Testing Infrastructure/Config Management Changes

Overview

It's probably you and your tests. This presentation will also be known as "my life as Dilbert comics".

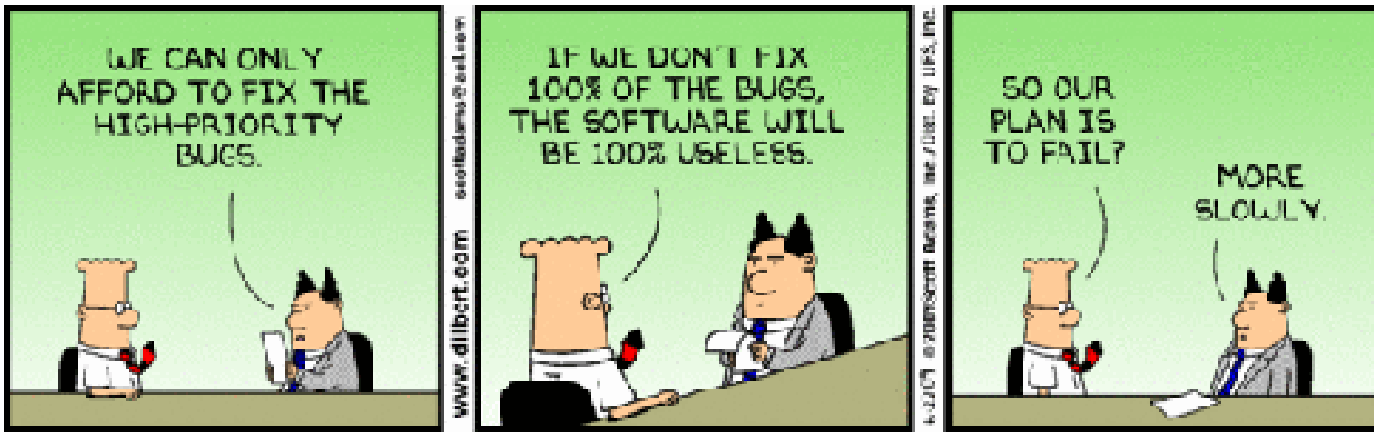
What is this "testing"?

Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation

Integration testing is the phase in software testing in which individual software modules are combined and tested as a group.

Thanks Wikipedia.

Why do we do it?



High level process to do it

> You can't automate a process you've never done manually.

This

- **Document**
  Document the steps for performing a process. For infrastructure, you'll document the type and version of components such as operating system, application and web servers, and databases. You'll also describe in detail how to install, configure, and run these infrastructure components. When using this approach, the key is to ***document to automate***, because you will eventually dispose of the documentation.
- **Test**
  Write an automated test that describes the intended outcome. For infrastructure, you can define features or expected outcomes such as a server running in a certain location, or the presence of a certain file or directory.
- **Script**
  Script all of the actions for the process. For infrastructure, you'll use tools such as Ansible/SaltStack/Chef/Puppet to define these environments according to the tests specified in the preceding step.
- **Version**
  Version source files. For infrastructure, these source files are defined in infrastructure automation scripts such as those created with Ansible or SaltStack.
- **Continuous**
  Whenever changes are made, test everything. Be sure that the system as a whole still behaves properly.

When to do it

1. Before committing your code back to the repository
2. Definitely before opening a PR
3. Absolutely before merging someones changes. You should trust your team, but verify their changes.

4. In a perfect world where we have more time/hands, upon merging in the code.
5. In a way more perfect world, automated upon merge/PR open/commit

---

Tooling to test your code

> There's a lot. Use the right tool for the right job. Or don't, it's your choice. If you wanted to combine testing frameworks, do it. Just test your shit.

- BATS - Bash Automated Testing System
- ServerSpec - RSpec tests for your infrastructure
- Inspec - Chef Auditing and Testing Framework
- Testinfra - Python project
- Ansible itself
- Molecule - beta-ish Ansible testing framework
- Cucumber - Behavior Driven Development
- Gauntlt - Security as code in a Cucumber-y syntax
- snyk - Scan your NodeJS code for security bugs
- PHPUnit - Because you technically could
- ...

What does this testing look like?

Unit testing

BATS

```
#!/usr/bin/env bats

@test "Ensure docker is installed" {
    run rpm -q docker-engine
    [ "$status" -eq 0 ]
}

@test "Ensure docker is running" {
    run systemctl is-active docker
    [ "$status" -eq 0 ]
}

@test "Install the latest docker-py" {
    run pip install docker-py --upgrade
    [ "$status" -eq 0 ]
}

@test "Run the playbook again to test the revert to docker-py 1.9.0 code" {
    run bash -c "ansible-playbook -i /tmp/kitchen/hosts
/tmp/kitchen/default.yml -c local && exit 0 || exit 1"
    [ "$status" -eq 0 ]
}
```

