

# Vagrant and Splunk Development

Building Splunk Apps for Any  
Environment

Jason Rauen

October 2019





## Splunk Usergroup Slack

Senior Lead Technologist | Booz Allen Hamilton



## Badarsebard

Splunk Usergroup Slack

# Forward-Looking Statements



During the course of this presentation, we may make forward-looking statements regarding future events or plans of the company. We caution you that such statements reflect our current expectations and estimates based on factors currently known to us and that actual events or results may differ materially. The forward-looking statements made in the this presentation are being made as of the time and date of its live presentation. If reviewed after its live presentation, it may not contain current or accurate information. We do not assume any obligation to update any forward-looking statements made herein.

In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only, and shall not be incorporated into any contract or other commitment. Splunk undertakes no obligation either to develop the features or functionalities described or to include any such feature or functionality in a future release.

Splunk, Splunk>, Turn Data Into Doing, The Engine for Machine Data, Splunk Cloud, Splunk Light and SPL are trademarks and registered trademarks of Splunk Inc. in the United States and other countries. All other brand names, product names, or trademarks belong to their respective owners. © 2019 Splunk Inc. All rights reserved.





# Building Splunk Apps

---

Should be easy right?

# How It Starts

Simple apps require simple solutions

But what happens when things get complicated?

- Distributed and clustered architectures
- Support for multiple architectures
- Multiple developers
- Operating system specific requirements
- Support for multiple operating systems



# Past Solutions

## Create a remote environment for developers

### Pros

- Access to the required operating system

### Cons

- Mock deployments
  - For *each* architecture
- Multiple deployments for multiple team members
- Developer collisions
- Typically relies on another team for provisioning





# Past Solutions

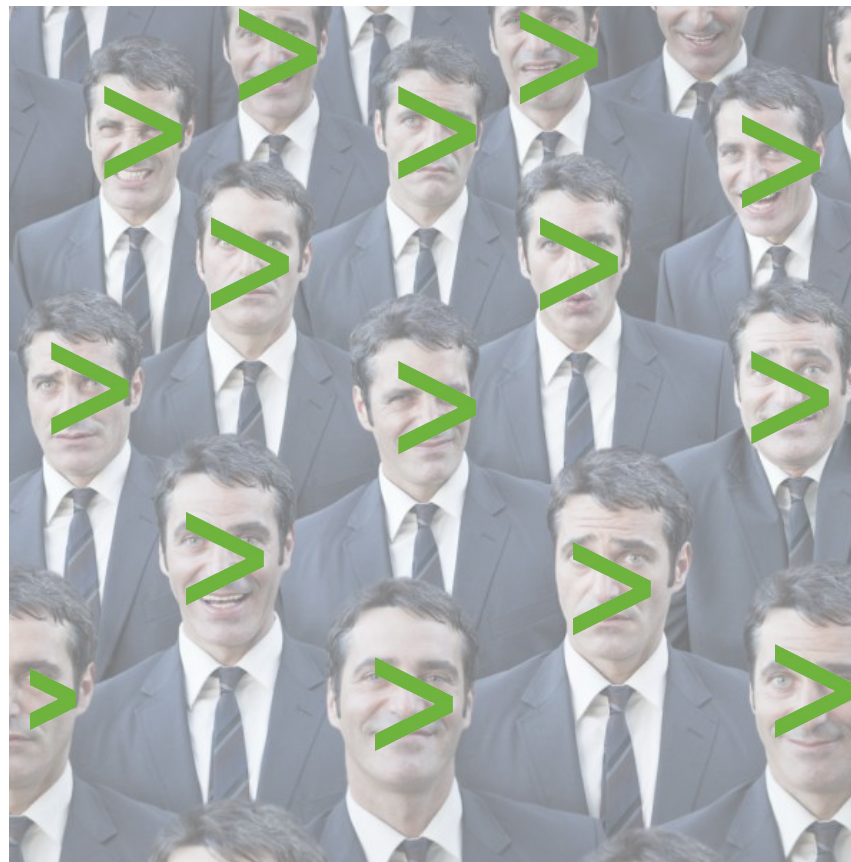
## Multiple Splunk instances on a host

### Pros

- Completely local solution
  - [https://wiki.splunk.com/Community:Runmultiple\\_Splunks\\_on\\_one\\_machine](https://wiki.splunk.com/Community:Runmultiple_Splunks_on_one_machine)

### Cons

- Not supported
- Difficult
- Unfriendly to version control
- Running is a pain



# Virtualization

Remote environments from the comfort of \$HOME

- Run a virtualization tool like Virtualbox, Vmware, or Hyper-V
- Sweet! Problem solved. Time for cake.





