

DevOps @ Unbabel

pinho@unbabel.com

Unbabel


Preparation
Translation
Distributed Translation (Editors)
Senior Editor
Quality Estimation
Quality Estimation
Quality Estimation

Unbabel

~40k Users

~40M Translated words

Google

Pinterest

coursera

Microsoft

udemy

Yummly

Old Infrastructure

Heroku + EC2 + compose.io

Deploys

```
git push heroku master
```

ssh + who knows what

Moving to AWS

Ansible

YAML

Agentless

Great at ensuring state

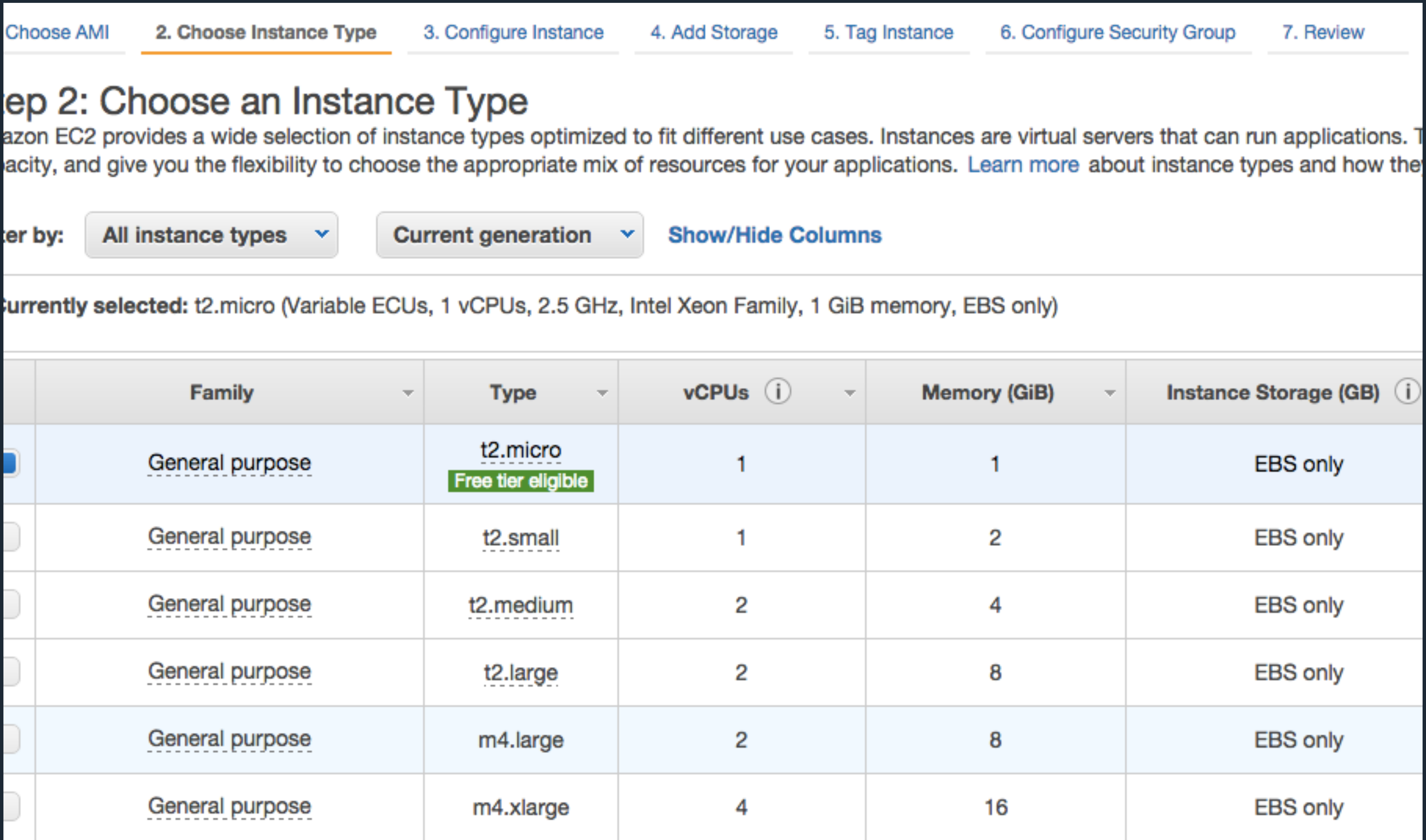
Ansible

Launch & scale infrastructure

Configure base images

Deploy!

Ansible



Choose AMI | **2. Choose Instance Type** | 3. Configure Instance | 4. Add Storage | 5. Tag Instance | 6. Configure Security Group | 7. Review

