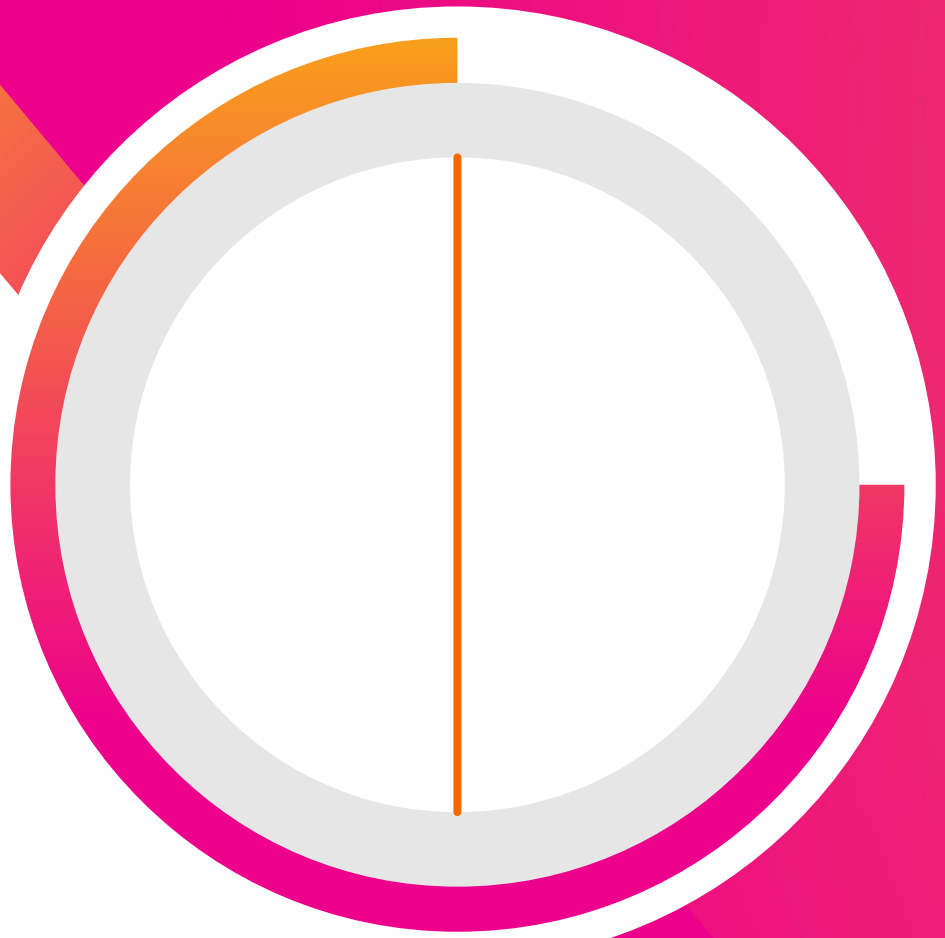


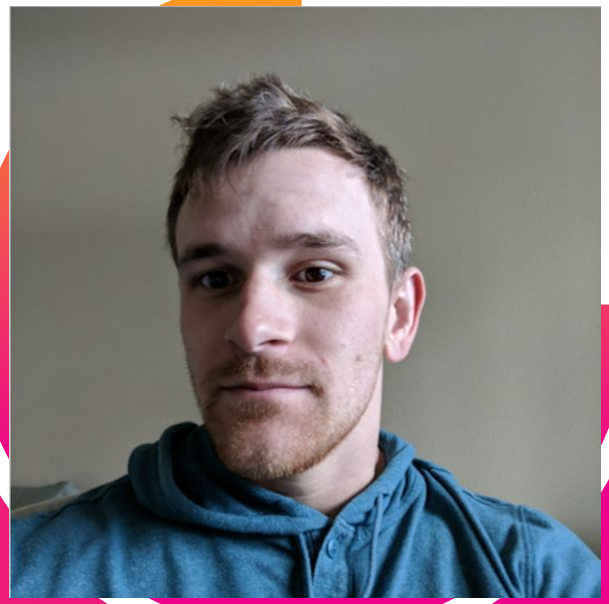
Can’t We Just Have a Bot Run Our Deployments?

Yes, Yes We Can.

Shelbie Wise & Mitchell Peters
Automation Stuff | Optum

splunk> .conf19

Shelbie Wise
Architect Analyst | Optum
Splunk Admin



Mitchell Peters
Sr IT Architect Analyst | Optum
Automation Engineer

splunk> .conf19

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.

Who is Optum?

splunk> .conf19

The Optum Splunk Project Scope

Started from the bottom now we're here

1. In 2018 our team was given the task to onboard **all of Optum's top critical** business applications into Splunk.

- 100+ Clients at 10-100+ hosts each
- Each client will receive their own index, and search app which means we will be pushing conf files per client on a regular basis

