Splunk Configuration Management and Deployment with Ansible

Jose Hernandez

Director – Security Solutions, Zenedge

Sean Delaney

Client Architect, Splunk

splunk>

Intros

splunk>

Disclaimer

During the course of this presentation, we may make forward looking statements regarding future events or the expected performance 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 could differ materially. For important factors that may cause actual results to differ from those contained in our forward-looking statements, please review our filings with the SEC. 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, this presentation may not contain current or accurate information. We do not assume any obligation to update any forward looking statements we may make.

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 functionality described or to include any such feature or functionality in a future release.

Agenda

- Deploying Splunk with Ansible
- Git for Configuration Management
- Git for Configuration Monitoring
- Demo
- Take Away

Deploying Splunk with Ansible

splunk>

Deployment Tools

- Why use a deployment tool?
  - Automate Deployment (full lifecycle)
    - Provision systems and Operating System
    - Create users and the application environment
    - Deploy/update binaries and scripts
    - Deploy/update configuration files
    - Control – start/stop/restart services
  - Deployment Server?
    - Only deploys Splunk configurations under $SPLUNK_HOME/etc/apps

Many choices:

- Ansible
- Puppet
- Chef
- CFEngine
- Salt
- BladeLogic
- ……

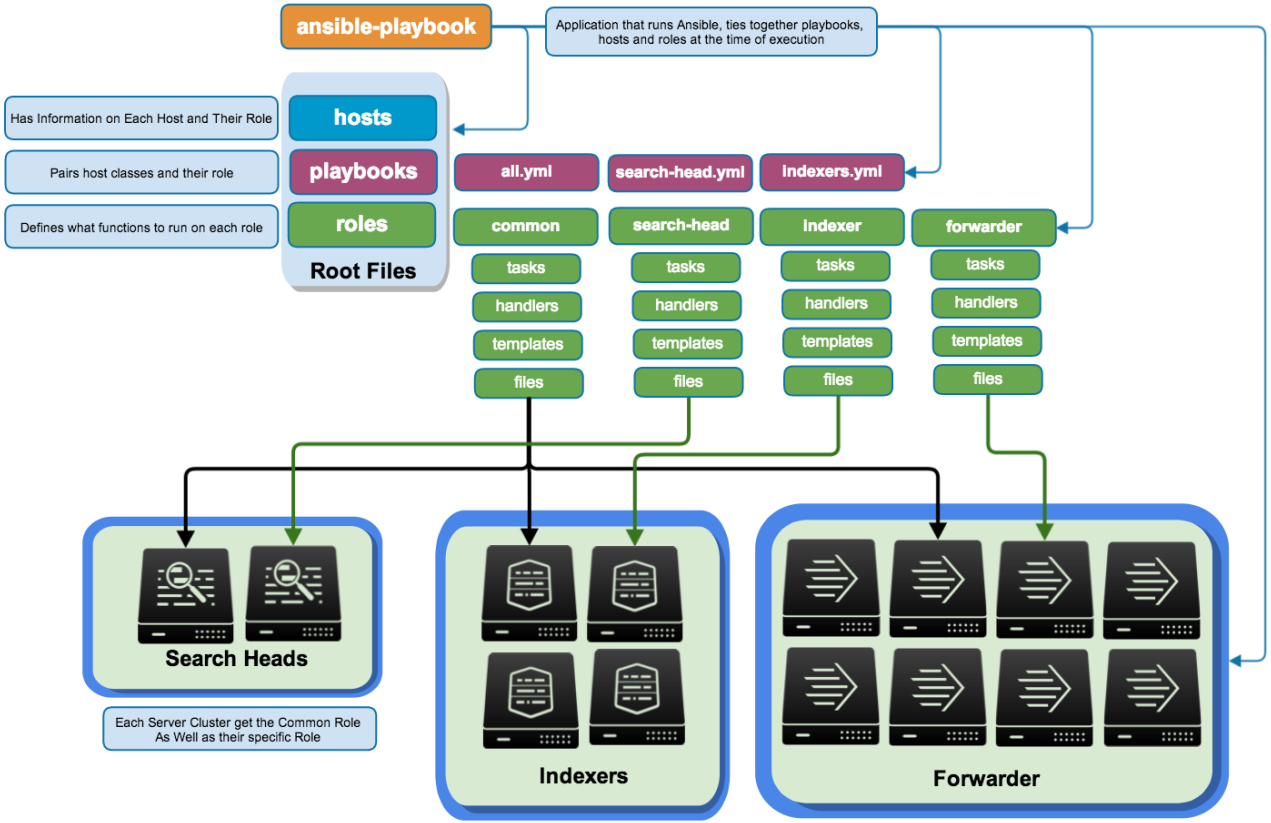
Why Ansible?