# Except...

We need to do the following for each architecture



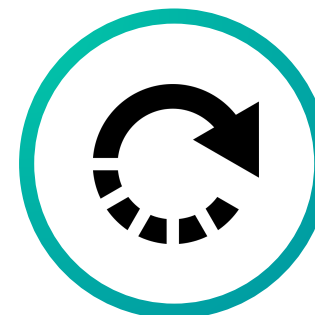
Create the virtual machine



Install and configure Splunk



Install and configure Splunk



Repeat for every server in every architecture



Document the process



# Vagrant

---

Virtualization made easy

# Enter Vagrant

## Stage left

- Manage and run virtual machines simply
- Single file configuration for reproducible, portable, automated VMs
- Not a virtualization provider





# Docker

## A Contextual Sidebar

- Docker is a possible solution for some of the previously mentioned challenges.
- See the following .conf talks for examples:
- [DE123416](#)
- [SF88089](#)
- [HL010](#)
- [FN118645](#)



# Goal

## Vagrant as a Solution

- Docker is a good solution.
- In some circumstances Vagrant is better.
- Primary goal is to highlight Vagrant so developers have choice.
- Why I used Vagrant for PyDen.



# Basic Anatomy of Vagrant

Three steps to virtualization victory



Vagrant CLI



Vagrantfile



Base Box



# Vagrant CLI

- A binary package you drop in your \$PATH



# Vagrantfile

- Ruby-like syntax configuration file
- Small and sharable through VCS



# Base Box

- Shared starting image.
- Selected and modified by Vagrantfile.
- Public repository at [app.vagrantup.com](https://app.vagrantup.com) or share through corporate file sharing tools.







# A Single Simple Workflow

---

Because ninjas are busy

# How to Splunk with Vagrant

- Vagrant provides us with a base virtual machine we can work on
- But how does it help us create a consistent deployment of Splunk from which to work?



# Provisioners

## Eureka!

- Automated provisioning will install and configure software on the virtual machine.
- Some common provisioner tools available for use:
  - Shell
  - Ansible
  - CFEngine
  - Chef
  - Docker
  - Puppet
  - Salt
- Defined in the Vagrantfile.





# Synced Folders

Easier than FTP

- Sync files between host and guest.
- Provider specific.
- Mapping defined in Vagrantfile.
- Default mapping between project root and /vagrant on guest.



+



=



# Vagrant and Multi-Machine Configs

## One Config to Rule Them ALL

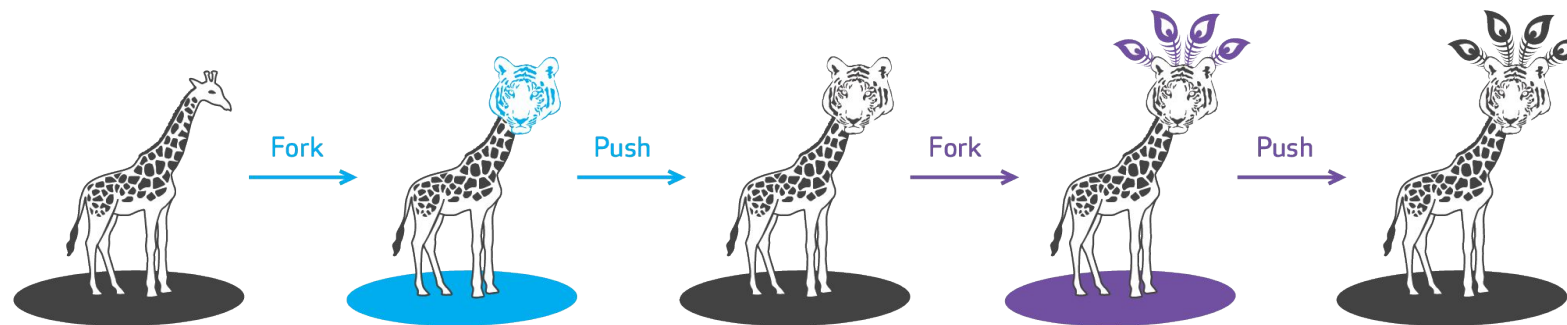
- Vagrant supports multi-machine environments
- Defined in the same Vagrantfile.
- Networking options available