Limitations at Our Company

Lack of root access for all Unix hosts

Adding Mounts/file systems

- Orchestration tools and daemons

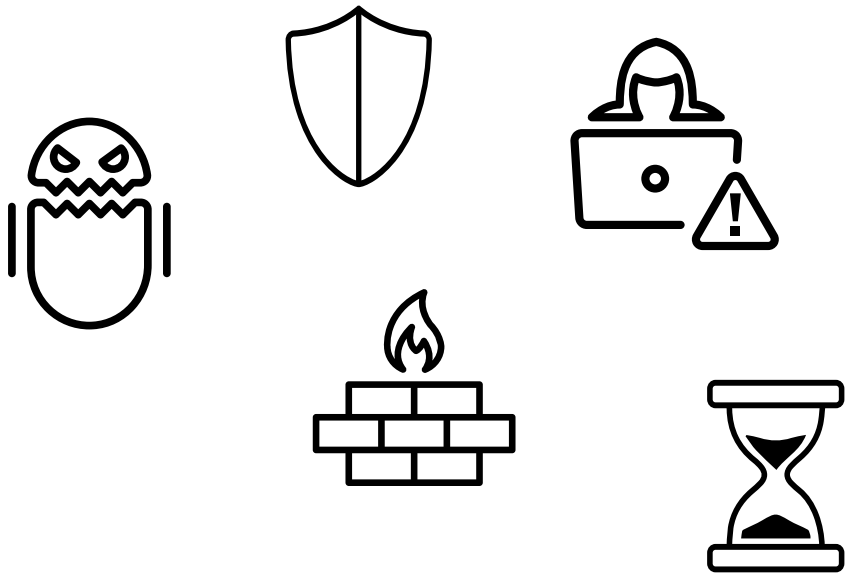
Segregated networks

- Talking is forbidden

Acquisitions

- Different infrastructures

Authentication



Goals:

Solutions > Problems

1. What is the fastest way we can install 1000+ forwarders
2. How can we schedule our deployments, so they can be hands free
3. How are we going to fix missing forwarders without boot start enabled
4. Who’s going to manage the cluster bundle updates?

The Current Optum Splunk Environment

EST 2018

Indexers

Search Heads

Universal Forwarders

End Users

Unlimited

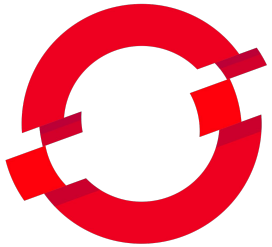
Splunk IT Service Intelligence™

The Technology We Use

Flowdock

GitHub

Jenkins

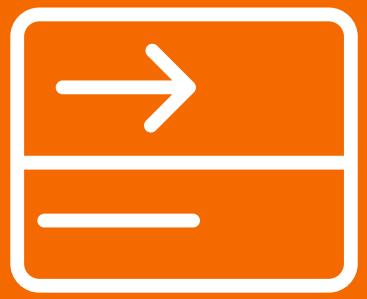


OPENSHIFT

bash-$

Bot Technology and Automation Breakdown

What and how we use our bot and automation on a regular basis



Deploy-UF

How we install and turn on our forwarders to thousands of hosts

Missing Forwarders

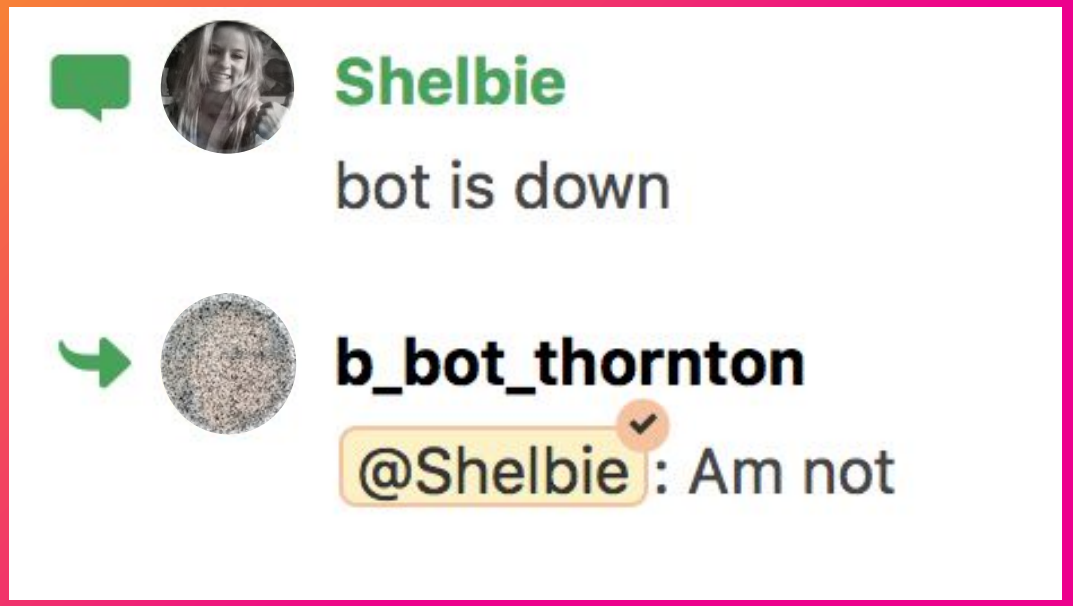
An alternative to enable boot start…because we don't have that privilege

Config Deployments

In case you wanted to see how we manage our bundle updates

Meet Our Bot

splunk> .conf19



Meet Our Bot

Legal made us cover his face

- Name: b_bot_thornton
- Breed: Hubot
- Home: OpenShift Origin
- Age: 1.5 years old