- No Agent Required
- Uses SSH as Transport
- Easy to pickup
- Low overhead and scales to huge deployments
- Python Base
- Windows deployments via Powershell

Ansible Primer

- Ansible-playbook: Ansible executable which runs the playbooks etc..
- Hosts: INI file which contains the role/group and host mapping
- Playbooks: Ties in Roles, host groups and task together to create orchestrated actions on target hosts
- Roles: contains the actions each group will complete (this is where the deployment logic lives)

Ansible Structure



Running Ansible

Requirements:

- Ansible Installed
- Splunk-admin user updated with your keys under /playbooks/splunk_creds/splunk-admin.pub
- Root password of hosts to run Ansible in
- Make sure you have ssh keys generated for root
- Hosts inventory updated

Running Ansible, cont…

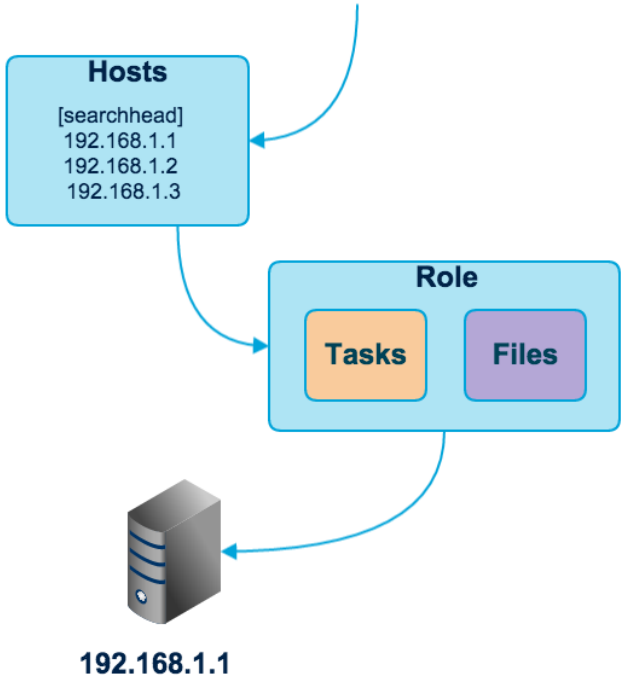
- Before running Ansible make sure that your environment is set correctly. Run:
- `. /opt/ansible/hacking/env-setup`

- To build a splunk server from scratch just run:
- `./ansible-playbook /etc/ansible/playbook/search_heads.yml`

- Make sure that you have hosts defined under hosts

Running a Playbook

ansible-playbook playbooks/searchhead.yml -l 192.168.1.1



Running Searchhead Playbook

```
/etc/ansible#ansible-playbook playbooks/search_heads.yml
PLAY [apply common configuration to all nodes] ********************************
GATHERING FACTS ***************************************************************
ok: [162.243.231.42]
TASK: [common | install security controls] *************************************
ok: [162.243.231.42] => (item=chkrootkit,rkhunter,clamav,fail2ban)
TASK: [common | install basic utilities] **************************************
ok: [162.243.231.42] => (item=vim,screen,iotop,htop,ioping,ntp)
TASK: [common | create splunk-admin] ******************************************
ok: [162.243.231.42]
TASK: [common | copy splunk-admin bash_profile] *******************************
ok: [162.243.231.42]
```