ep 2: Choose an Instance Type

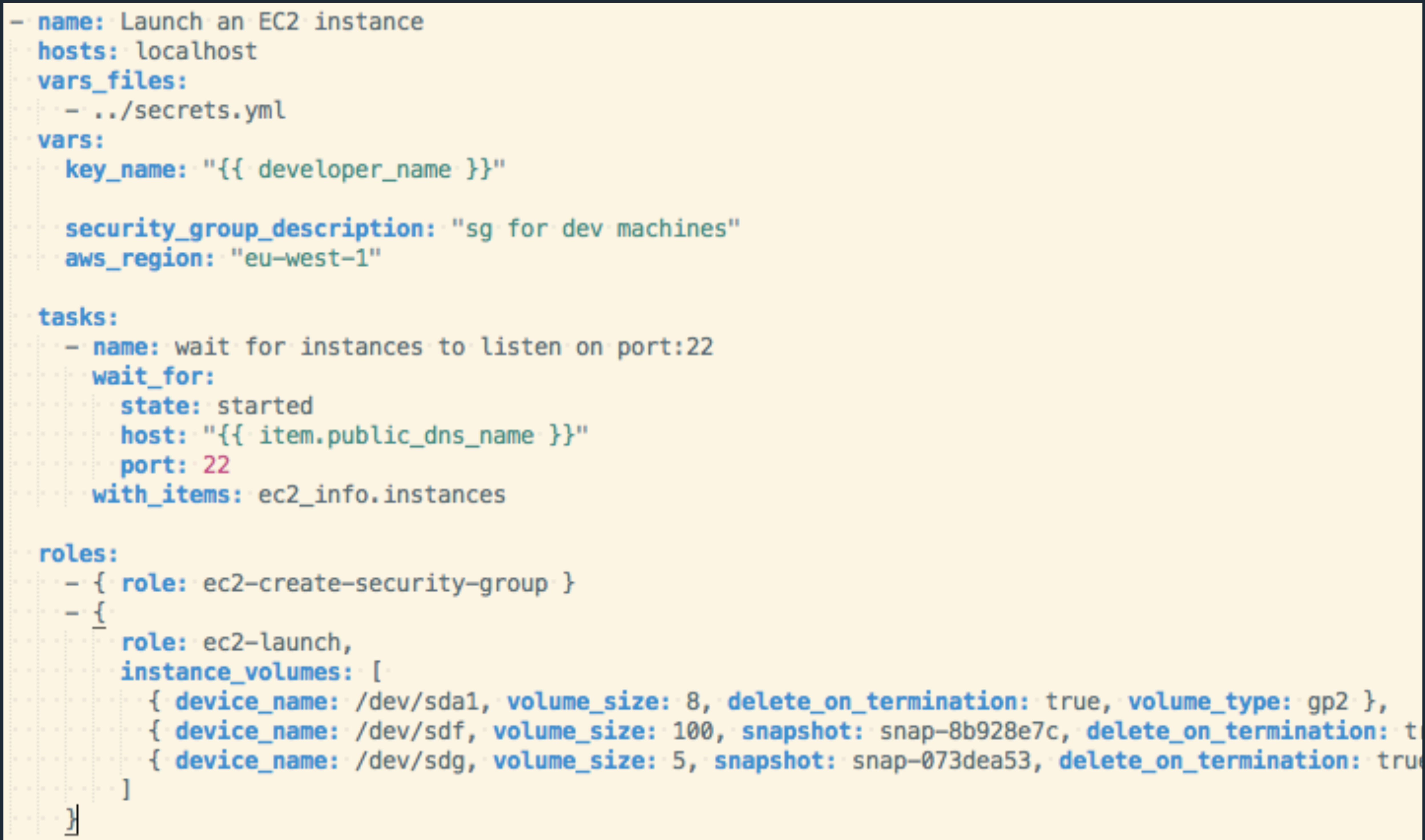
azon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. T
acity, and give you the flexibility to choose the appropriate mix of resources for your applications. Learn more about instance types and how the

er by: All instance types | Current generation | Show/Hide Columns

urrently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)
■	General purpose	t2.micro Free tier eligible	1	1	EBS only
	General purpose	t2.small	1	2	EBS only
	General purpose	t2.medium	2	4	EBS only
	General purpose	t2.large	2	8	EBS only
	General purpose	m4.large	2	8	EBS only
	General purpose	m4.xlarge	4	16	EBS only

Ansible



```
- name: Launch an EC2 instance
  hosts: localhost
  vars_files:
    - ../secrets.yml
  vars:
    key_name: "{{ developer_name }}"

    security_group_description: "sg for dev machines"
    aws_region: "eu-west-1"

  tasks:
    - name: wait for instances to listen on port:22
      wait_for:
        state: started
        host: "{{ item.public_dns_name }}"
        port: 22
      with_items: ec2_info.instances

  roles:
    - { role: ec2-create-security-group }
    - {
        role: ec2-launch,
        instance_volumes: [
          { device_name: /dev/sda1, volume_size: 8, delete_on_termination: true, volume_type: gp2 },
          { device_name: /dev/sdf, volume_size: 100, snapshot: snap-8b928e7c, delete_on_termination: t
          { device_name: /dev/sdg, volume_size: 5, snapshot: snap-073dea53, delete_on_termination: tru
        ]
    }
```

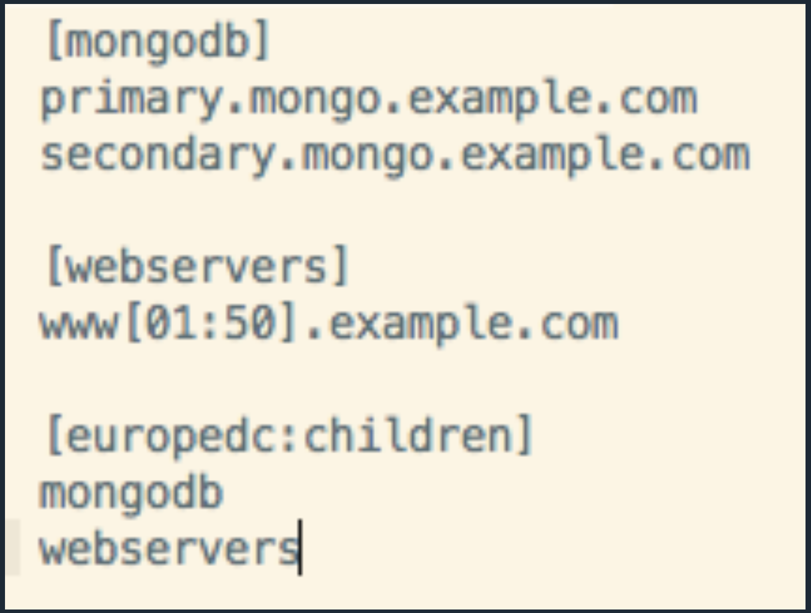
Ansible

AWS, Azure, Digital Ocean, Google Compute Engine, Linode, OpenStack, Rackspace...