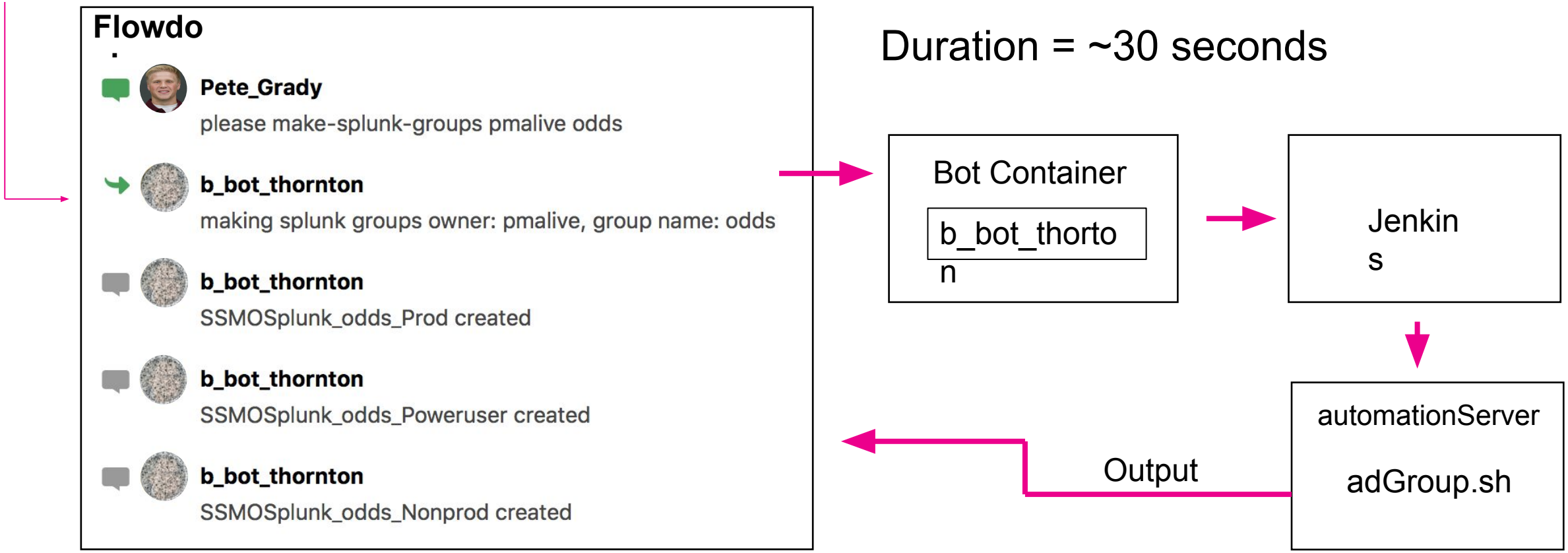
Setting Up the Bot

How we did it, how you can too

1. Set up the environment for the bot install
   - We chose to use a container because it was easily available to us at the time…not saying you *have* to do this
2. Install **Hubot** (open source)
3. Install the **Authentication Module**
   - Had to re-write some of the core module to allow configs to be stored in **OpenShift Origin**
4. Write your hooks

Bot Technology Flow

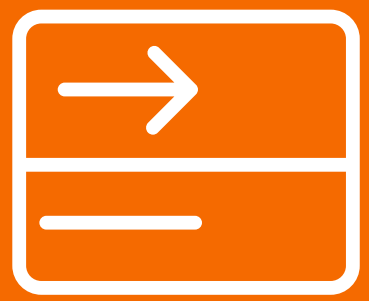
User posts their command



Example Above: Make-Groups command – our bot grabs the arguments and creates ad groups via api, groups are then used to map to Splunk roles

Bot Technology and Automation Breakdown

What and how we use our bot and automation on a regular basis



Deploy-UF

How we install and turn on our forwarders to thousands of hosts

Missing Forwarders

An alternative to enable boot start…because we don't have that privilege

Config Deployments

In case you wanted to see how we manage our bundle updates

deploy-uf

splunk> .conf19

deploy-uf – Prerequisites (optum pre-reqs)

Linux/Sun/AIX	Windows
• Key authentication • File System/mount point needs to be in place with proper permissions	• Need to have admin access • WinRm must be enabled

deploy-uf – Prerequisites

More details Splunk nuid & key authentication

Splunk NUID

We created one non user id account (nuid) to own the forwarder installs

- Git repo holds our teams public keys
- This allows our team to all have access via keys
- No Passwords

keyPush.sh

Nightly cron runs the above script and pushes keys to hosts

Scripts adds our nuid account and pushes keys

Uses Ansible, Bash, and Jenkins

deploy-uf.sh

What does it do?

- Manages the clients and forwarder installs on the hosts

What technology does it use?

- Flowdock (chat ops – splunk admins)
- GitHub (holds the host list)
- Jenkins (runs the job)
- Shell Scripting (controls logic checks & installs)
- Ansible (manages the connections to the hosts)

Logic Checks:

- Operating system
- Is telnet installed, if so:
  - Test connectivity to our splunk deployment server & indexers (some lol)
- File System
  - 10 GB available
  - Permissions (can we write)
- Does the Forwarder already exist?
  - Checksum command against the splunk binary
  - Is NTP running

Deploy-uf: How Do We Run It?

