Splunk, PCF and ITSI – Supporting PCF with ITSI at Scale

Shubham Jain, Kirk Hanson, Splunk

Forward-Looking Statements

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 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.

.conf19

splunk>

Kirk Hanson

IT Markets Specialist | Splunk

Shubham Jain

Senior Software Engineer| Splunk

Agenda

1. Pivotal Cloud Foundry (PCF) Introduction
2. Splunk Nozzle for PCF Design and Architecture
3. Splunk Nozzle for PCF Installation and Configuration
4. What is ITSI?
5. PCF Module for IT Service Intelligence
6. Happiness with PCF & ITSI
7. Q&A

Splunk & Pivotal Introduction

Overview

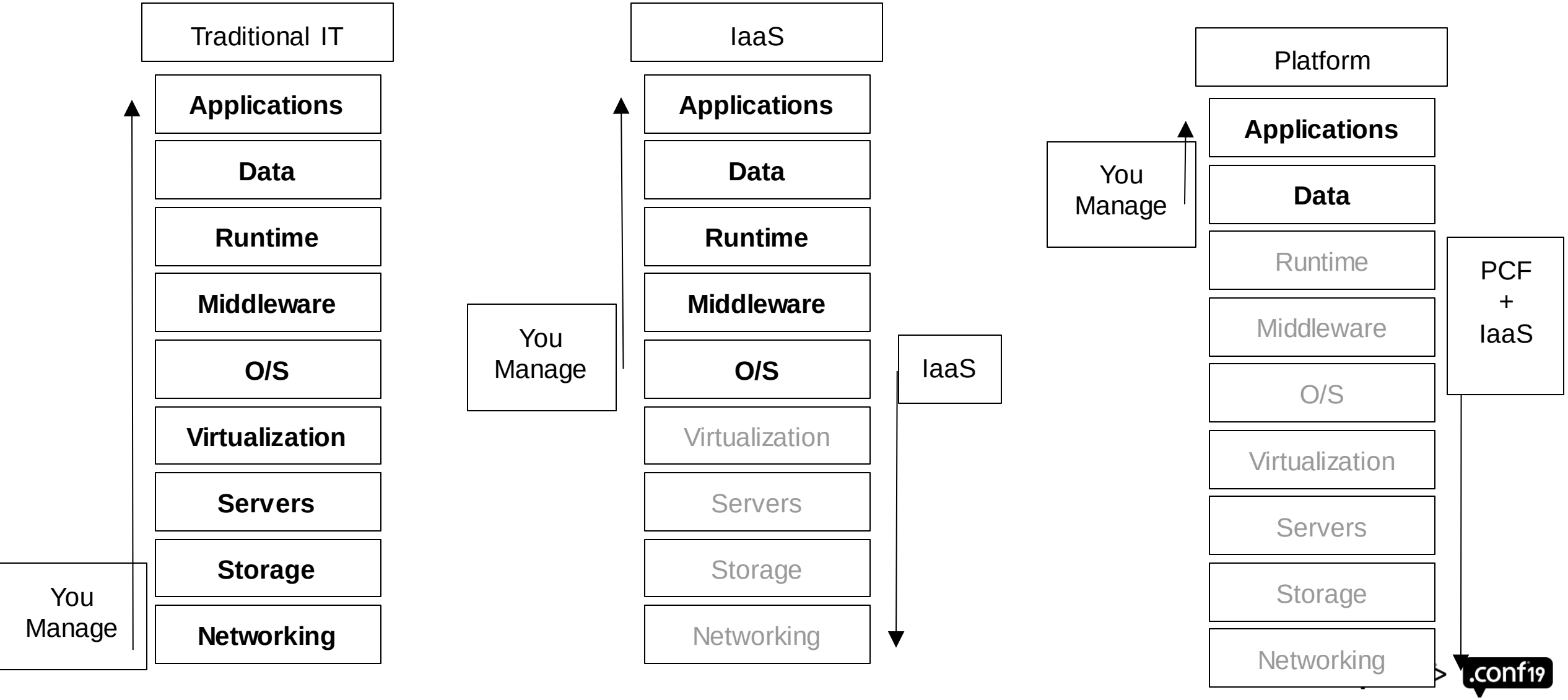
.conf19
splunk>

Pivotal Cloud Foundry

PCF is the proven solution build on top of Cloud Foundry platform for companies seeking software-led, digital transformation which provides:

- Developer Productivity – Accelerate feature delivery.
- Operator Efficiency – 200:1 developer to operator ratio.
- Comprehensive Security – Reduce risk in app portfolio.
- High Availability – Deliver enterprise SLAs at scale.

The Power of the PCF Platform



Splunk Nozzle for PCF

Architecture & Overview

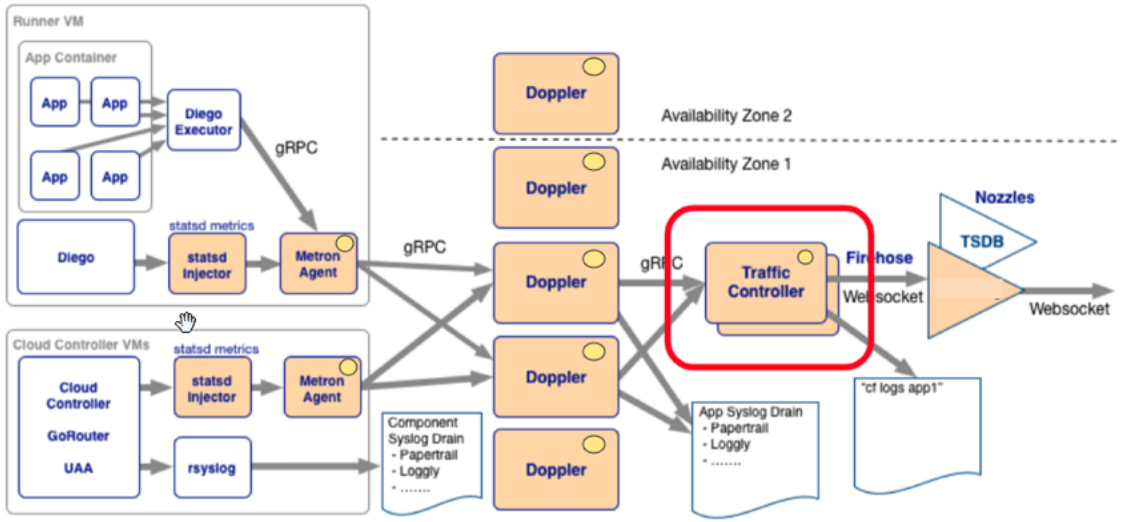
.conf19
splunk>

Nozzle Overview

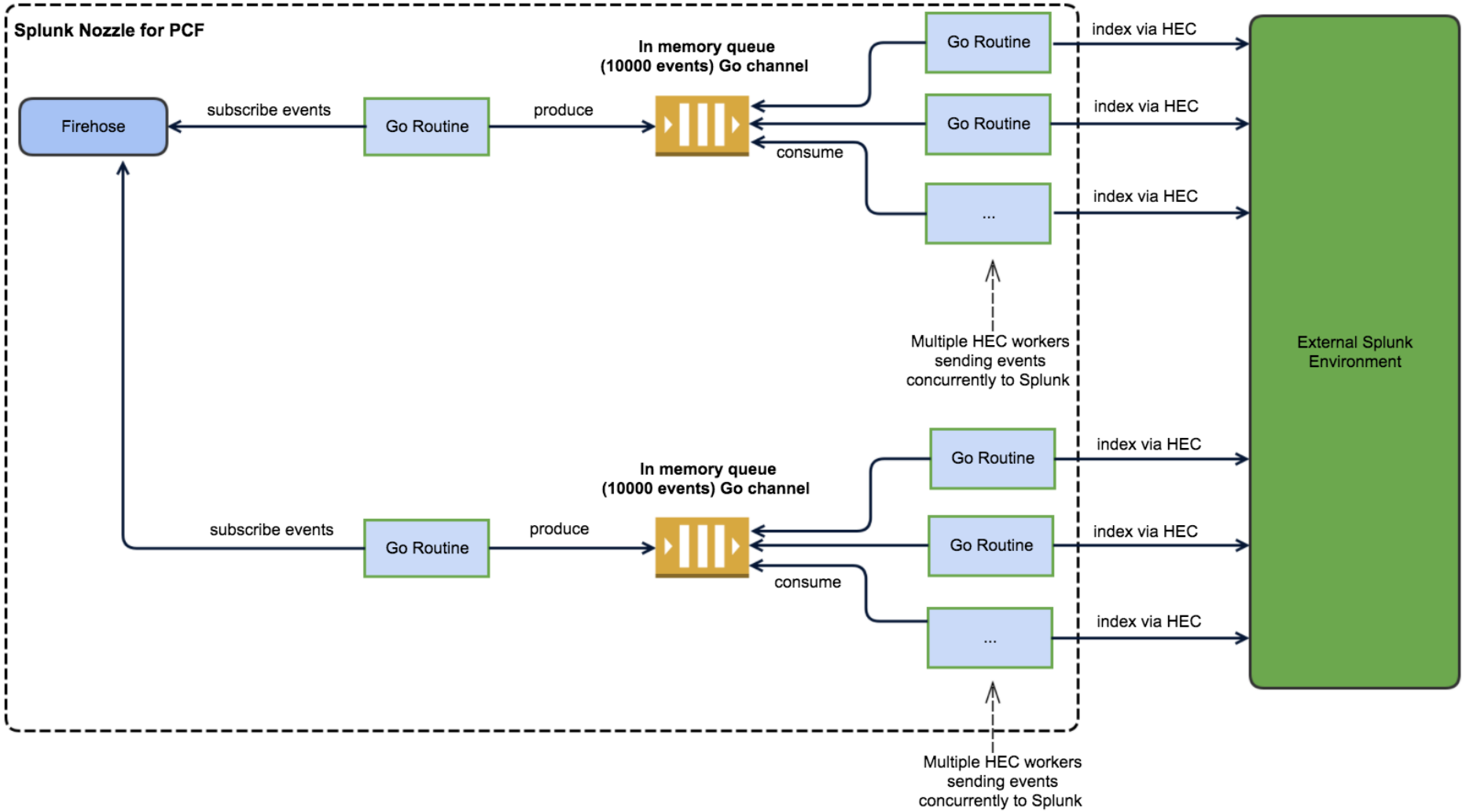
What is a Nozzle?