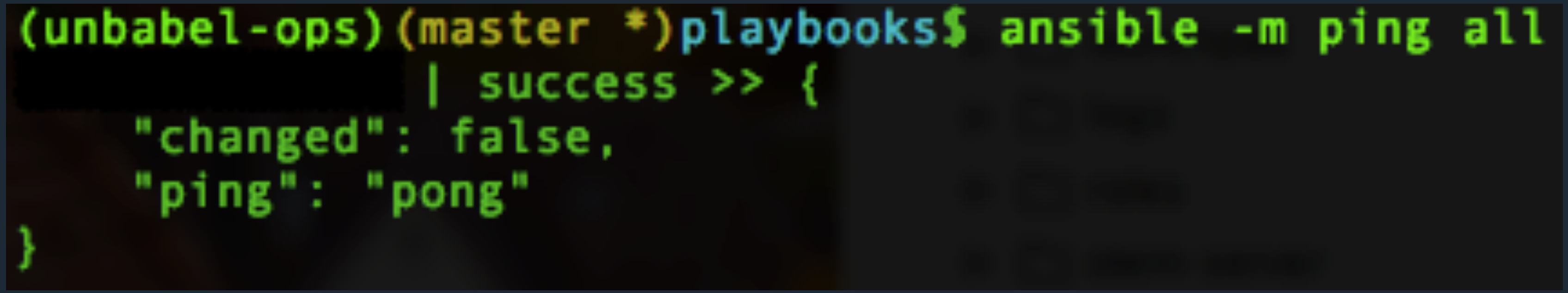
Ansible

Where my servers at?

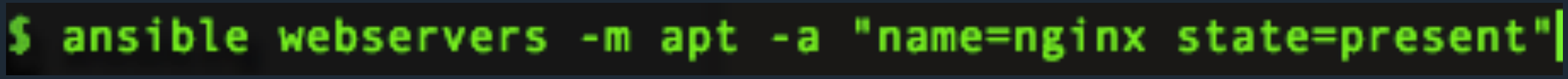
Inventory File



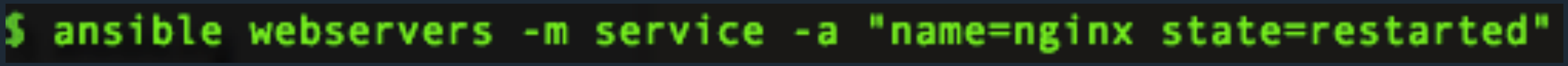
```
[mongodb]
primary.mongo.example.com
secondary.mongo.example.com

[webservers]
www[01:50].example.com

[europedc:children]
mongodb
webservers
```



```
(unbabel-ops)(master *)playbooks$ ansible -m ping all
[redacted] | success >> {
    "changed": false,
    "ping": "pong"
}
```

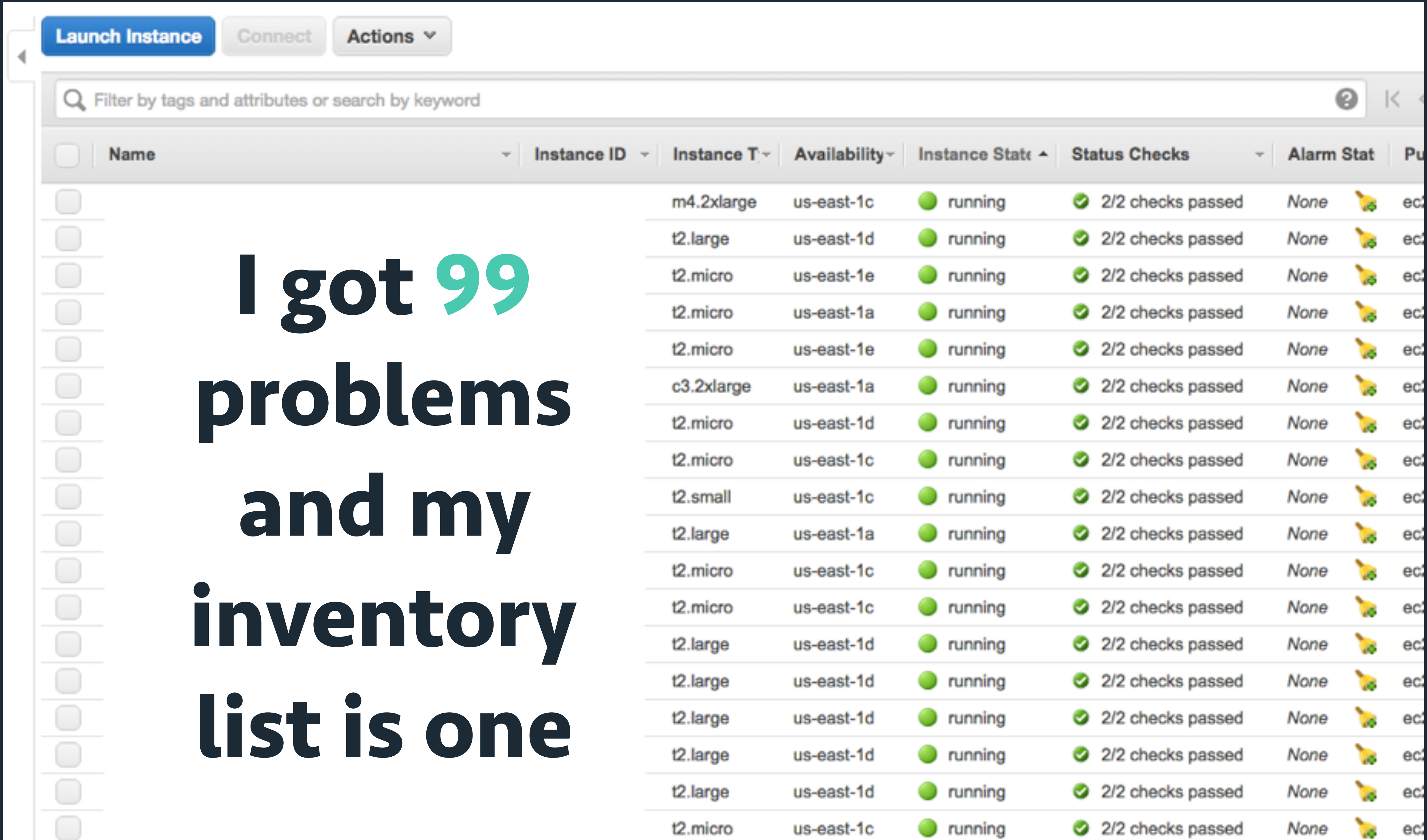


```
$ ansible webservers -m apt -a "name=nginx state=present"
```



```
$ ansible webservers -m service -a "name=nginx state=restarted"
```