From a Splunk admin's point of view

Splunk Admin Steps:

1. Create the host list in Github
2. Run the bot command in Flowdock
3. Validate and trouble shoot any failures

Flowdock – Bot deploy-uf run command

Date & Time

Host list

On/Off

Notice the run finished and posted back by 15:01

Shelbie

at 2019-06-25 15:01:00 deploy-uf OrxTibcoSW.inv on

b_bot_thornton

Here's the plan: date: Tue Jun 25 2019 15:01:00 GMT-0500 (CDT), file: OrxTibcoSW.inv, command: on .. to cancel, say:

```
deploy-uf cancel OrxTibcoSW.inv_on_1561492860000
```

Jun 25, 2019 15:01

b_bot_thornton

[tibco]

#badufdeployer run by: splunknu

tickets: none

hostname: , success: yes

hostname: , success: yes

hostname: , success: yes

hostname: , success: yes

hostname: , success: yes

hostname: , success: yes

hostname: , success: yes

hostname: , success: yes

hostname: , success: yes

hostname: , success: yes

hosts names

Script Output

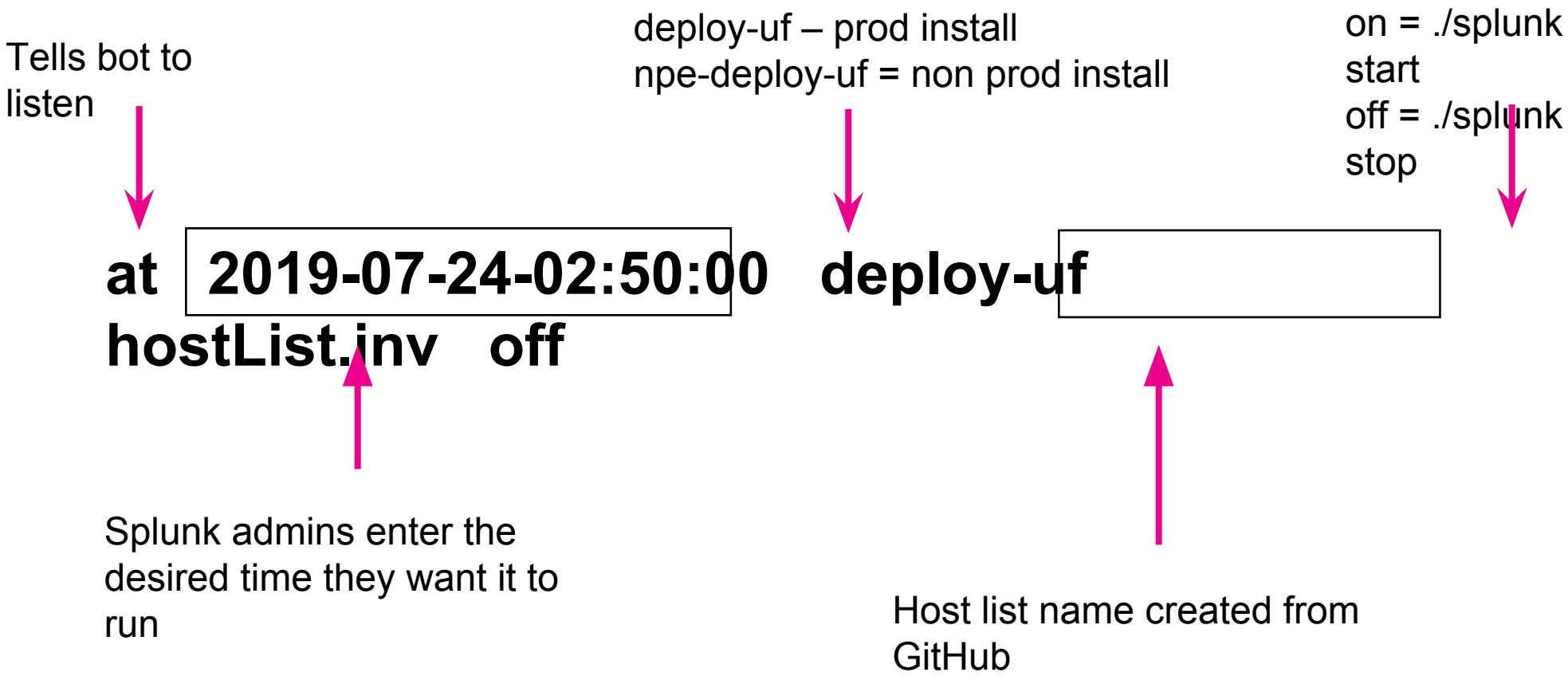
From the logging/splunk perspective

List ▾ Format 50 Per Page ▾

All Fields

i	Time	Event
>	8/27/19 11:01:23.000 AM	{ [-] date: Tue Aug 27 11:01:23 CDT 2019 deploymentSum: cksum-is-good hostname: [redacted] isItReally: Its-running monitorGood: /monitor-exists-is-readable-and-is-writable-splunknu running: no success: yes time-manager: ntp-running } Show as raw text host = [redacted] source = /deployer/npe-runstatus.stuff sourcetype = deployer_status
>	8/27/19 10:30:35.000 AM	{ [-] date: Tue Aug 27 10:30:35 CDT 2019