# Multiple Architectures

Building for all the things

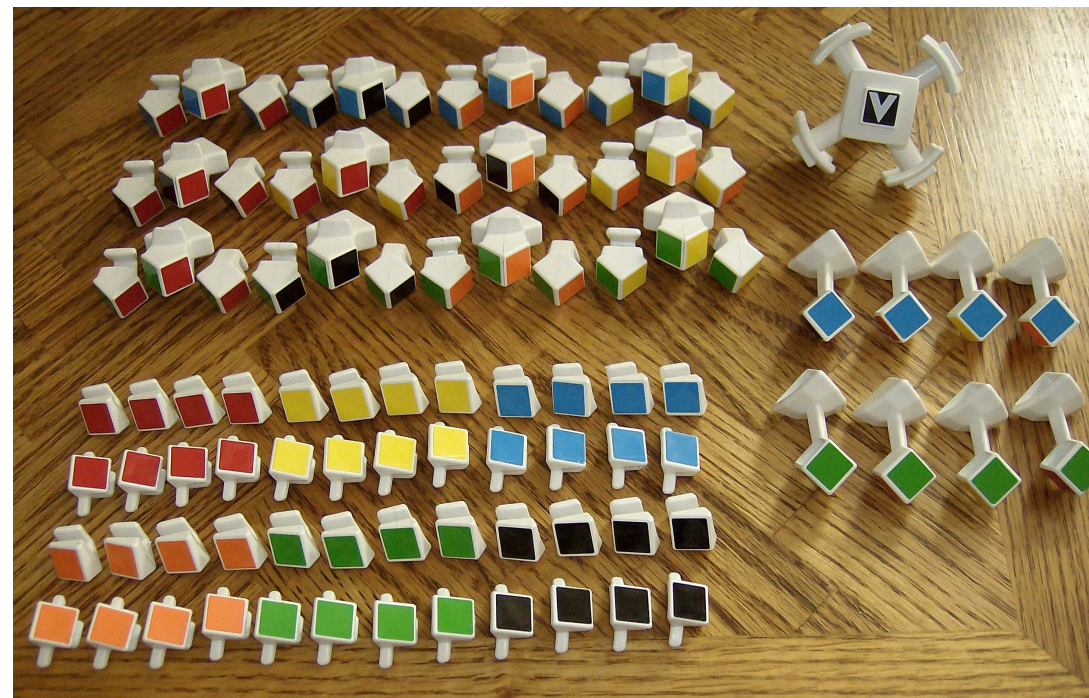
- Support for multiple architectures through multiple Vagrantfiles
- No limit to Vagrantfile location
- Control particular architecture via changing working directory



# Putting It All Together

The pieces fit!

- Single binary and config file
- Single Configuration for:
  - Creating and provisioning machines
  - Syncing files between them and the host
  - Networking requirements
  - Multiple machine architectures
- Multiple Configurations for multiple architectures



# Developer Workflow

## A day in the life of a Developer with Vagrant

### Initial

- Download Vagrant
- Clone repository or initialize Vagrant
- Install preferred virtualization provider

### Daily

- Pull the latest commits from the repo
- Run Vagrant
- Start making code changes and see them reflected in Splunk