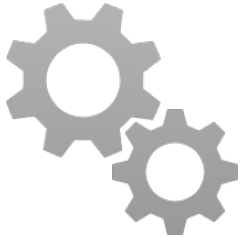
Splunk and Git

Part 1: Configuration Management

splunk>

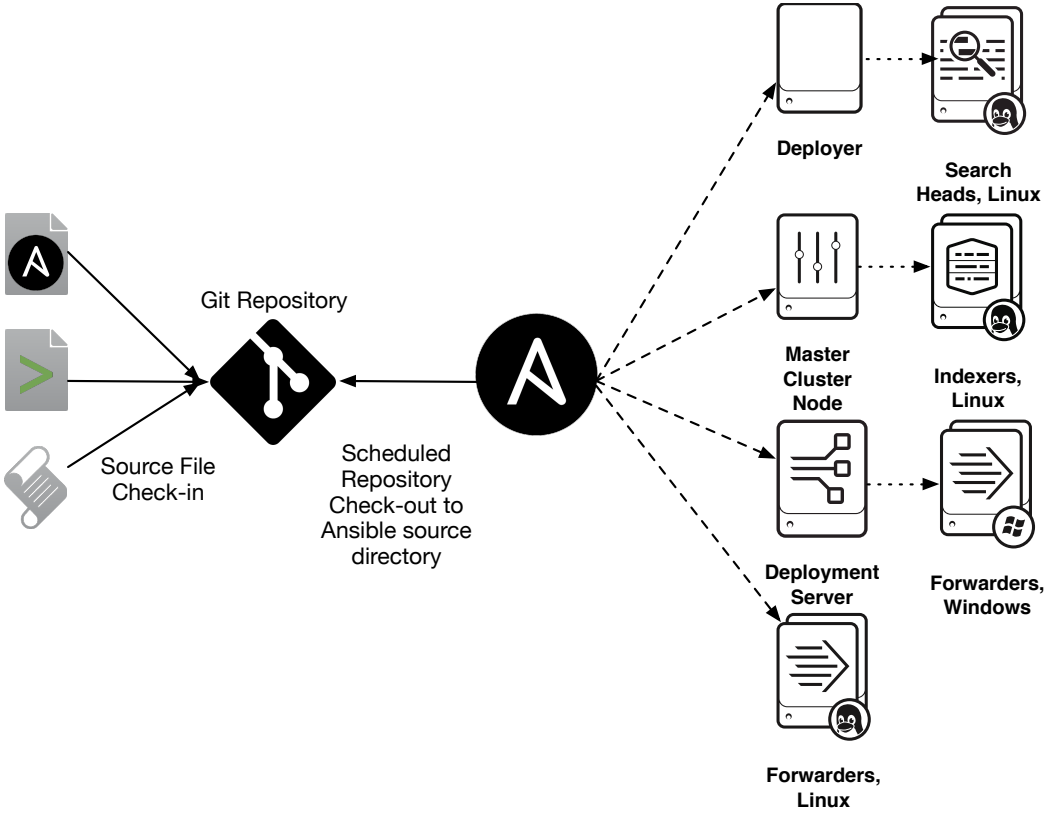
DevOps Approach

- Treat Configuration Files as code
  and test, deploy programmatically

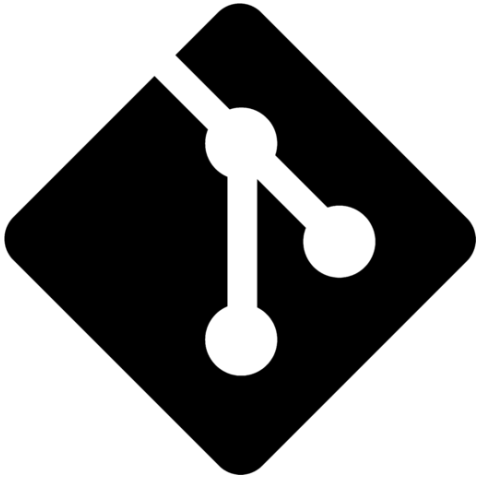


- Apply QA/change management controls
  - Gold reference copy
  - Check-ins and diffs (Who, What, When changed)
  - Combine with CM/Ticketing System (Who and Why)
  - Easy roll-back to known good state