Ansible



Launch Instance | Connect | Actions

Filter by tags and attributes or search by keyword

Name	Instance ID	Instance T	Availability	Instance Stat	Status Checks	Alarm Stat	Pu
		m4.2xlarge	us-east-1c	● running	✓ 2/2 checks passed	None	ec
		t2.large	us-east-1d	● running	✓ 2/2 checks passed	None	ec
		t2.micro	us-east-1e	● running	✓ 2/2 checks passed	None	ec
		t2.micro	us-east-1a	● running	✓ 2/2 checks passed	None	ec
		t2.micro	us-east-1e	● running	✓ 2/2 checks passed	None	ec
		c3.2xlarge	us-east-1a	● running	✓ 2/2 checks passed	None	ec
		t2.micro	us-east-1d	● running	✓ 2/2 checks passed	None	ec
		t2.micro	us-east-1c	● running	✓ 2/2 checks passed	None	ec
		t2.small	us-east-1c	● running	✓ 2/2 checks passed	None	ec
		t2.large	us-east-1a	● running	✓ 2/2 checks passed	None	ec
		t2.micro	us-east-1c	● running	✓ 2/2 checks passed	None	ec
		t2.micro	us-east-1c	● running	✓ 2/2 checks passed	None	ec
		t2.large	us-east-1d	● running	✓ 2/2 checks passed	None	ec
		t2.large	us-east-1d	● running	✓ 2/2 checks passed	None	ec
		t2.large	us-east-1d	● running	✓ 2/2 checks passed	None	ec
		t2.large	us-east-1d	● running	✓ 2/2 checks passed	None	ec
		t2.large	us-east-1d	● running	✓ 2/2 checks passed	None	ec
		t2.micro	us-east-1c	● running	✓ 2/2 checks passed	None	ec