- A nozzle is a component dedicated to reading and processing data that streams from the Firehose
  - PCF consolidates application logs and platform components' metrics to a PCF component known as Firehose Loggregator.
  - To get events out of PCF and into your Splunk environment, you need a Nozzle that attaches to the Loggregator Firehose.

PCF Loggregator Architecture

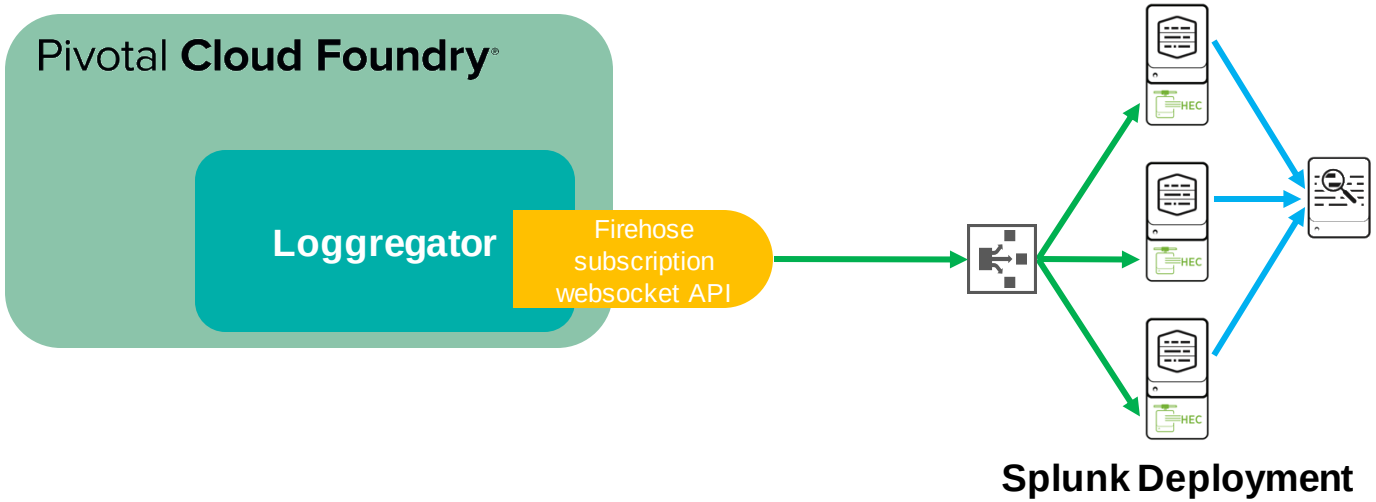


Splunk-built internal components



Pivotal Cloud Foundry integration

- The Splunk Firehose Nozzle for PCF collects events from the PCF **Loggregator** endpoint and streams them to Splunk via HTTP Event Collector
  - High performance and reliability with Nozzle's in-memory queue buffers, and parallel clients to scale out multiple ingestion channels to HEC
  - Simple deployment natively within a PCF environment using a tile, or through CLI
  - Easy scalability of ingest by adding more HEC data collection nodes behind a load balancer

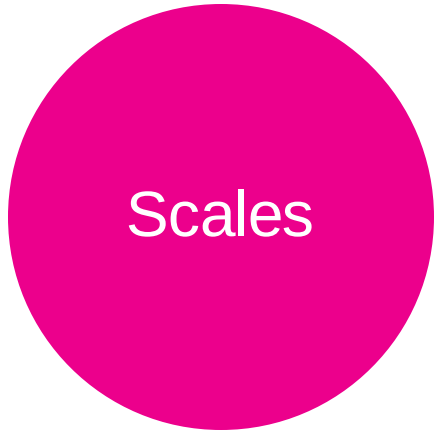


Splunk Nozzle for PCF Features

Setup with out-of-the-box data parsing and enrichment for various PCF event types.

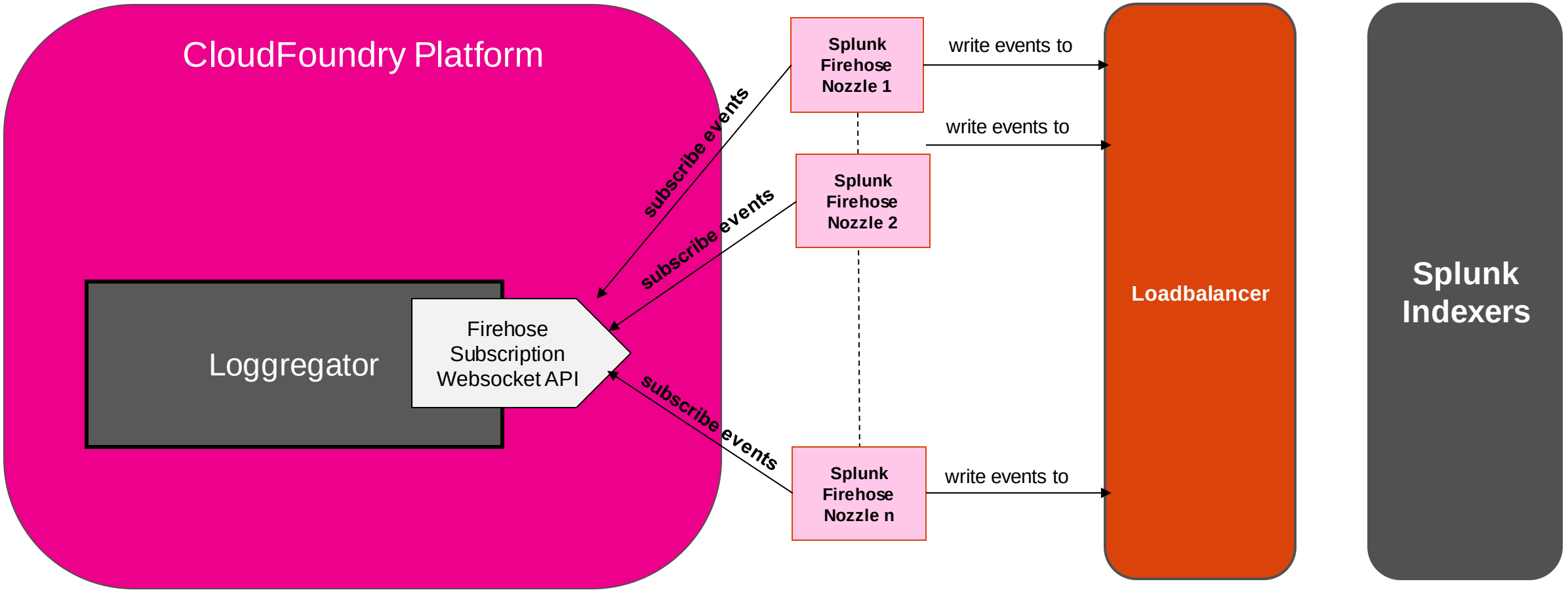
Reliable event delivery by leveraging Splunk's HTTP Event Collector endpoint.

Secure forwarding from PCF into external Splunk environments using user-provided SSL certificates.



Scales out to meet increasing data volume and number of apps.