Configuration Deployment



Git Repository Tree



```
ansible-splunk-simple]
└── $tree -L 3 -d
.
├── group_vars
├── images
├── playbooks
│   ├── splunk_binaries
│   └── splunk_creds
│       ├── splunk_binaries
│       └── splunk_creds
└── roles
    ├── cluster_master
    │   ├── files
    │   ├── handlers
    │   └── tasks
    ├── common
    │   ├── files
    │   ├── handlers
    │   ├── tasks
    │   ├── templates
    │   └── vars
    ├── ec2
    │   ├── defaults
    │   ├── handlers
    │   ├── meta
    │   ├── tasks
    │   └── vars
    ├── indexer
    │   ├── files
    │   ├── handlers
    │   └── tasks
    ├── license_master
    │   ├── files
    │   ├── handlers
    │   └── tasks
    ├── search_head
    │   ├── files
    │   ├── handlers
    │   └── tasks
    └── universal_forwarder
        ├── files
        ├── handlers
        └── tasks
```

Git: Clone, Sample And Create Your Own Repository

- `git clone <repo> /etc/ansible`
- `rm -rf .git`
- `git init`
- `git add *`
- `git commit -m 'my first commit'`
- `git remote add origin <your new repo url>`
- `git push -u origin masterFile check-in`

Git: Checking Updated Files

- `git status`

```
[redacted]nsible:/etc/ansible$ git status
On branch rc
Your branch is up-to-date with 'origin/rc'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        modifiedfile.txt

nothing added to commit but untracked files present (use "git add" to track)
```

- `git add modifiedfile.txt`

```
[redacted]@ansible:/etc/ansible$ git status
On branch rc
Your branch is up-to-date with 'origin/rc'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   modifiedfile.txt
```

- `git commit -m 'add your commit message here'`
- `git push origin master`

Git: Checkout to Ansible Source

- On the Ansible server run the following in a script via cron

- `git fetch --all`
- `git reset --hard origin/master`

Splunk and Git

Part 2: Configuration Monitoring

splunk>

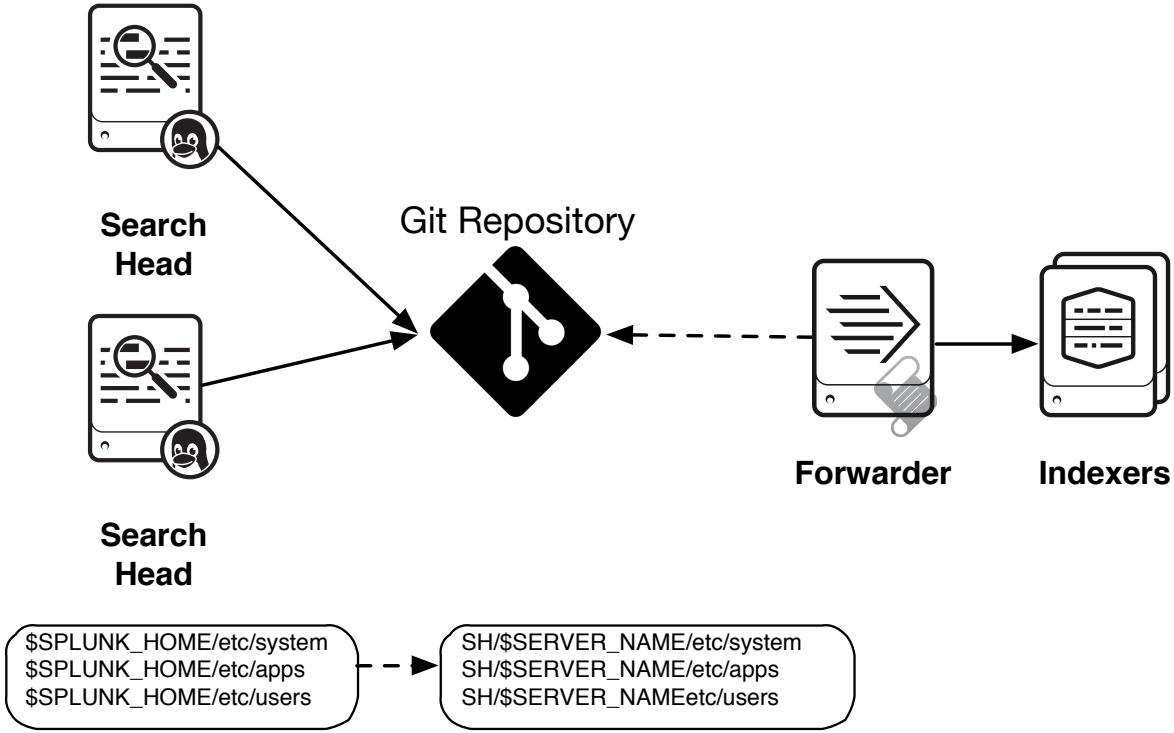
Problem: Search Load Gone Crazy?