I got 99 problems and my inventory list is one

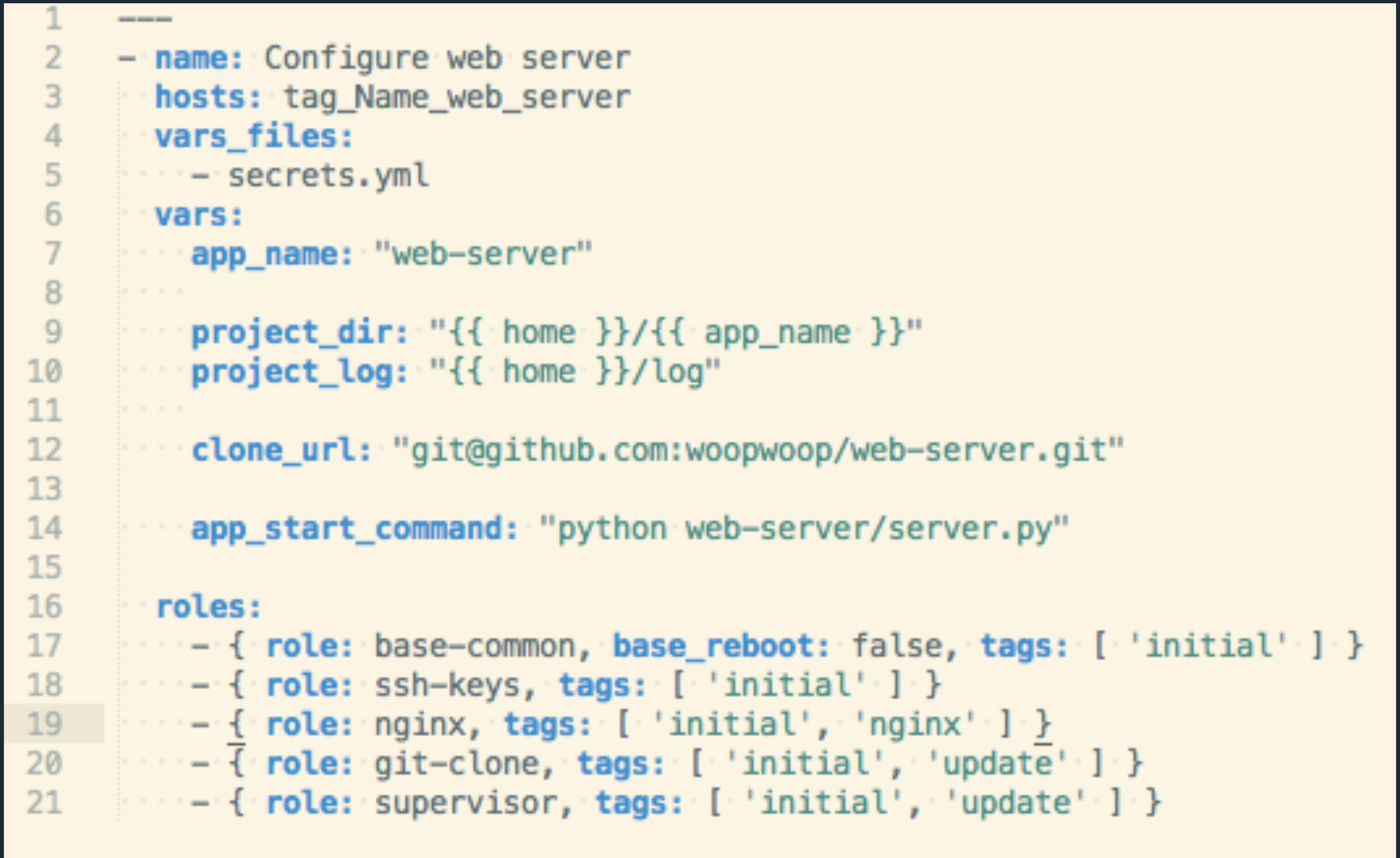
Ansible

Dynamic Inventory

Instance ID, Region, Availability Zone, Security Group, Tag

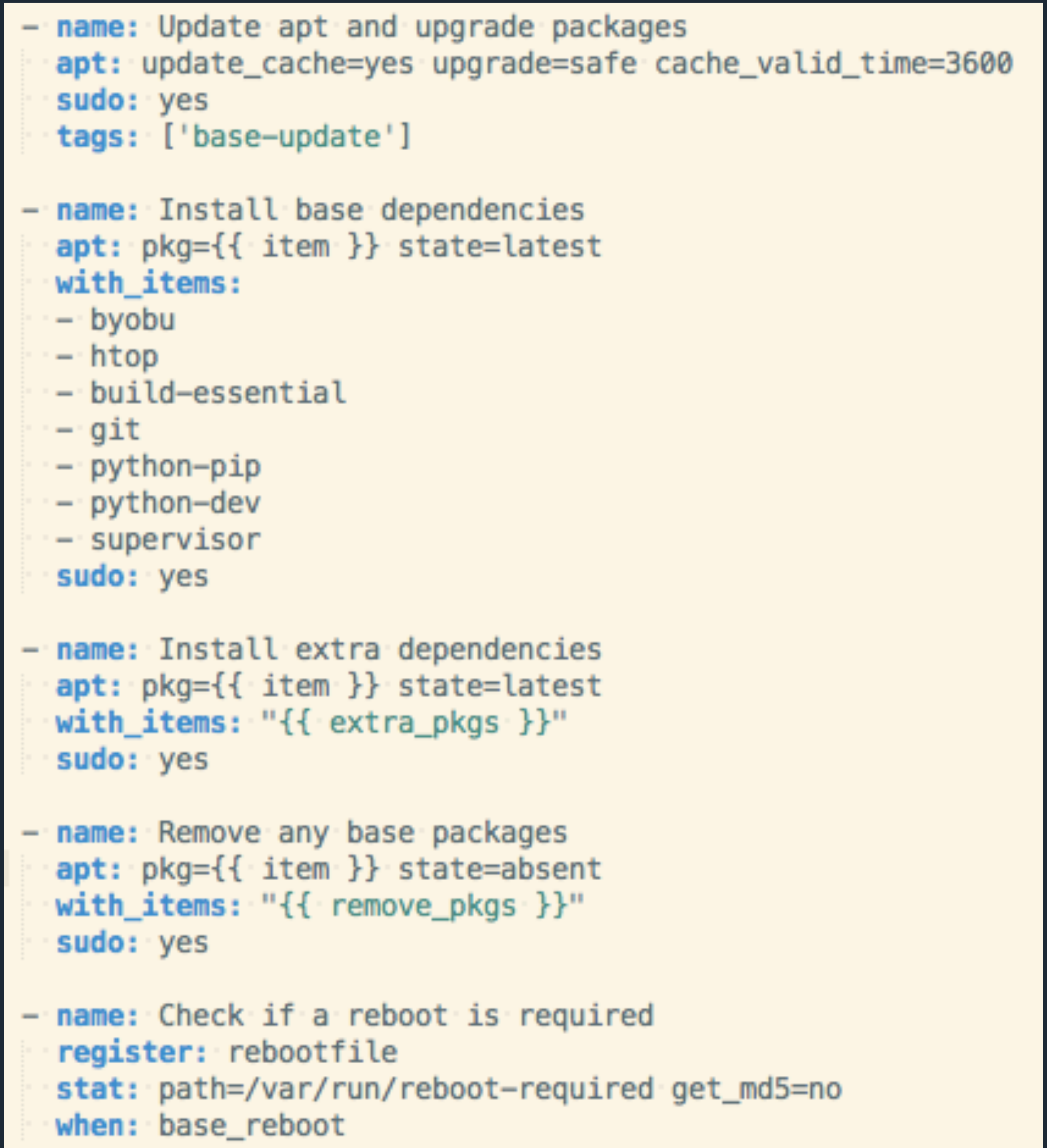
Automatic variables

Ansible



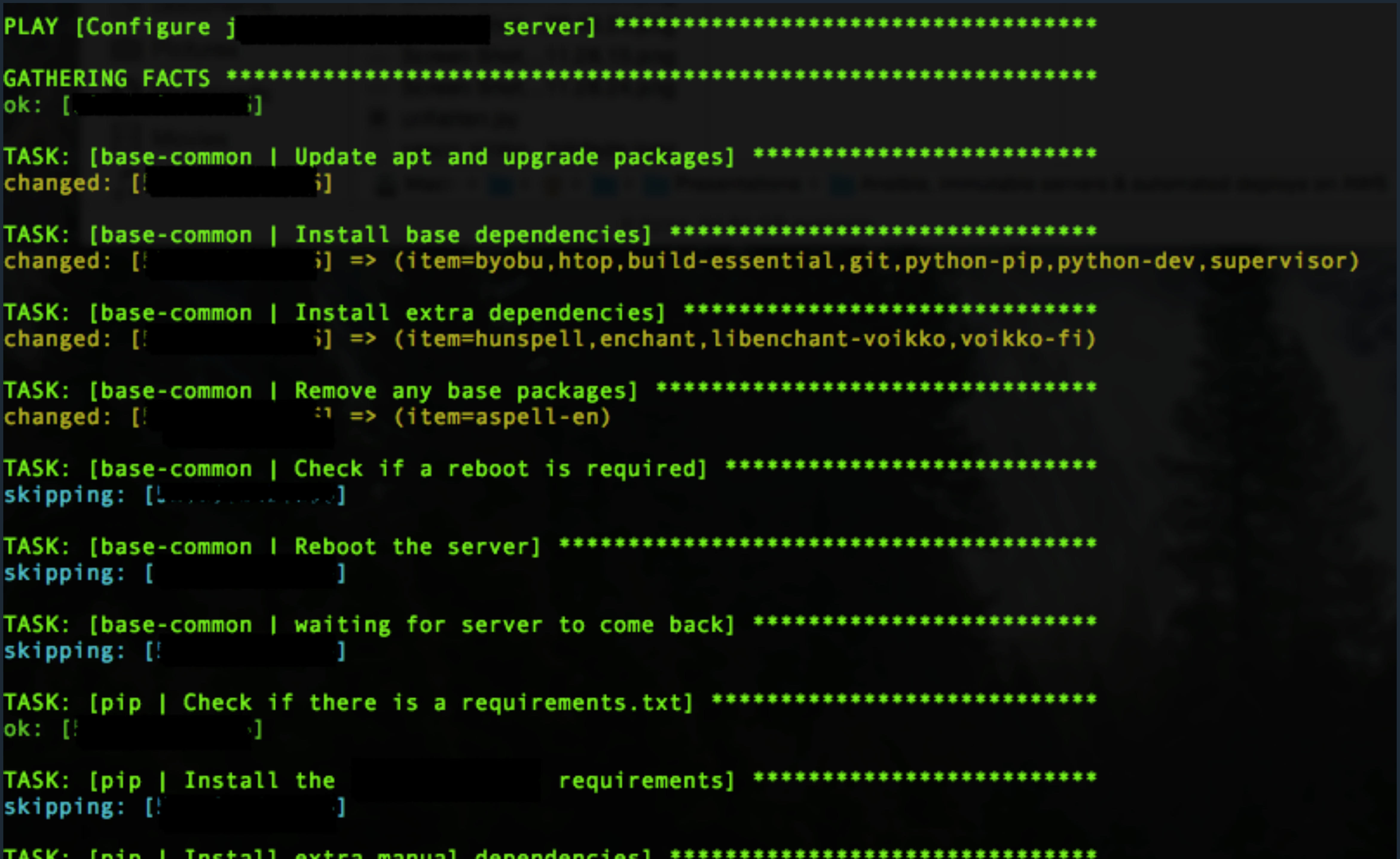
```yaml
---
- name: Configure web server
  hosts: tag_Name_web_server
  vars_files:
    - secrets.yml
  vars:
    app_name: "web-server"

    project_dir: "{{ home }}/{{ app_name }}"
    project_log: "{{ home }}/log"

    clone_url: "git@github.com:woopwoop/web-server.git"

    app_start_command: "python web-server/server.py"

  roles:
    - { role: base-common, base_reboot: false, tags: [ 'initial' ] }
    - { role: ssh-keys, tags: [ 'initial' ] }
    - { role: nginx, tags: [ 'initial', 'nginx' ] }
    - { role: git-clone, tags: [ 'initial', 'update' ] }