Splunk Nozzle for PCF Deployment Architecture

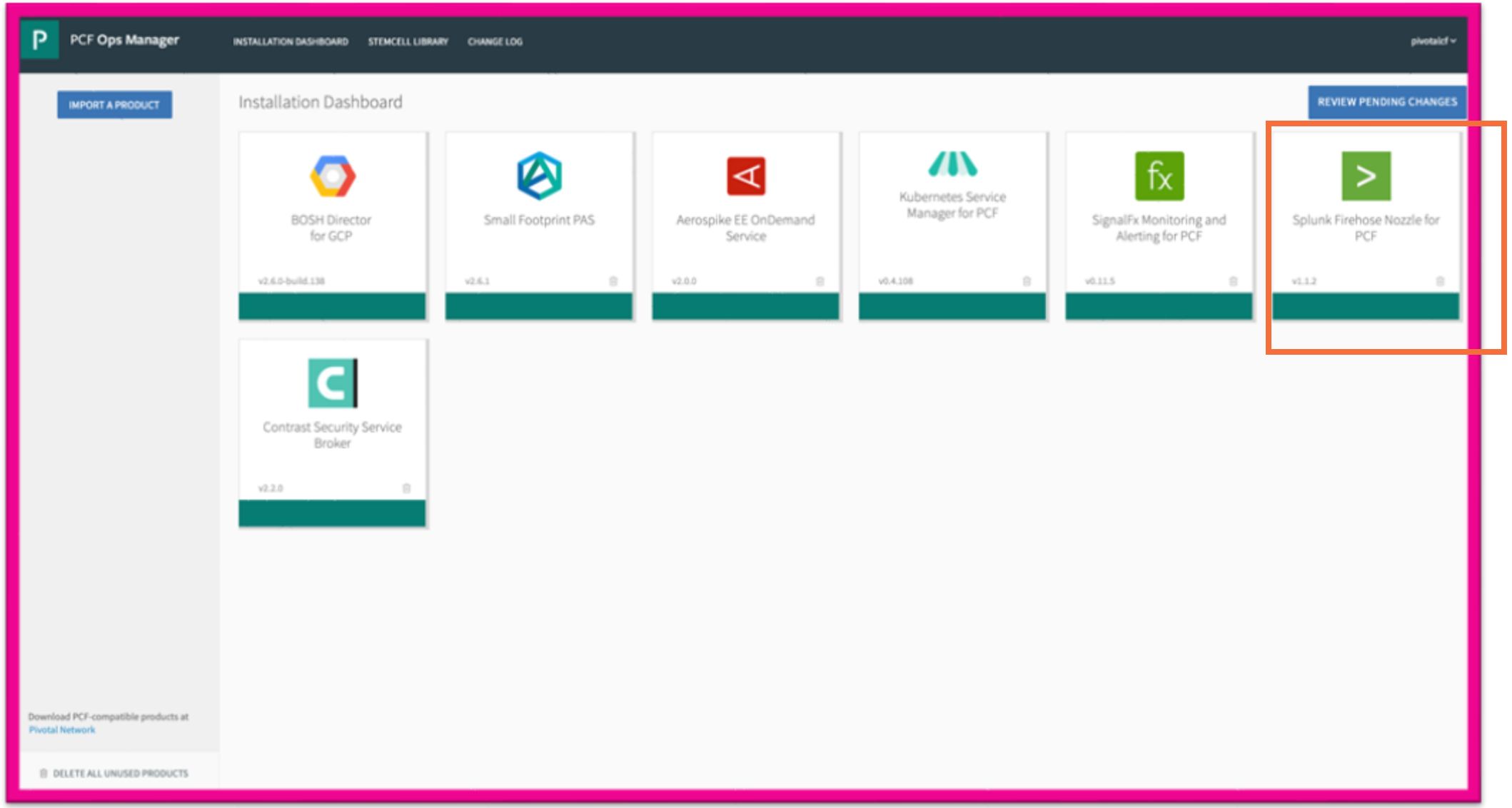


Installation and Deployment

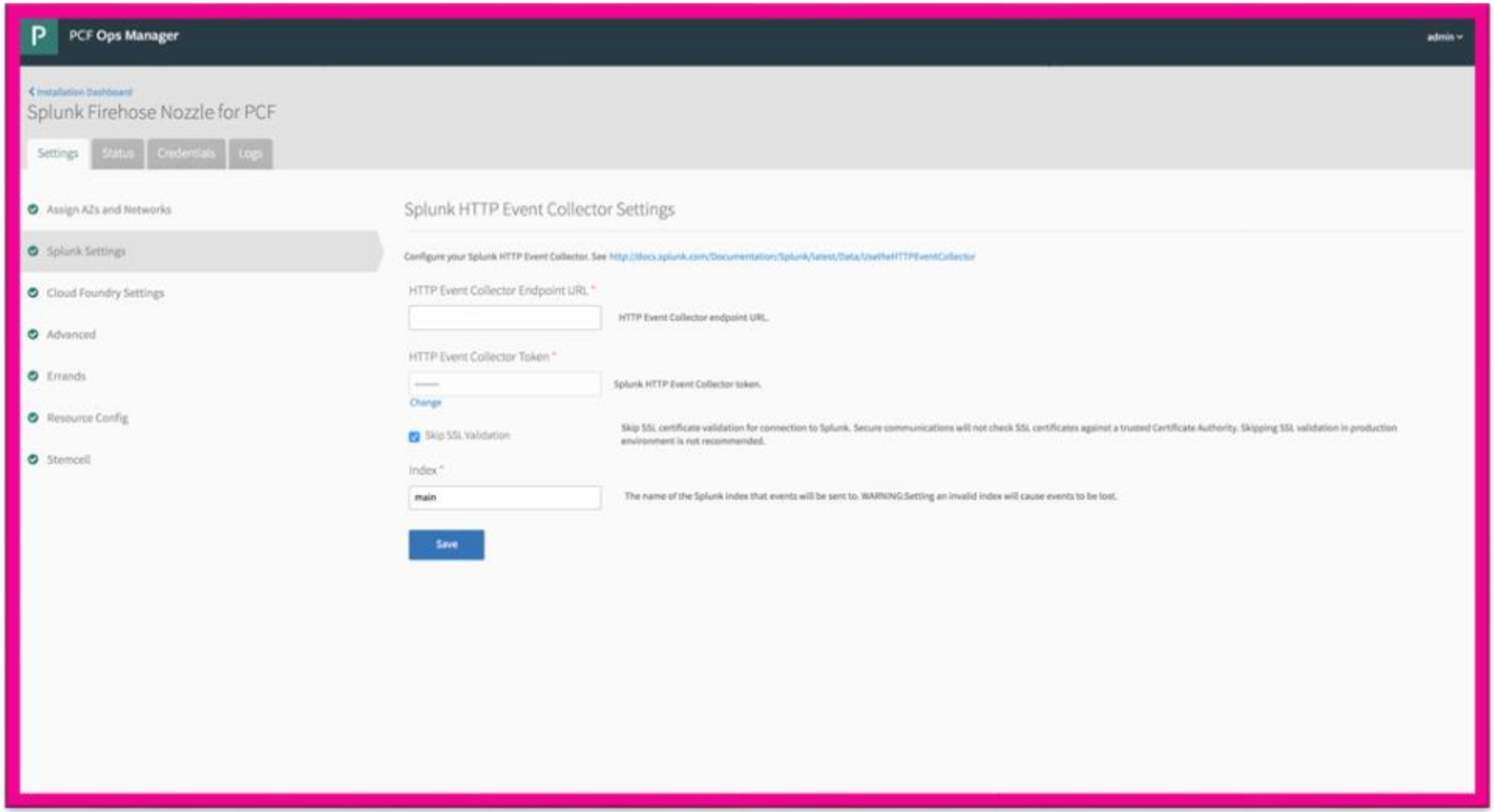
How to install and deploy Splunk Firehose Nozzle in PCF?

- Download and install the Splunk tile from Pivotal Network.
- Navigate to the Ops Manager Installation Dashboard and click Import a Product to upload the product file and add it to your staging area
- Configure the Tile as per requirements
- Apply changes to deploy the nozzle