Integration testing

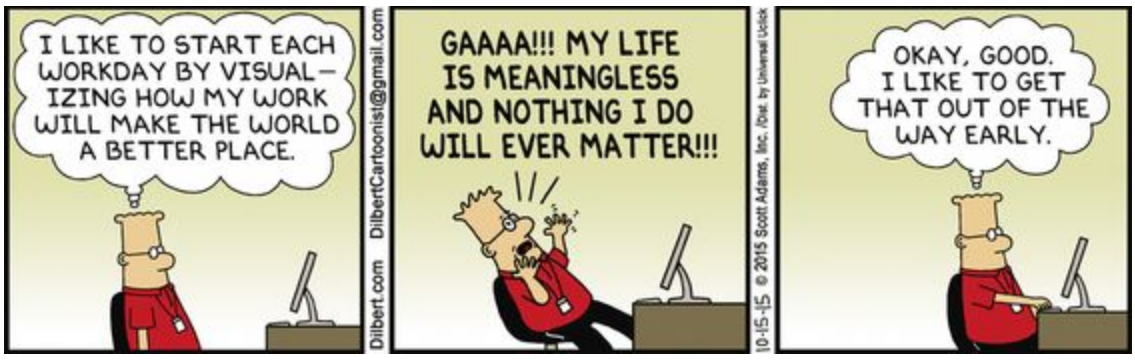
Test-Kitchen

```
driver:
  name: vagrant
  use_sudo: false

provisioner:
  hosts: test-kitchen
  name: ansible_playbook
  require_ansible_repo: false
  require_ansible_omnibus: true
  ansible_verbosity: 2
  ansible_verbose: true
  ansible_diff: true
  require_chef_for_busser: true
  update_package_repos: false
  requirements_path: test/requirements.yml
  ssh_known_hosts:
    - bitbucket.org

platforms:
  - name: centos/7

suites:
  - name: default

transport:
  forward_agent: true
```

Integration Testing pt. 2

```
#!/usr/bin/env bats

@test "Idempotence test - the second run should change nothing" {
    run bash -c "ansible-playbook -i /tmp/kitchen/hosts
/tmp/kitchen/default.yml -c local | grep -q 'changed=0.*failed=0' && exit 0
|| exit 1"
    [ "$status" -eq 0 ]
}
```

How to start doing it less manually



Remember folks, you're still dealing with computers and as my momma always said, "Computers is the devil!" She didn't really say that, but she could have.

```
git clone git@bitbucket.org:greenlancer/ansible-role-docker.git

cd ansible-role-docker

bundler install

bundler exec kitchen list

bundler exec kitchen create

ssh-add -D
ssh-add -k ~/.ssh/$YOUR_BITBUCKET_KEY
bundler exec kitchen setup

# This is the actual testing portion
bundler exec kitchen verify
```

```
-----> Starting Kitchen (v1.12.0)
-----> Converging <default-centos-7>...
       Preparing files for transfer
       Preparing playbook
       Preparing inventory
       Preparing modules
       nothing to do for modules
       Preparing roles
       Preparing ansible.cfg file
       Found existing ansible.cfg
       Preparing group_vars
       nothing to do for group_vars
       Preparing additional_copy_path
       Preparing host_vars
       nothing to do for host_vars
       Preparing hosts file
       Preparing spec
       nothing to do for spec
       Preparing library plugins
       nothing to do for library plugins
       Preparing callback plugins
```

```
       nothing to do for callback plugins
       Preparing filter_plugins
       nothing to do for filter_plugins
       Preparing lookup_plugins
       nothing to do for lookup_plugins
       Finished Preparing files for transfer
       Installing ansible using ansible omnibus
-----> Installing Ansible Omnibus
       downloading
https://raw.githubusercontent.com/neillturner/omnibus-ansible/master/ansib
le_install.sh
         to file /tmp/ansible_install.sh
       trying curl...
         % Total    % Received % Xferd  Average Speed   Time    Time
Time  Current
                                Dload  Upload   Total   Spent    Left
Speed
100  4954  100  4954    0     0  25143      0 --:--:-- --:--:-- --:--:--
25275
       Transferring files to <default-centos-7>
       Add bitbucket.org to ~/.ssh/known_hosts
       TASK [ansible-role-docker : Gather facts again to become aware of
any new interfaces] ***
       task path: /tmp/kitchen/roles/ansible-role-docker/tasks/main.yml:54
       ok: [localhost]

       PLAY RECAP
*********************************************************************
       localhost                  : ok=10   changed=0    unreachable=0
failed=0

       Finished converging <default-centos-7> (0m24.40s).
-----> Setting up <default-centos-7>...
       Finished setting up <default-centos-7> (0m0.00s).
-----> Kitchen is finished. (0m24.74s)


....


-----> Starting Kitchen (v1.12.0)
-----> Verifying <default-centos-7>...
       Preparing files for transfer
-----> Busser installation detected (busser)
       Installing Busser plugins: busser-bats
       Plugin bats already installed
       Removing /tmp/verifier/suites/bats
       Transferring files to <default-centos-7>
-----> Running bats test suite
  Ensure docker is installed
  Ensure docker is running
  Install the latest docker-py
  Run the playbook again to test the revert to docker-py 1.9.0 code
```

```
Idempotence test - the second run should change nothing
```

```
       5 tests, 0 failures
       Finished verifying <default-centos-7> (0m10.66s).
-----> Kitchen is finished. (0m11.00s)
```