    - { role: supervisor, tags: [ 'initial', 'update' ] }
```

Ansible



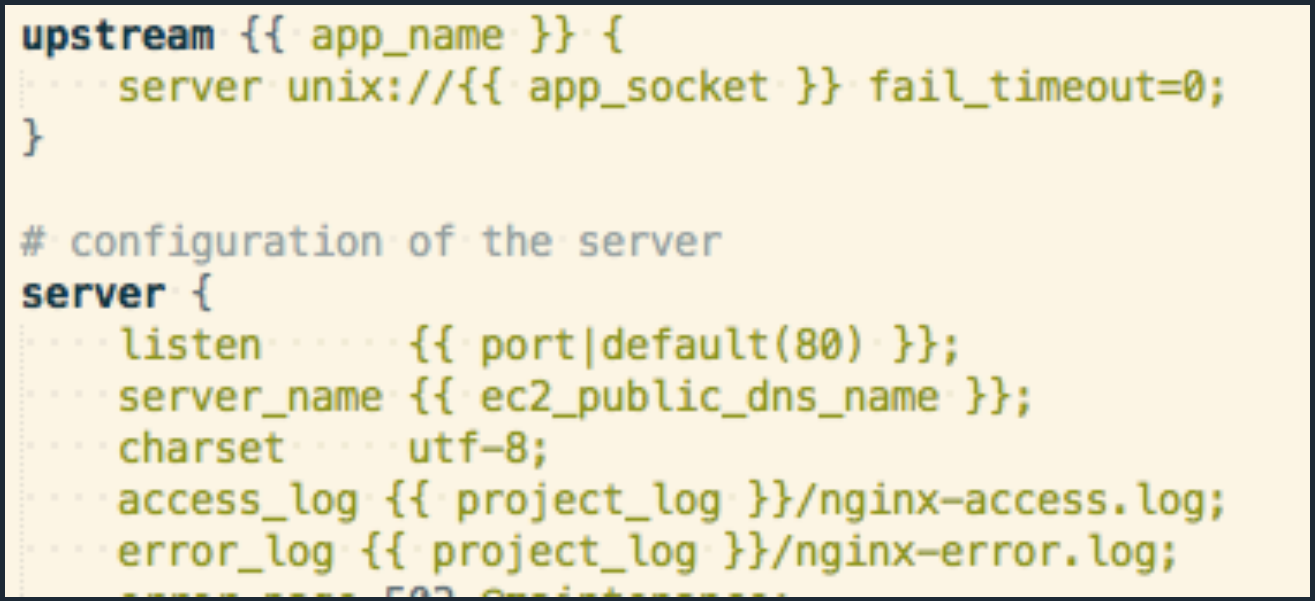
```
- name: Update apt and upgrade packages
  apt: update_cache=yes upgrade=safe cache_valid_time=3600
  sudo: yes
  tags: ['base-update']

- name: Install base dependencies
  apt: pkg={{ item }} state=latest
  with_items:
  - byobu
  - htop
  - build-essential
  - git
  - python-pip
  - python-dev
  - supervisor
  sudo: yes

- name: Install extra dependencies
  apt: pkg={{ item }} state=latest
  with_items: "{{ extra_pkgs }}"
  sudo: yes

- name: Remove any base packages
  apt: pkg={{ item }} state=absent
  with_items: "{{ remove_pkgs }}"
  sudo: yes

- name: Check if a reboot is required
  register: rebootfile
  stat: path=/var/run/reboot-required get_md5=no
  when: base_reboot
```