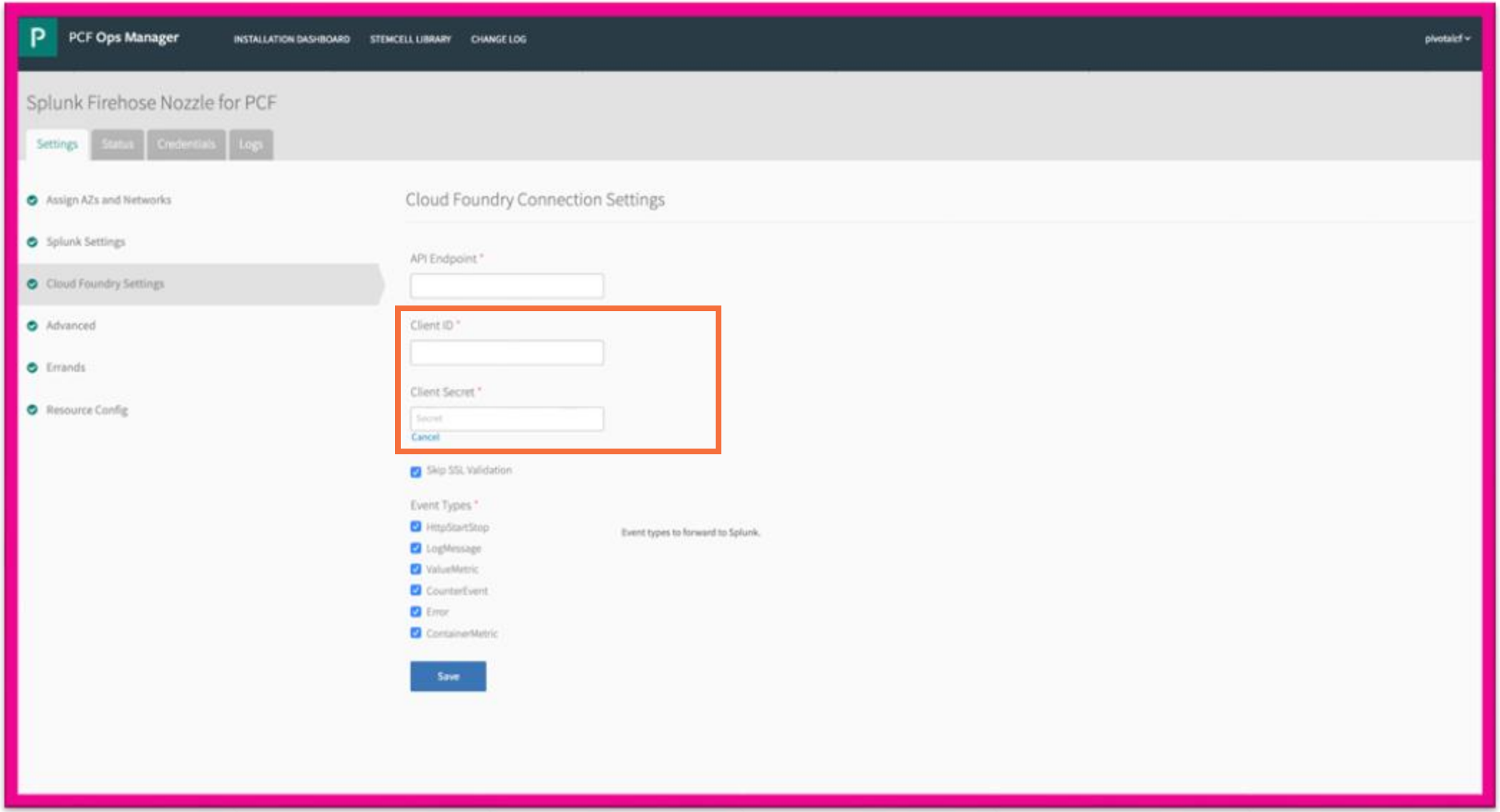
Deployment through Tile



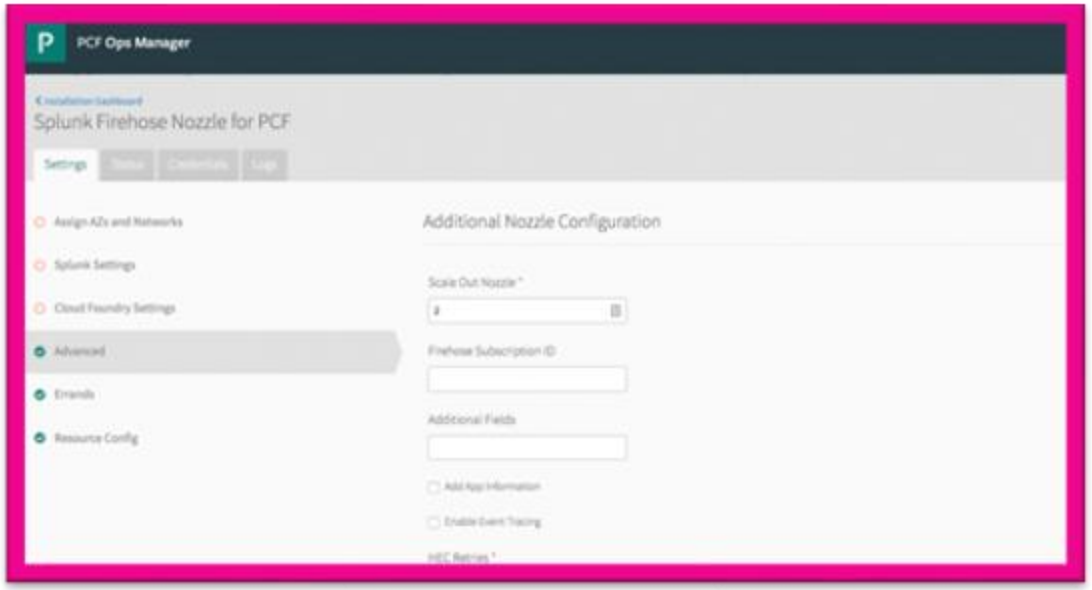
Splunk Configuration



Cloud Foundry Configuration



Advanced Configuration



Complete the fields as follows:

- **Scale Out Nozzle**: Scale out Splunk Nozzle. Splunk recommends running two or more nozzles for high availability.
- **Firehose Subscription ID**: The Loggregator does a round-robin of events across connections having the same ID. An operator would only need to change this if some other connection is also using the default value.
- **Additional Fields**: Arbitrary set of key:value pairs in the form key1:value1,key2:value2,key3:value3 that do not occur in event payload itself. These fields are indexed with every event payload, and are used as metadata for indexing and searching. One situation in which this might be useful is differentiating logs from multiple foundations going into the same Splunk Enterprise index.
- **Add App Information**: When true, the nozzle calls back to the Cloud Foundry API and queries events that have an app GUID to add additional information, such as app name, space name, space GUID, organization name, organization GUID. This causes a one-time load increase on the API machines proportional the number of running apps, because each Nozzle populates a cache of app metadata.
- **Enable Event Tracing**: Enables event trace logging. Splunk events now contain a UUID, Splunk Nozzle Event Counts, and a Subscription-ID for Splunk correlation searches.
- **HEC Retries**: The retry count for sending events to the Splunk platform. Events not successfully sent after this number of retries will be dropped, causing data loss.
- **HEC Batch Size**: The number of events per batch sent to Splunk HTTP Event Collector.
- **HEC Workers**: The number of concurrent workers sending data to Splunk HTTP Event Collector. Scale this number to your Splunk platform data collection capacity accordingly.
- **Consumer Queue Size**: The internal consumer queue buffer size. Events will be sent to your Splunk platform after queue is full.
- **Flush Interval**: Time interval (in seconds) for flushing queue to the Splunk platform regardless of Consumer Queue Size. Prevents stale events in low throughput systems.
- **Missing App Cache Invalidate TTL**: Time interval (in seconds) between refreshing the missing app info cache. Set to 0 to maintain cache until nozzle restart.
- **App Cache Invalidate TTL**: Time interval (in seconds) between refreshing the app info local cache. Set to 0 to only populate the cache during startup or restart of the nozzle.
- **App Limits**: The number of apps for which metadata is gathered when refreshing the app metadata cache (order based on app creation date). Set to 0 to remove limit.
- **Ignore Missing App**: Enables event trace logging. Do not trigger an app metadata refresh when encountering data from an app without info in the app metadata cache.