# Demo

```
>vagrant init
```

```
A `Vagrantfile` has been placed in this directory. You are now  
ready to `vagrant up` your first virtual environment! Please read  
the comments in the Vagrantfile as well as documentation on  
`vagrantup.com` for more information on using Vagrant.
```

```
>
```

```
>cat .\Vagrantfile
# -*- mode: ruby -*-
# vi: set ft=ruby :

# All Vagrant configuration is done below. The "2" in Vagrant.configure
# configures the configuration version (we support older styles for
# backwards compatibility). Please don't change it unless you know what
# you're doing.
Vagrant.configure("2") do |config|
  # The most common configuration options are documented and commented below.
  # For a complete reference, please see the online documentation at
  # https://docs.vagrantup.com.

  # Every Vagrant development environment requires a box. You can search for
  # boxes at https://vagrantcloud.com/search.
  config.vm.box = "base"

  # Disable automatic box update checking. If you disable this, then
  # boxes will only be checked for updates when the user runs
  # `vagrant box outdated`. This is not recommended.
  # config.vm.box_check_update = false

  # Create a forwarded port mapping which allows access to a specific port
  # within the machine from a port on the host machine. In the example below,
  # accessing "localhost:8080" will access port 80 on the guest machine.
  # NOTE: This will enable public access to the opened port
  # config.vm.network "forwarded_port", guest: 80, host: 8080

  # Create a forwarded port mapping which allows access to a specific port
```

```
>cat .\Vagrantfile
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/bionic64"
end
>
```



```
vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
=> default: Importing base box 'ubuntu/bionic64'...
=> default: Matching MAC address for NAT networking...
=> default: Checking if box 'ubuntu/bionic64' version '20190417.0.0' is up to date...
=> default: Setting the name of the VM: ubuntu_default_1566259631633_21513
=> default: Fixed port collision for 22 => 2222. Now on port 2200.
=> default: Clearing any previously set network interfaces...
=> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
=> default: Forwarding ports...
    default: 22 (guest) => 2200 (host) (adapter 1)
=> default: Running 'pre-boot' VM customizations...
=> default: Booting VM...
=> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 127.0.0.1:2200
    default: SSH username: vagrant
    default: SSH auth method: private key
    default: Warning: Connection aborted. Retrying...
    default: Warning: Connection reset. Retrying...
    default: Warning: Remote connection disconnect. Retrying...
    default:
    default: Vagrant insecure key detected. Vagrant will automatically replace
    default: this with a newly generated keypair for better security.
    default:
    default: Inserting generated public key within guest...
    default: Removing insecure key from the guest if it's present...
    default: Key inserted! Disconnecting and reconnecting using new SSH key...
=> default: Machine booted and ready!
=> default: Checking for guest additions in VM...
    default: The guest additions on this VM do not match the installed version of
    default: VirtualBox! In most cases this is fine, but in rare cases it can
    default: prevent things such as shared folders from working properly. If you see
    default: shared folder errors, please make sure the guest additions within the
    default: virtual machine match the version of VirtualBox you have installed on
    default: your host and reload your VM.
    default:
    default: Guest Additions Version: 5.2.18
    default: VirtualBox Version: 6.0
=> default: [vagrant-hostsupdater] Checking for host entries
=> default: Mounting shared folders...
    default: /vagrant => C:/Users/584363/Documents/vagrant-sandboxes/ubuntu
```



```
>vagrant ssh
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-47-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Tue Aug 20 00:11:05 UTC 2019

System load:  0.02          Processes:            99
Usage of /:   9.9% of 9.63GB Users logged in:             0
Memory usage: 12%          IP address for enp0s3: 10.0.2.15
Swap usage:   0%

2 packages can be updated.
2 updates are security updates.

vagrant@ubuntu-bionic:~$
```

```
>vagrant suspend
==> default: Saving VM state and suspending execution...
==> default: [vagrant-hostsupdater] Removing hosts
>
```

```
>vagrant halt  
==> default: Attempting graceful shutdown of VM...  
==> default: [vagrant-hostsupdater] Removing hosts  
>  
■
```

```
> vagrant destroy  
  default: Are you sure you want to destroy the 'default' VM? [y/N] y  
==> default: Destroying VM and associated drives...  
==> default: [vagrant-hostsupdater] Removing hosts  
> █
```

```
>cat .\Vagrantfile
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config|
  config.vm.box = "badarsebard/ubuntu-18.04-splunk"
  config.vm.hostname = "mysplunkbox"
  config.vm.network "private_network", ip: "192.168.33.10"
end
>
```



```
>vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Importing base box 'badarsebard/ubuntu-18.04-splunk'...
==> default: Matching MAC address for NAT networking...
==> default: Checking if box 'badarsebard/ubuntu-18.04-splunk' version '7.2.4.0' is up to date...
==> default: Setting the name of the VM: ubuntu_default_1566261422978_2636
==> default: Fixed port collision for 22 => 2222. Now on port 2200.
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
==> default: Forwarding ports...
    default: 22 (guest) => 2200 (host) (adapter 1)
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 127.0.0.1:2200
    default: SSH username: vagrant
    default: SSH auth method: private key
    default:
    default: Vagrant insecure key detected. Vagrant will automatically replace
    default: this with a newly generated keypair for better security.
    default:
    default: Inserting generated public key within guest...
    default: Removing insecure key from the guest if it's present...
    default: Key inserted! Disconnecting and reconnecting using new SSH key...
==> default: Machine booted and ready!
==> default: Checking for guest additions in VM...
    default: The guest additions on this VM do not match the installed version of
    default: VirtualBox! In most cases this is fine, but in rare cases it can
    default: prevent things such as shared folders from working properly. If you see
    default: shared folder errors, please make sure the guest additions within the
    default: virtual machine match the version of VirtualBox you have installed on
    default: your host and reload your VM.
    default:
    default: Guest Additions Version: 5.2.18
    default: VirtualBox Version: 6.0
==> default: [vagrant-hostsupdater] Checking for host entries
==> default: Mounting shared folders...
    default: /vagrant => C:/Users/584363/Documents/vagrant-sandboxes/ubuntu
>
```

```
vagrant@mysplunkbox:~$ sudo su - splunk
splunk@mysplunkbox:~$ /opt/splunk/bin/splunk start --accept-license
```

This appears to be your first time running this version of Splunk.