Ansible

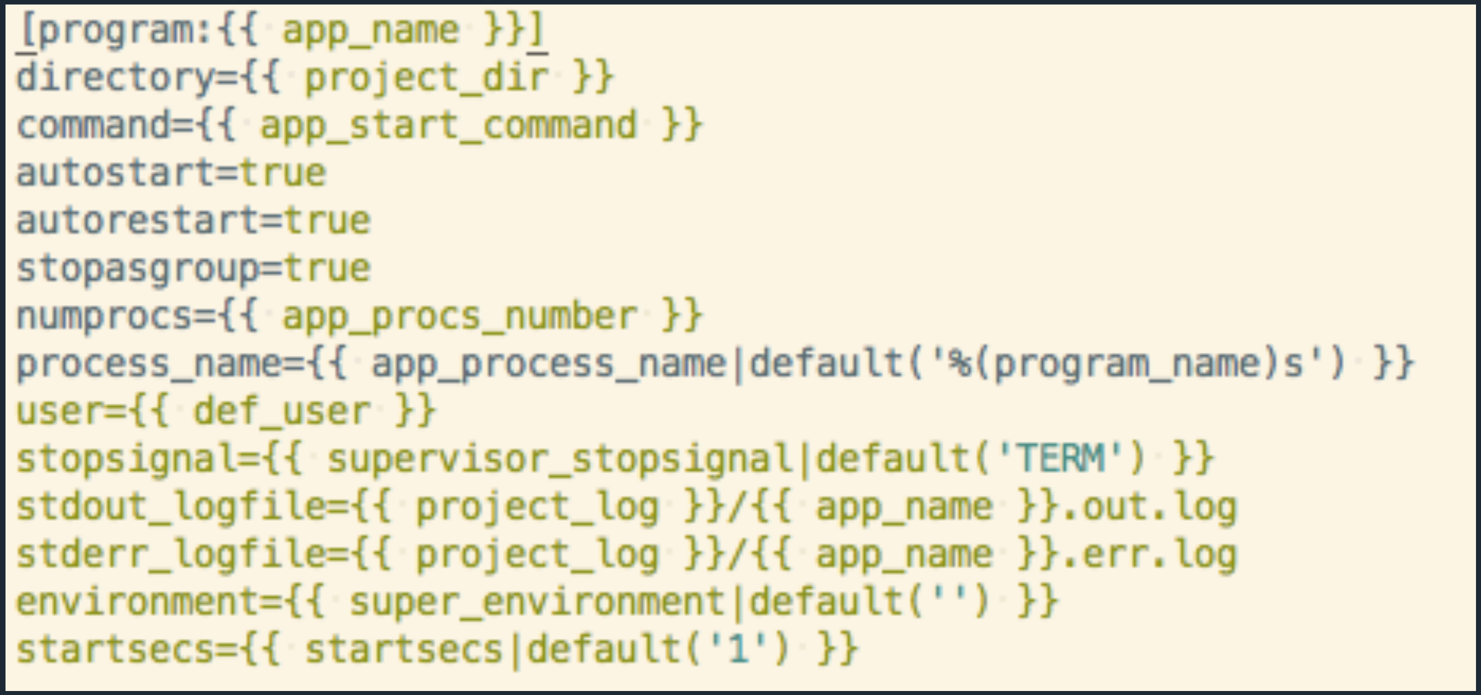


```
PLAY [Configure j[redacted] server] ***************************************

GATHERING FACTS ***********************************************************
ok: [[redacted]]

TASK: [base-common | Update apt and upgrade packages] *************************
changed: [[redacted]]

TASK: [base-common | Install base dependencies] *******************************
changed: [[redacted]] => (item=byobu,htop,build-essential,git,python-pip,python-dev,supervisor)

TASK: [base-common | Install extra dependencies] ******************************
changed: [[redacted]] => (item=hunspell,enchant,libenchant-voikko,voikko-fi)

TASK: [base-common | Remove any base packages] ********************************
changed: [[redacted]] => (item=aspell-en)

TASK: [base-common | Check if a reboot is required] ***************************
skipping: [[redacted]]

TASK: [base-common | Reboot the server] ***************************************
skipping: [[redacted]]

TASK: [base-common | waiting for server to come back] *************************
skipping: [[redacted]]

TASK: [pip | Check if there is a requirements.txt] ****************************
ok: [[redacted]]

TASK: [pip | Install the [redacted] requirements] *****************************
skipping: [[redacted]]

TASK: [pip | Install extra manual dependencies] *******************************
```