Brief ITSI Introduction

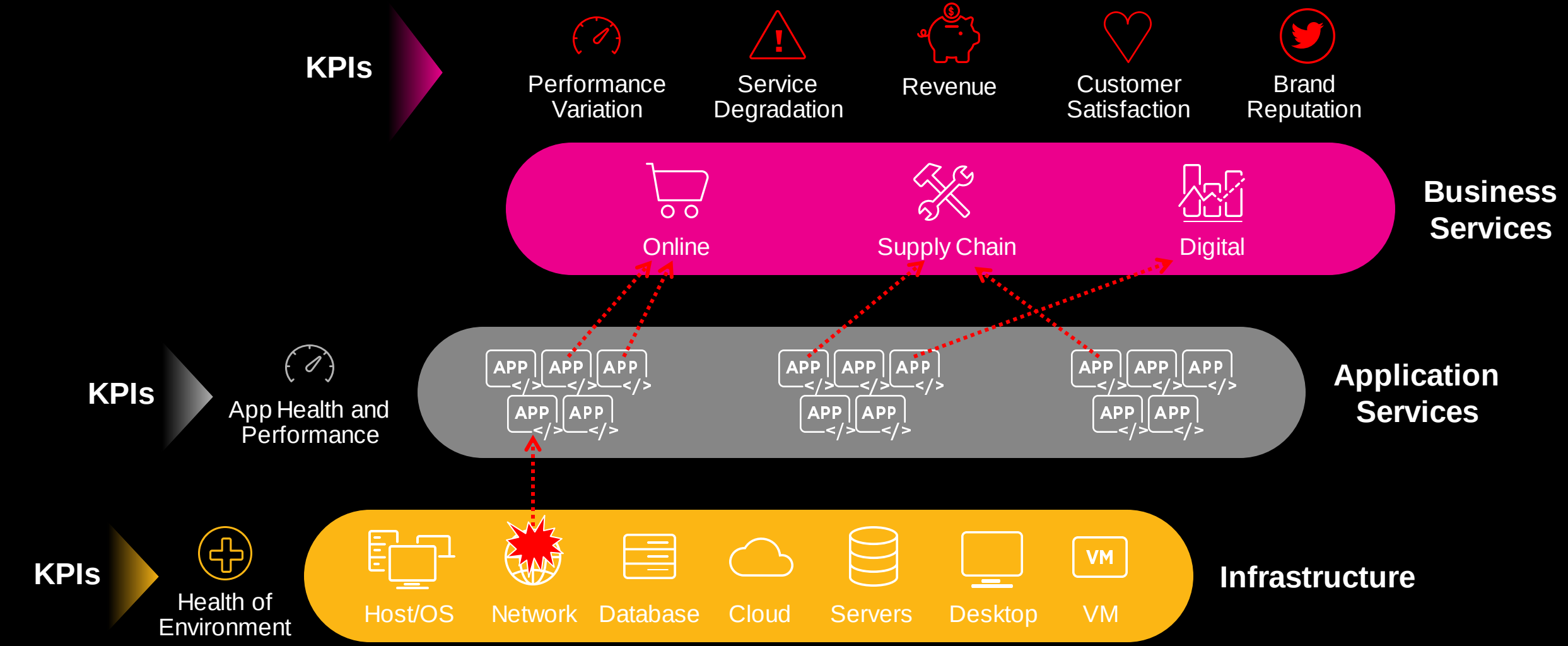
Module design, overview and walkthrough

.conf19
splunk>

IT Struggles to Identify, Investigate and Resolve Critical Service Issues

- Guesswork
- Reactive
- Unproductive

IT Struggles to Identify, Investigate, and Resolve Critical Issues



Splunk IT Service Intelligence (ITSI)

Predictive analytics for real-time insights, simplified operations and root-cause isolation

Predict and Prevent Outages
while reducing event noise & MTTR

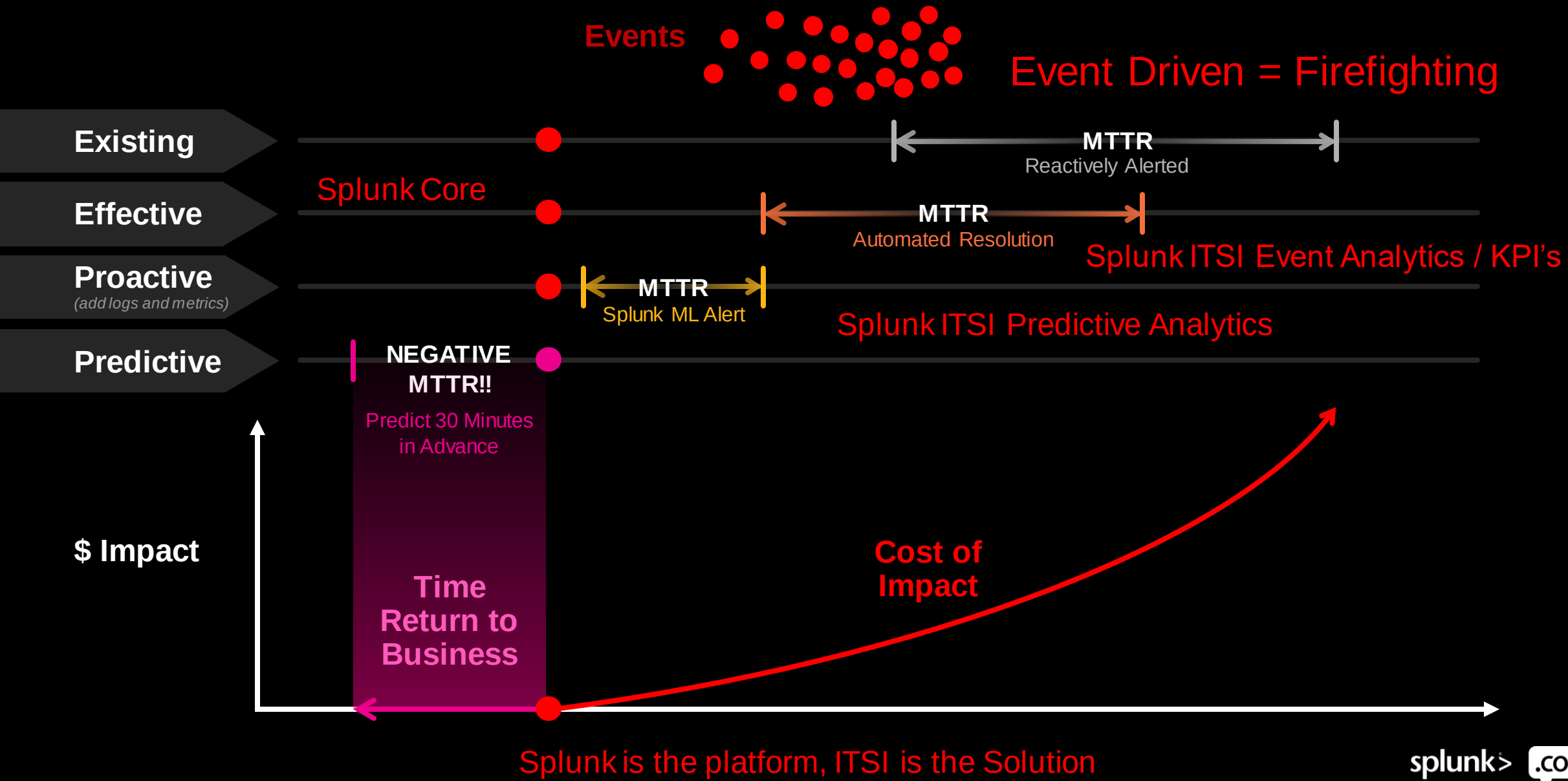
Create a 360-degree View
of real-time insights across all business & IT services

Trust the Splunk Platform
for scalability and versatility with artificial intelligence (AI) at its core

PREDICT
imminent outages
20-30
MINUTES
in advance

REDUCE
event storms
& noise by
+95%

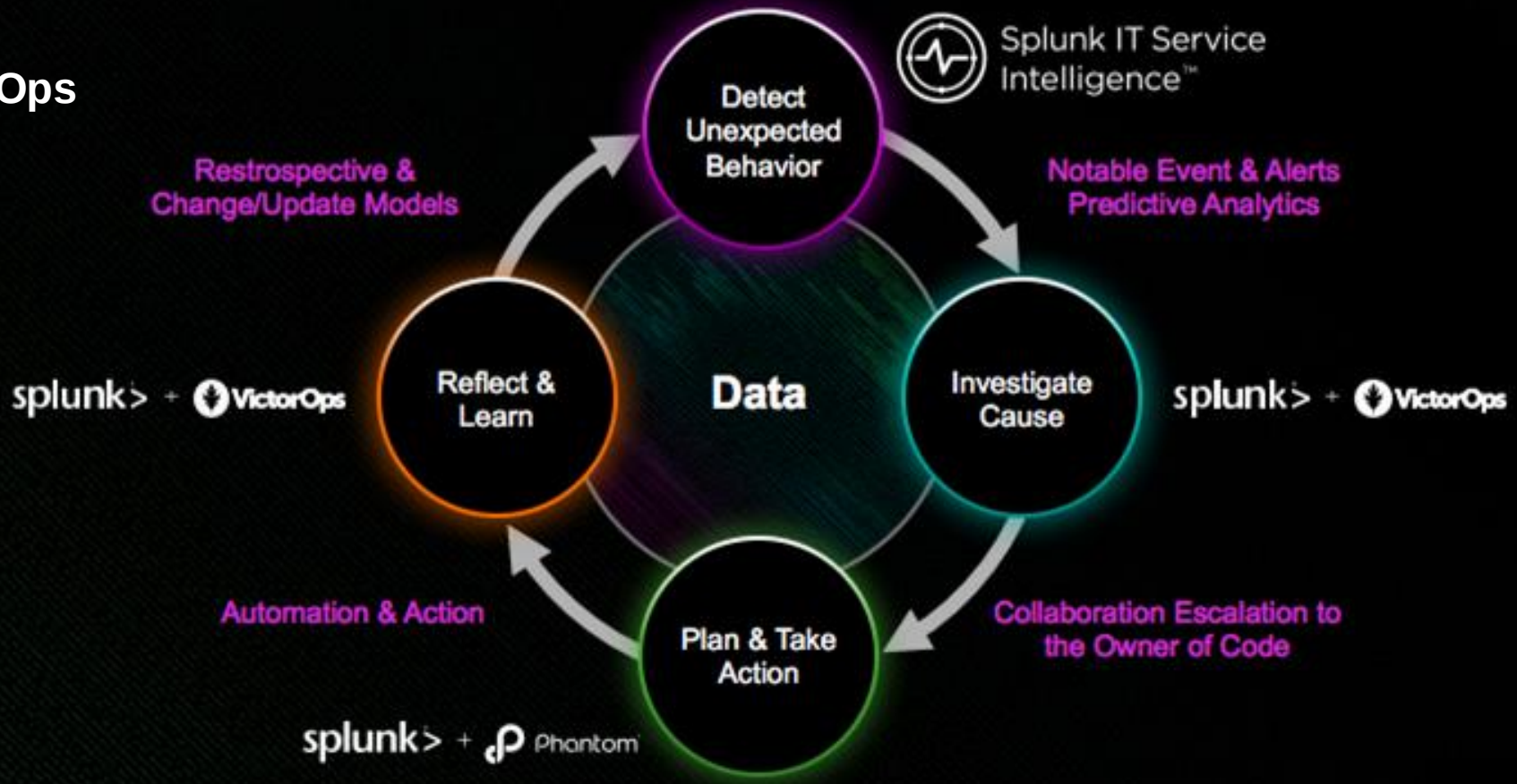
Predict and Prevent Operational Issues with AI