Splunk software must create an administrator account during startup. Otherwise, you cannot log in.  
Create credentials for the administrator account.  
Characters do not appear on the screen when you type in credentials.

Please enter an administrator username: admin

Password must contain at least:

- \* 8 total printable ASCII character(s).

Please enter a new password:

Please confirm new password:

Copying '/opt/splunk/etc/openldap/ldap.conf.default' to '/opt/splunk/etc/openldap/ldap.conf'.

Generating RSA private key, 2048 bit long modulus

.....+++++

.....+++++

e is 65537 (0x10001)

writing RSA key

Generating RSA private key, 2048 bit long modulus

.....+++++

.....+++++

e is 65537 (0x10001)

writing RSA key

Moving '/opt/splunk/share/splunk/search\_mrsparkle/modules.new' to '/opt/splunk/share/splunk/search\_mrsparkle/modu



# splunk>enterprise

## First time signing in?

If you installed this instance, use the username and password you created at installation. Otherwise, use the username and password that your Splunk administrator gave you.

If you've forgotten your credentials, contact your Splunk administrator.

First time signing in?

- WARNING
- Do not use Vagrant in a production environment.
- Vagrant is inherently insecure.





# Case Study: PyDen

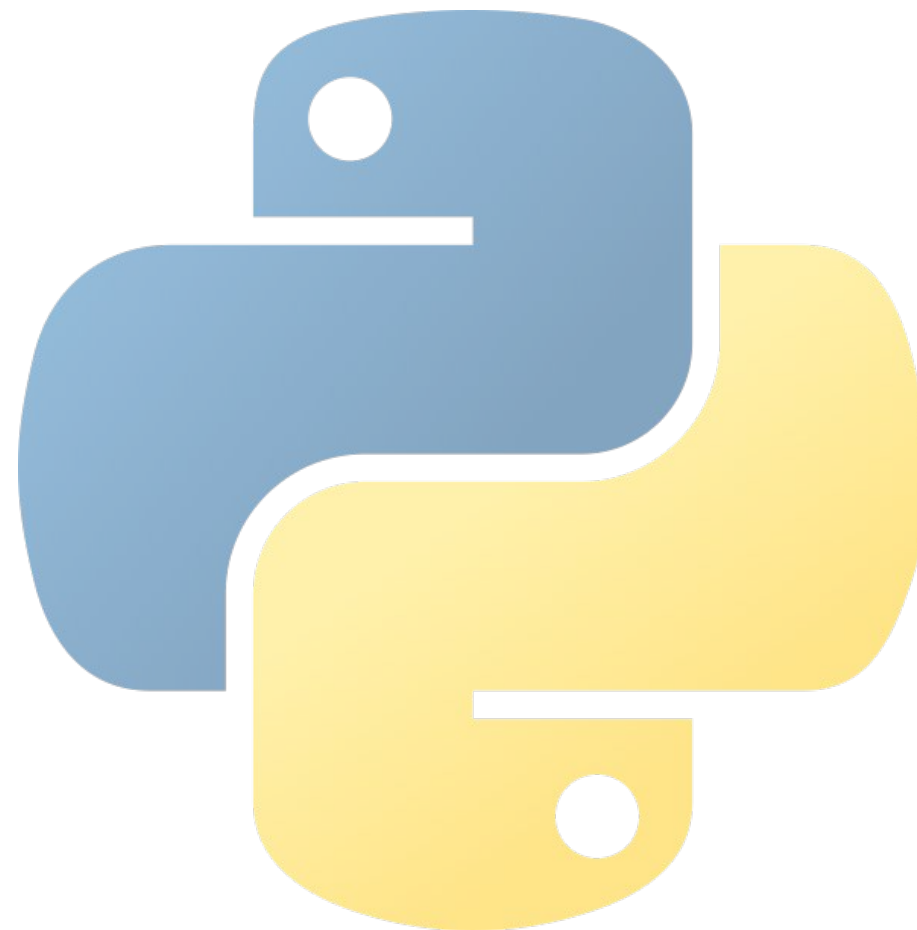
The start of my journey with Vagrant



# PyDen

## Why have just one Python?

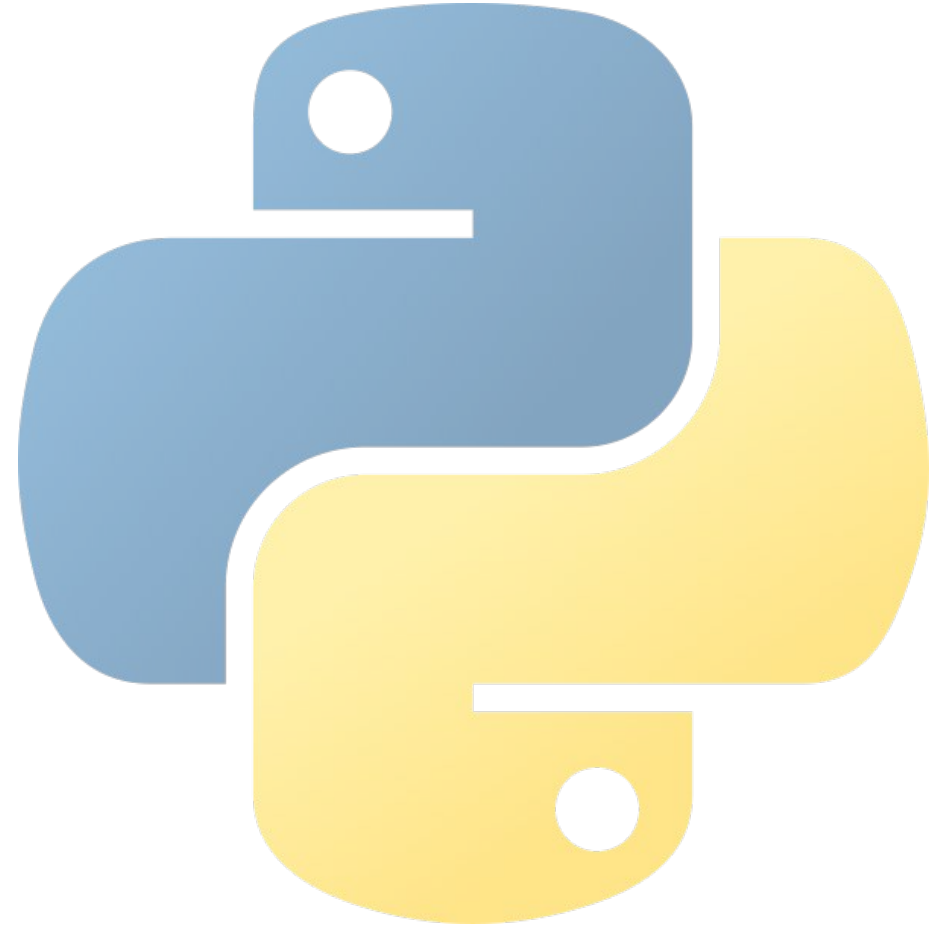