Ansible



```
upstream {{ app_name }} {
    server unix://{{ app_socket }} fail_timeout=0;
}

# configuration of the server
server {
    listen      {{ port|default(80) }};
    server_name {{ ec2_public_dns_name }};
    charset     utf-8;
    access_log {{ project_log }}/nginx-access.log;
    error_log {{ project_log }}/nginx-error.log;
```



```
[program:{{ app_name }}]
directory={{ project_dir }}
command={{ app_start_command }}
autostart=true
autorestart=true
stopasgroup=true
numprocs={{ app_procs_number }}
process_name={{ app_process_name|default('%(program_name)s') }}
user={{ def_user }}
stopsignal={{ supervisor_stopsignal|default('TERM') }}
stdout_logfile={{ project_log }}/{{ app_name }}.out.log
stderr_logfile={{ project_log }}/{{ app_name }}.err.log
environment={{ super_environment|default('') }}
startsecs={{ startsecs|default('1') }}
```

Ansible

Totally unrelated sidenotes

SSH keys, agent forwarding, environment variables and where to put passwords

(hint: not in source control)

Ansible

```
ansible-vault create secrets.yml

ansible-vault edit secrets.yml

ansible-vault view secrets.yml
```

Immutable Servers

When servers go online, they should *never* be touched

Avoid:

Configuration drift, unknown states, and the “nobody-really-knows-how-to-deploy-server-X-since-the-guy-with-the-beard-left” problem

Immutable Servers

But I thought you said Ansible was really good at managing servers?

Immutable Servers

Ansible is really good at ensuring a machine is in a known state. Which makes it great for setting up base images.

Immutable Servers

Package update?

New image!

Vulnerability patch?

New image!

Deploying multiple times a day?

New image(s)!

Base Images

1. Launch a machine
2. Run Ansible
3. Create an AMI

Base Images

packer.io

Builders: AWS EC2, Digital Ocean, Docker…

Provisioners: Shell, Ansible, Chef, Puppet…

Post-processors: Docker, Vagrant…

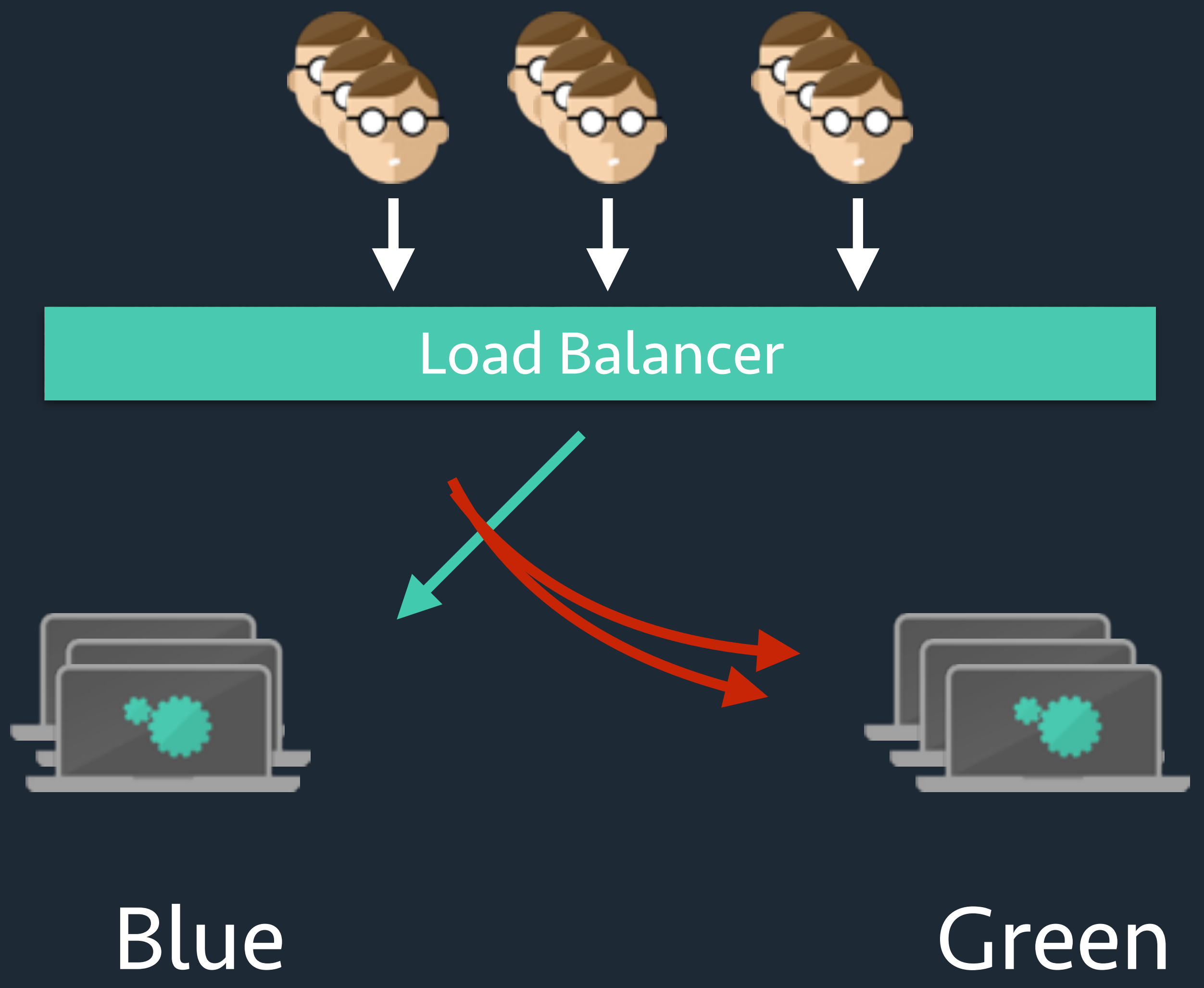
Deploying

Zero Downtime

No request drops

Allow rollbacks

Blue-Green Deployment



Unbabel

jobs@unbabel.com