- One or many users have created or modified a dashboard search or saved scheduled that is creating excess load on your Splunk servers
- How do you find which search is the culprit?

Solution: Monitoring Changes to Search Configs

- On search heads setup a cron script to check-in any changes Git on the following directories:
- `$SPLUNK_HOME/etc/system`
- `$SPLUNK_HOME/etc/apps`
- `$SPLUNK_HOME/etc/users`
- Use a scheduled scripted input on a forwarder to collect regular file changes and index the changes in Splunk
- `git whatchanged`
- Once indexed you can search for changes over a time window

Splunking Searchhead Config Changes



Putting it in Action

Demo Time

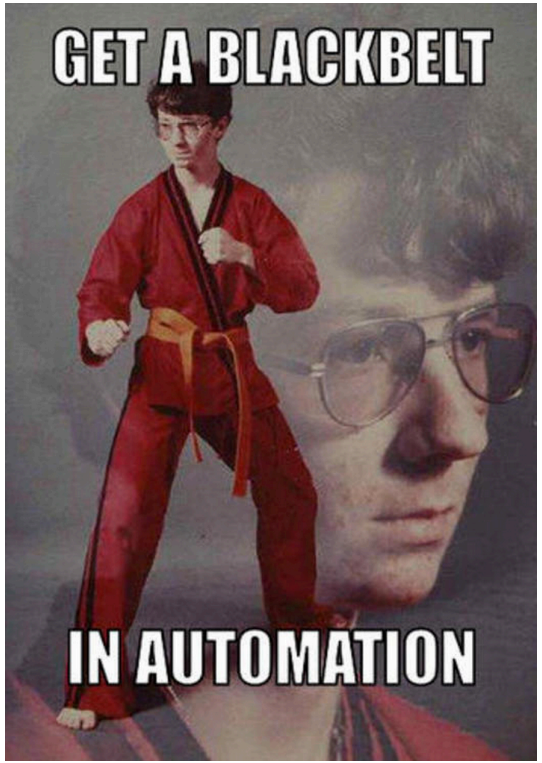
splunk>

Take Away

- Automation with Ansible takes some work up from, but will will make life simpler in the long run
- Using Git for Splunk/Ansible configuration management allows for change management and simplified roll backs.
- Checking in Searchhead configs into Splunk provides the ability to detect Admin and User Search changes

Resources

- Deploying Splunk Securely with Ansible Config Management – Part 1
- http://blogs.splunk.com/2014/07/12/deploying-splunk-securely-with-ansible-config-management-part-1/
- Deploying Splunk Securely with Ansible Config Management – Part 2
- http://blogs.splunk.com/2015/02/09/deploying-splunk-securely-with-ansible-config-management-part-2/



What Now?

Related breakout sessions and activities…

THANK YOU

splunk>