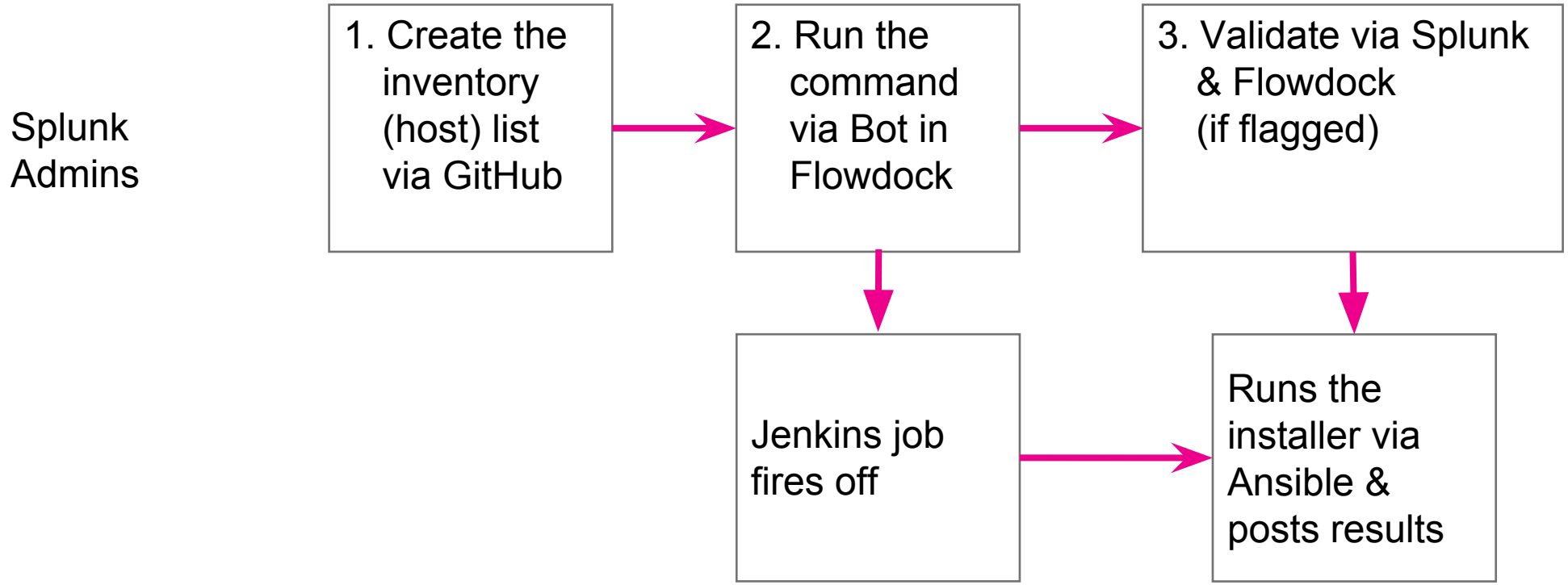
deploy-uf – Command Break Down



NOTE: we use “off” to install the UF without starting it

deploy-uf – Overall Flow

What is happening behind the scenes



deploy-uf – How Did This Help Our Team?

It **saved us LOADS** of time

It allowed us to **schedule** the run

It allowed us to **grant offshore access** to run automation

We make **less mistakes**

Because of the success with the automation our team was able to prove that we no longer were considered a **“risky”** install and were able to **achieve a pre approval process** so that we could schedule CRs whenever ***we*** needed.

Bot Technology and Automation Breakdown

What and how we use our bot and automation on a regular basis

Deploy-UF

How we install and turn on our forwarders to thousands of hosts

Missing Forwarders

An alternative to enable boot start…because we don't have that privilege

Config Deployments

In case you wanted to see how we do it

missing_forwarders

splunk> .conf19

Missing Forwarders

A work around solution for not having root privs

What does it do?

Utilizes our deploy-uf script to run every 30 minutes against all hosts that report missing from the Monitoring Console

Why did we need this?

- We don’t have root access to enable boot-start on our linux/aix/sun hosts
- Helps us cleanse out retired hosts
- Assists in identifying corrupt file systems

Missing Forwarders

General flow – this process runs every 30 minutes

Monitoring Console

Generate the missing forwarder list via **dmc_forwarder_assets** lookup

Custom alert is set up via splunk to trigger a script to the ansible server

ansibleServer

Takes the list from the **dmc_forwarder_assets** and checks it against the **serverclass.conf**

IF a host is in BOTH lists, missing_forwarders.sh will call Jenkins to run deploy-uf.sh