splunk> .conf19

Splunk for IT Vision

Shift to NewOps



Monitoring | Collaboration | Automation

splunk>

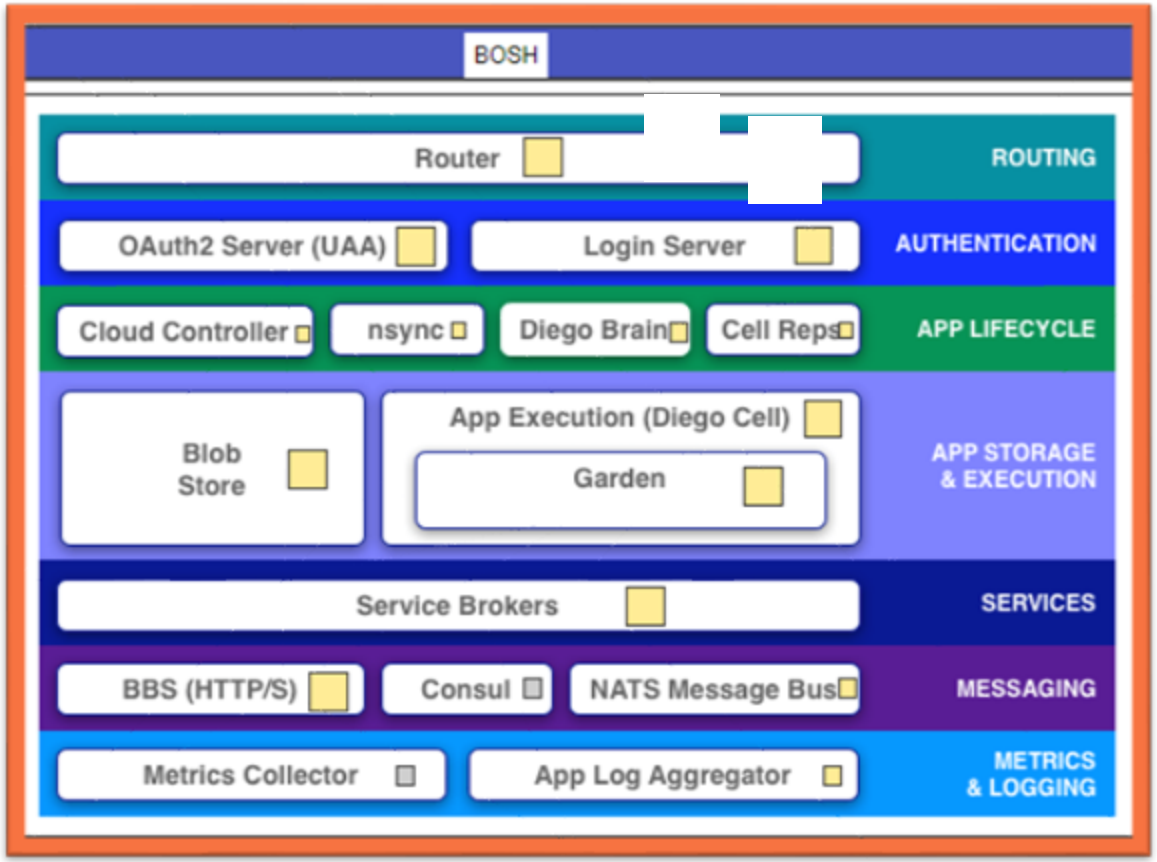
Splunk Nozzle PCF ITSI ContentPack-Walkthrough

Module design, overview and walkthrough

.conf19
splunk>

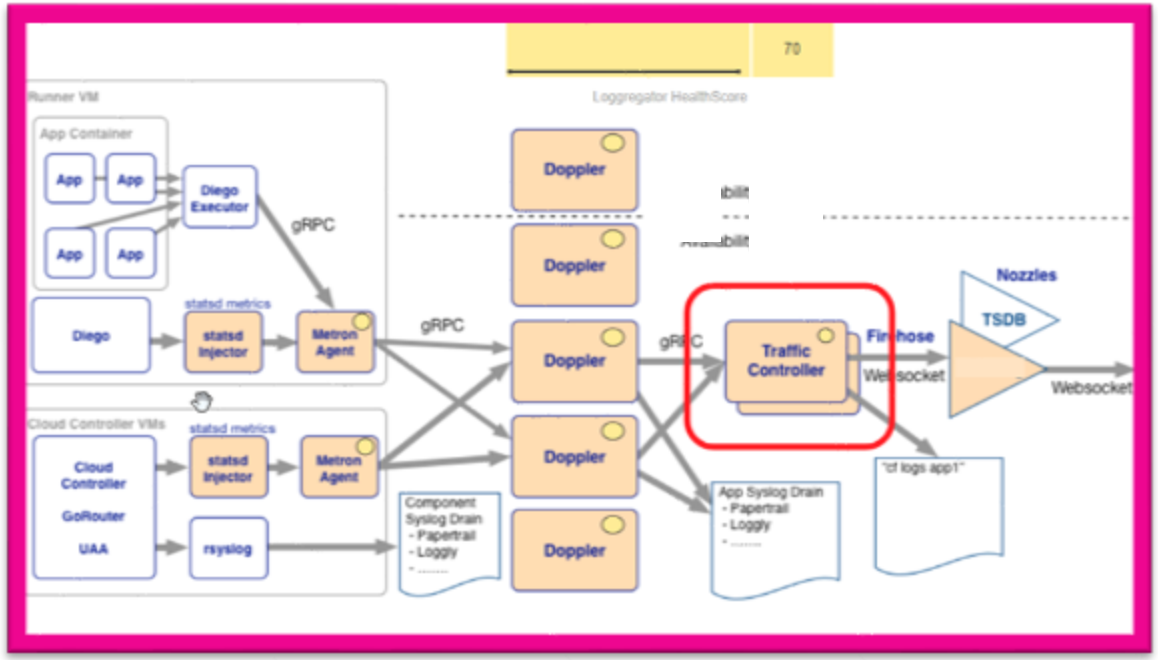
CF Architecture

- A glass table is designed to provide a high-level view
- Taken right from Pivotal's architecture
- Live KPI's from Service Health Score's are embedded in the Glass table allowing for an actionable interface