- PyDen is an app for leveraging multiple Python versions, virtual environments, and the Python Package Index inside Splunk
- Deploying the app is a two step process:
  - Create the Python distributions and virtual environments on a staging server (e.g. deployer)
  - Deploy the prepared app out to the production servers (e.g. search heads, indexers)
- Developed using Vagrant with multiple architectures
- <https://github.com/badarsebard/pyden-suite>



# PyDen

## The behind the scenes tale

- Builds Python from source
- Does a process swap and the OS level.
- Multiple components
- Multiple architectures supported.





# Vagabond

---

The Vagrant for Splunk Project

# Vagabond

A rogue. A knave. A rascalion!

- Open-source “barebones” Splunk project utilizing Vagrant.
- Hosted on GitLab and GitHub.
  - <https://gitlab.com/badarsebard/vagabond>
  - <https://github.com/badarsebard/vagabond>
- Jumping off point for Splunk projects.
- Supports developing multiple apps within single project
- Examples of multi-machine architectures
- Includes (optional) reference to a Splunk base box built on Ubuntu 18



# Q&A

---





# Contact

---

Email: [jason.rauen@gmail.com](mailto:jason.rauen@gmail.com)

Slack: Badarsebard

Linkedin: [www.linkedin.com/in/jason-rauen](https://www.linkedin.com/in/jason-rauen)



# Thank

# You



Go to the .conf19 mobile app to

**RATE THIS SESSION**