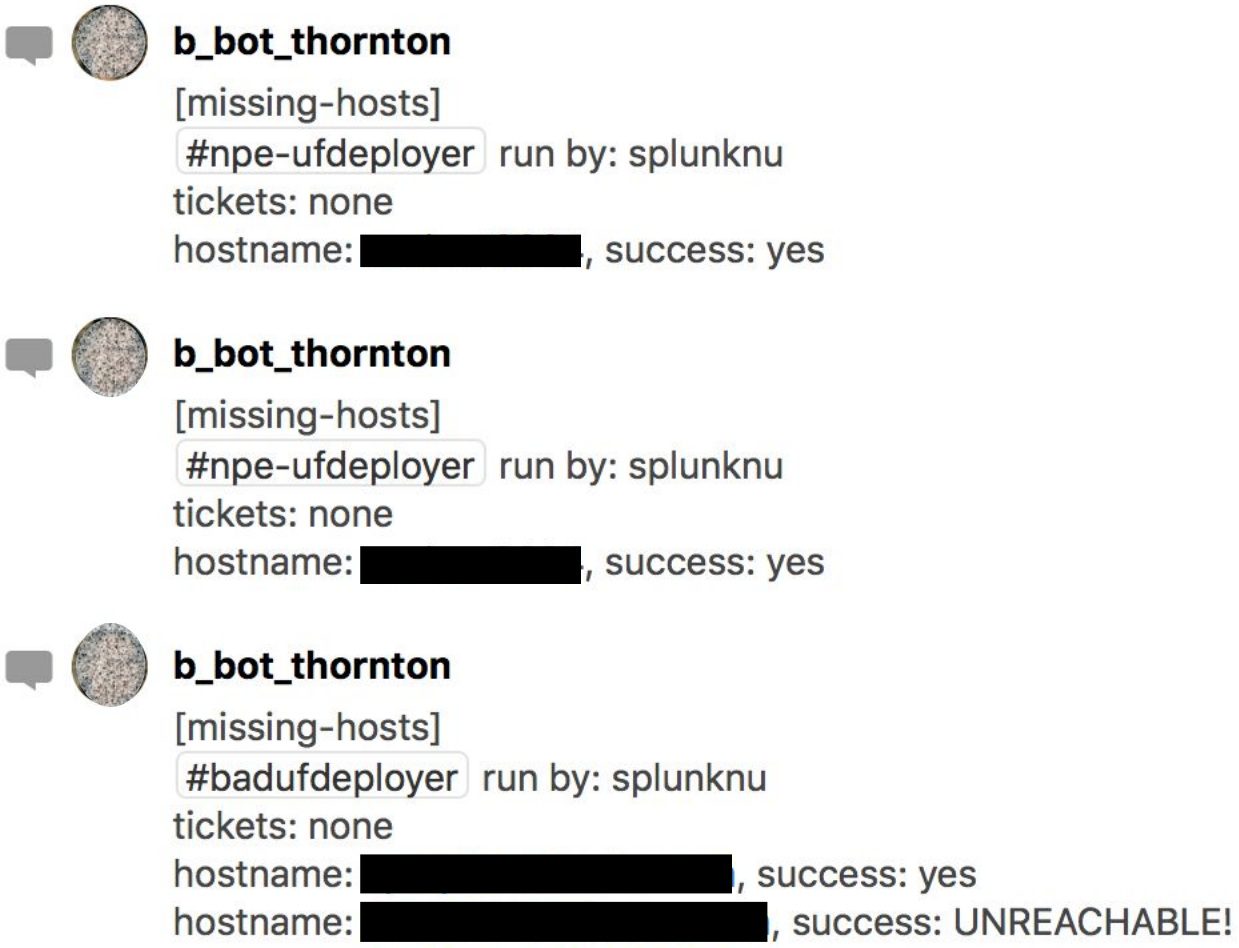
deploy-uf.sh

Attempts to restart/reinstall UF

Posts output to Flowdock

Missing Forwarders

Output in Flowdock



Note: Hostnames and images have been covered for personal reasons

Bot Technology and Automation Breakdown

What and how we use our bot and automation on a regular basis

Deploy-UF

How we install and turn on our forwarders to thousands of hosts

Missing Forwarders

An alternative to enable boot start…because we don't have that privilege

Config Deployments

In case you wanted to see how we do it

Config Deployments

splunk> .conf19

Deployment Configs

How we have our backend configs set up in git

Deployment Server

DS Repo

- serverclass.conf
- inputs.conf

Cluster Master

CM Repo

- indexes.conf
- props.conf
- transforms.conf

Search Head Deployer

SHD Repo

- app.conf
- authorize.conf
- authentication.conf
- local.meta
- user-prefs

Each repo has their own Jenkins job with a cron

Config Deployments – Technology Used

Git hooks post notifications to Jenkins

- Set some notifications in git, set Jenkins to listen to them

Jenkins executes the scripts, and notifies of success and failure

- Define your next steps either in a Jenkins or leverage existing scripts and save some re-work for another day

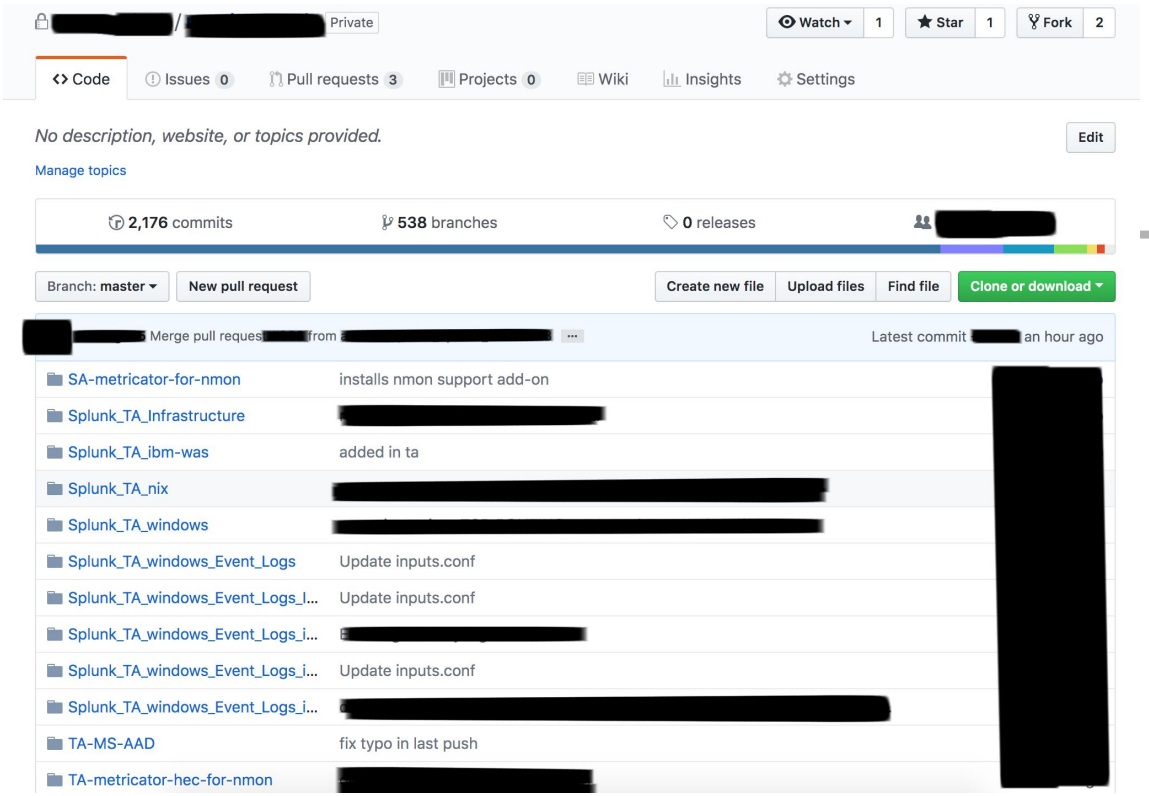
Script runs on splunk cluster member, reports status back to chat