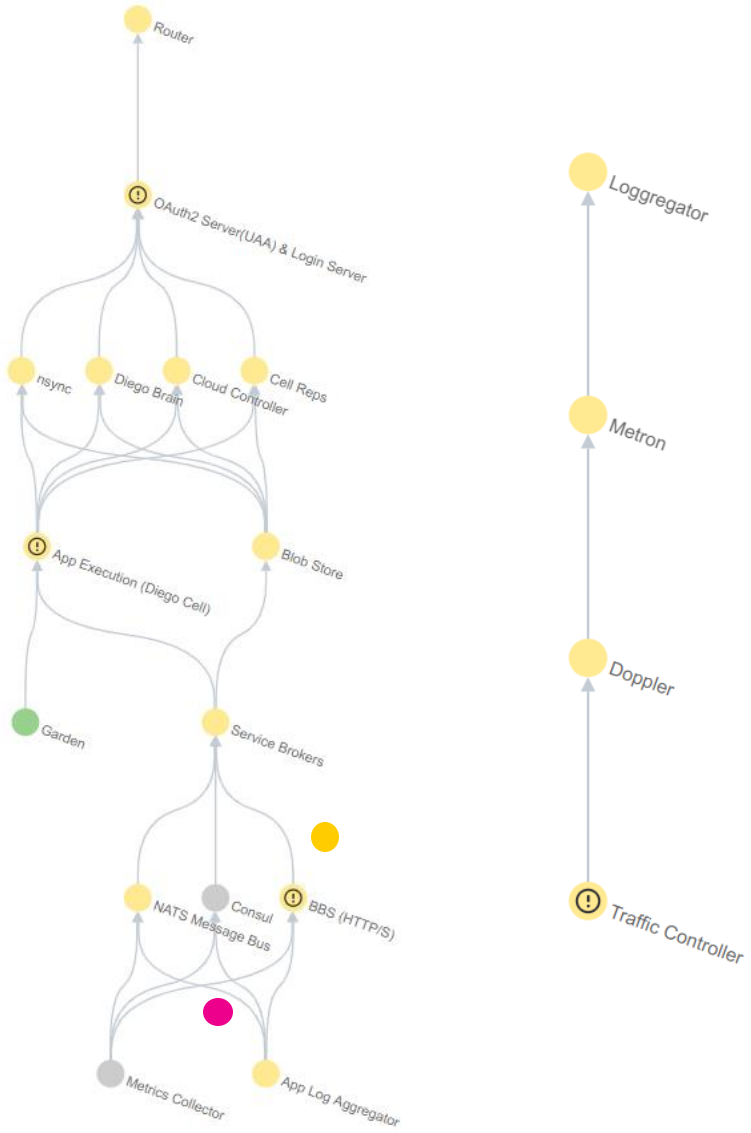
Loggregator GT

- Displays overall Loggregator Health
- Based on the Loggregator components that are logged in the firehose



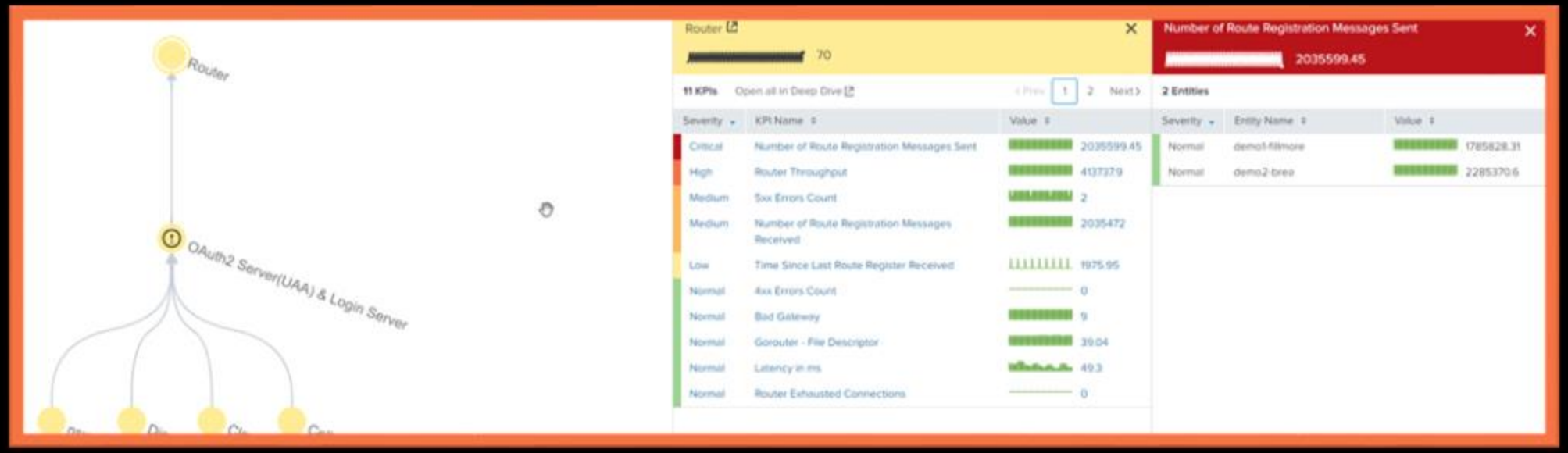
Service Analyzer

- Hierarchy based on overall Service Architecture
- Live data used to build the service topology
- Best practice KPI thresholds used when created



Service Analyzer

- Broken down by Foundation
- Many KPI's based on best practices
- Quickly understand what component is having trouble by foundation

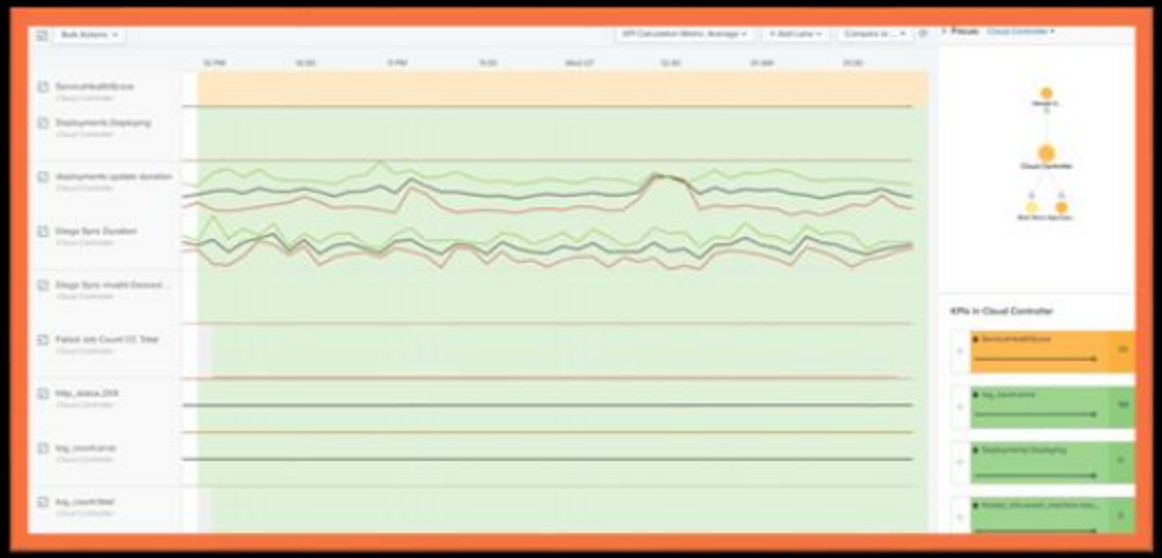


Service Deep Dive

- Next Level of granularity that is split by Foundation to compare which foundation might be facing issue
- Swim lanes that give a clear picture of not only the health but the trend

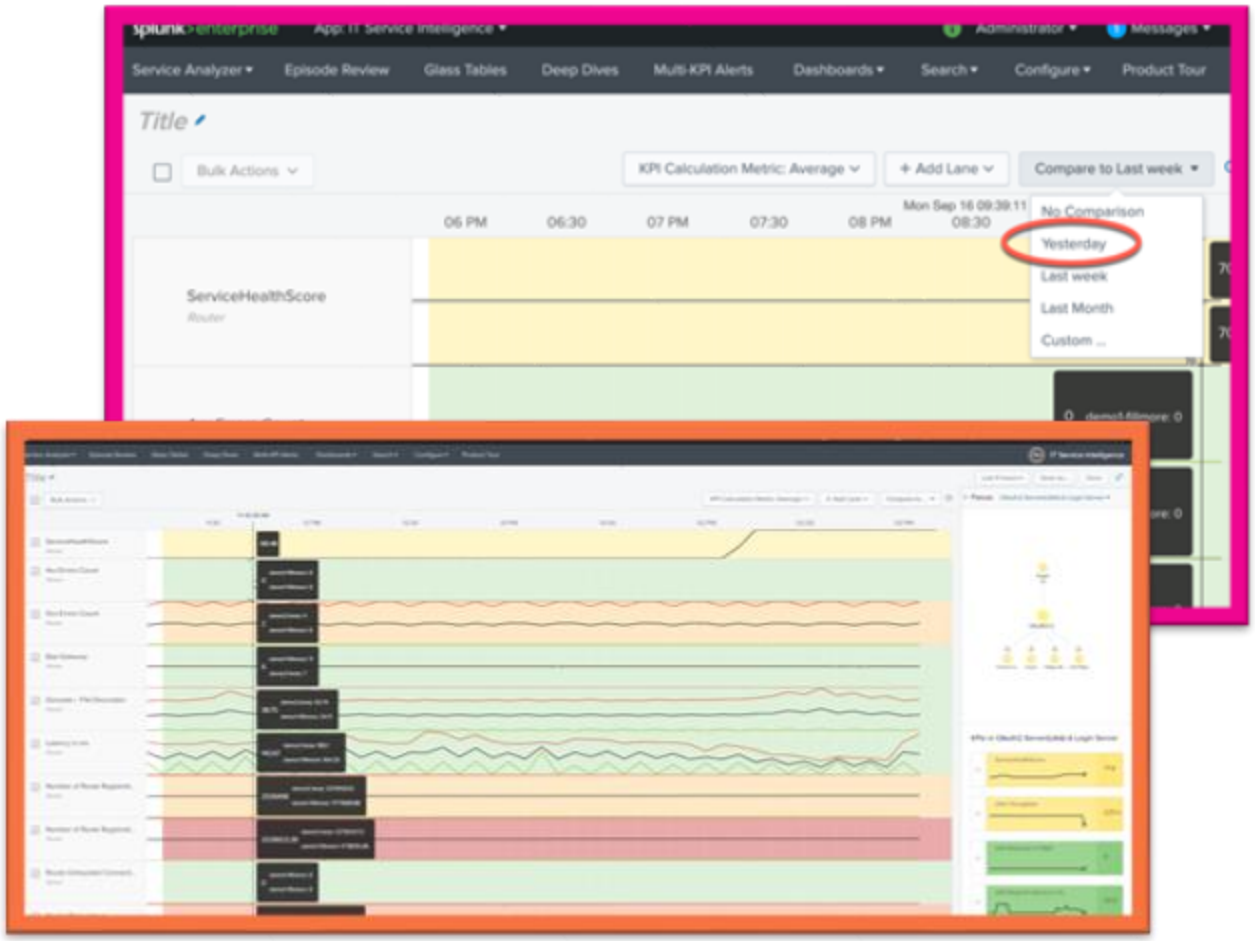
Service Deep Dive

- Next Level of granularity that is split by Foundation to compare which foundation might be facing issue
- Swim lanes that give a clear picture of not only the health but the trend