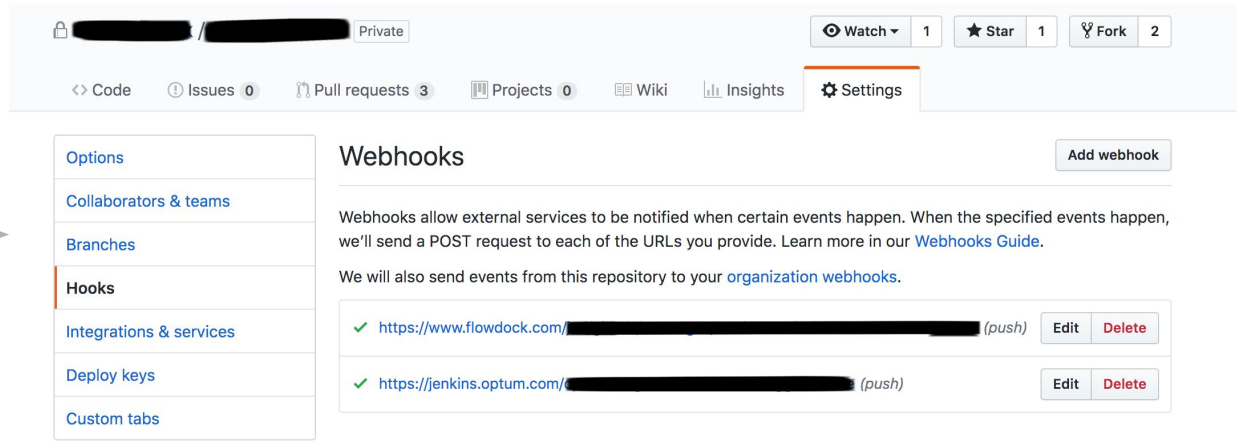
- Make sure you enable hooks where you need them, (see next slide)

You can run it all in one place if your environments support it, ours does not

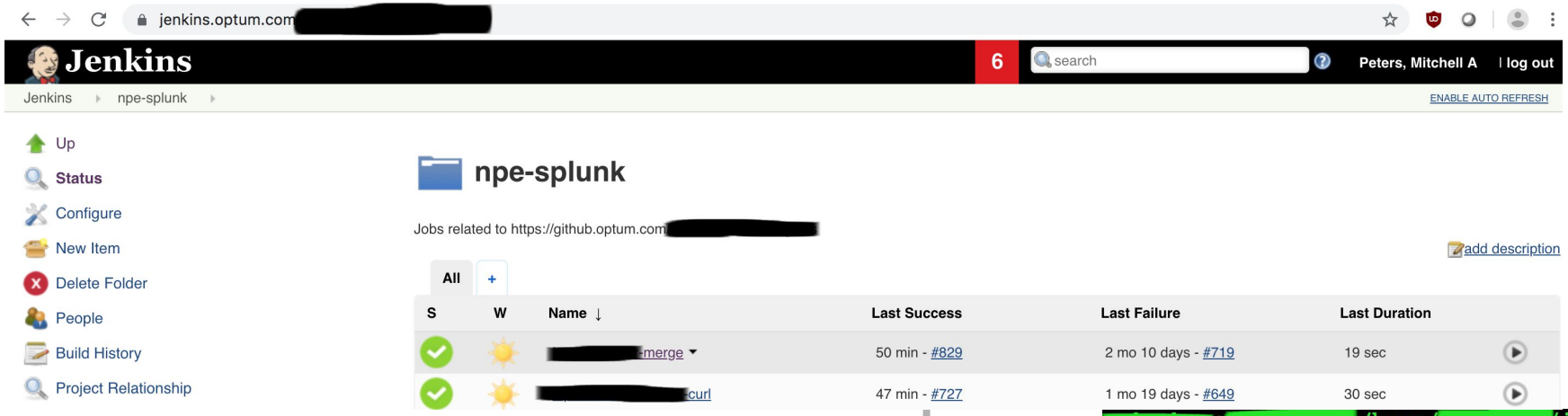
What it Looks Like

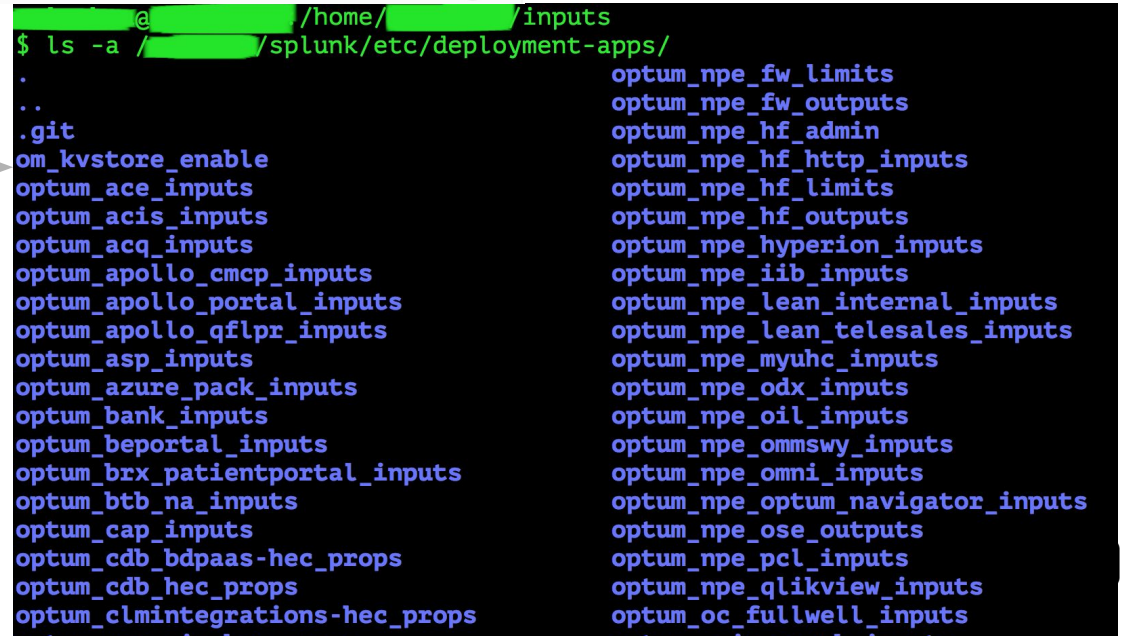
From a GitHub perspective – Repo names blurred





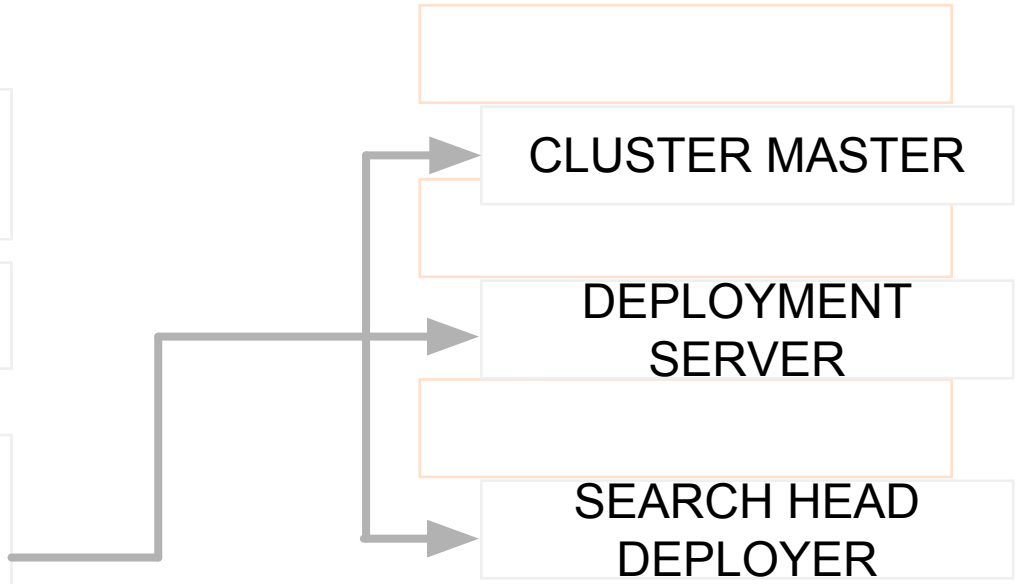
What it Also Looks Like





Technology Flow – Bundle Updates

1. Splunk Admins Create the change in GitHub
2. Jenkins notes there is a change ready
3. Cron job on from our automation host fires off cluster bundle update when it's time to each splunk member



Note: ALL backend configs are held in Github – Splunk admins absolutely never log into ANY splunk hardware to update/make changes

Config Deployments

Why did we set it up this way aka how does it help?

Manages all of the cluster bundle updates for us

Prevents admins from logging into boxes

Cron allows us to keep pushing configs knowing the order of configs will be taken care of (example props before inputs)

Tips: Set your cron jobs off of order of configs and low traffic hours

Q&A

Shelbie Wise | Architect Analyst

Mitchell Peters | Sr IT Architect Analyst

splunk> .conf19

Final Tips

Notes you can save and read later

Key authentication – always push keys when you can. This prevents user accounts from getting locked

Consider using one **NUID** for your forwarder installs – this promotes consistency among hosts

Take advantage of **open source technology**

Use **git/version control** for ALL of and everything you do, this eliminates the need to have access to splunk hardware

.conf19

splunk>

Thank You !

Go to the .conf19 mobile app to

RATE THIS SESSION