Service Deep Dive

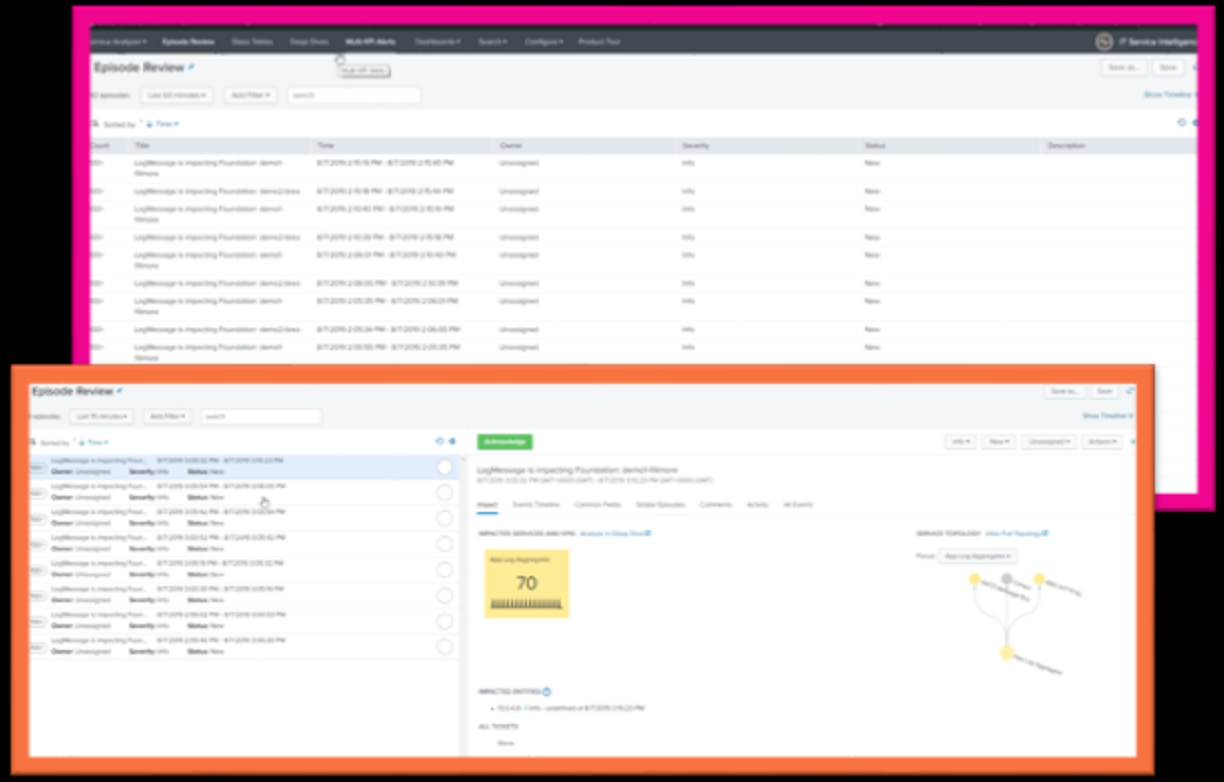
- Next Level of granularity that is split by Foundation to compare which foundation might be facing issue
- Swim lanes that give a clear picture of not only the health but the trend
- Allows for Entity Overlay to compare how the service is behaving on each entity
- Time Comparison allows for easy trend analysis across the entire service
- Get to Root Cause quicker!



Episode Review

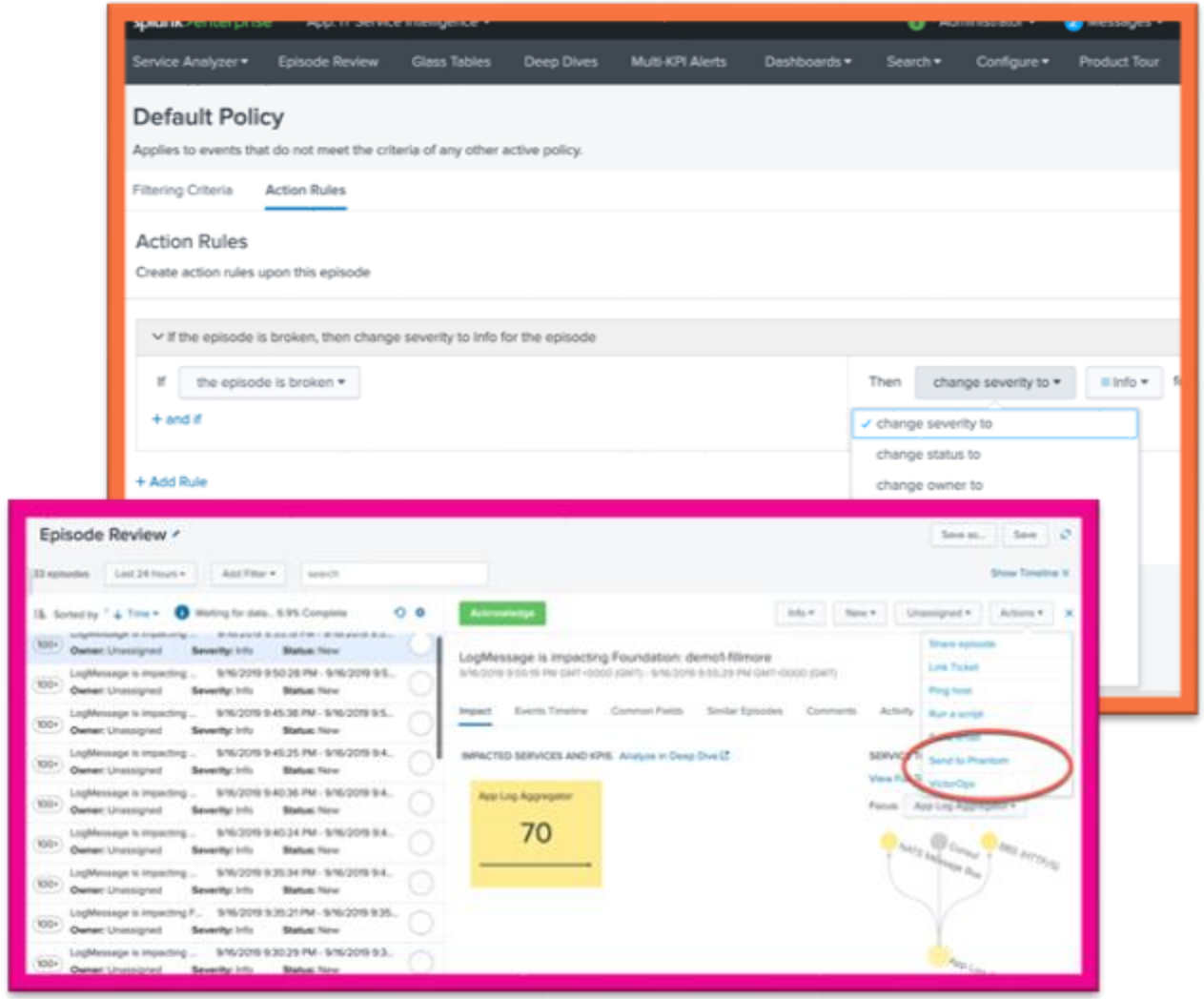
Event Analytics

- Provides for consolidated view of all alerts/alarms generated by correlation rule and NEAP
- Gives Health Context and Detailed understanding of what



Episode Review

- Provides for consolidated view of all alerts/alarms generated by correlation rule and NEAP
- Gives Health Context and Detailed understanding of what is wrong
- Take action! Create an automated remediation or alert the right team
- Automate those actions based the Aggregation policy

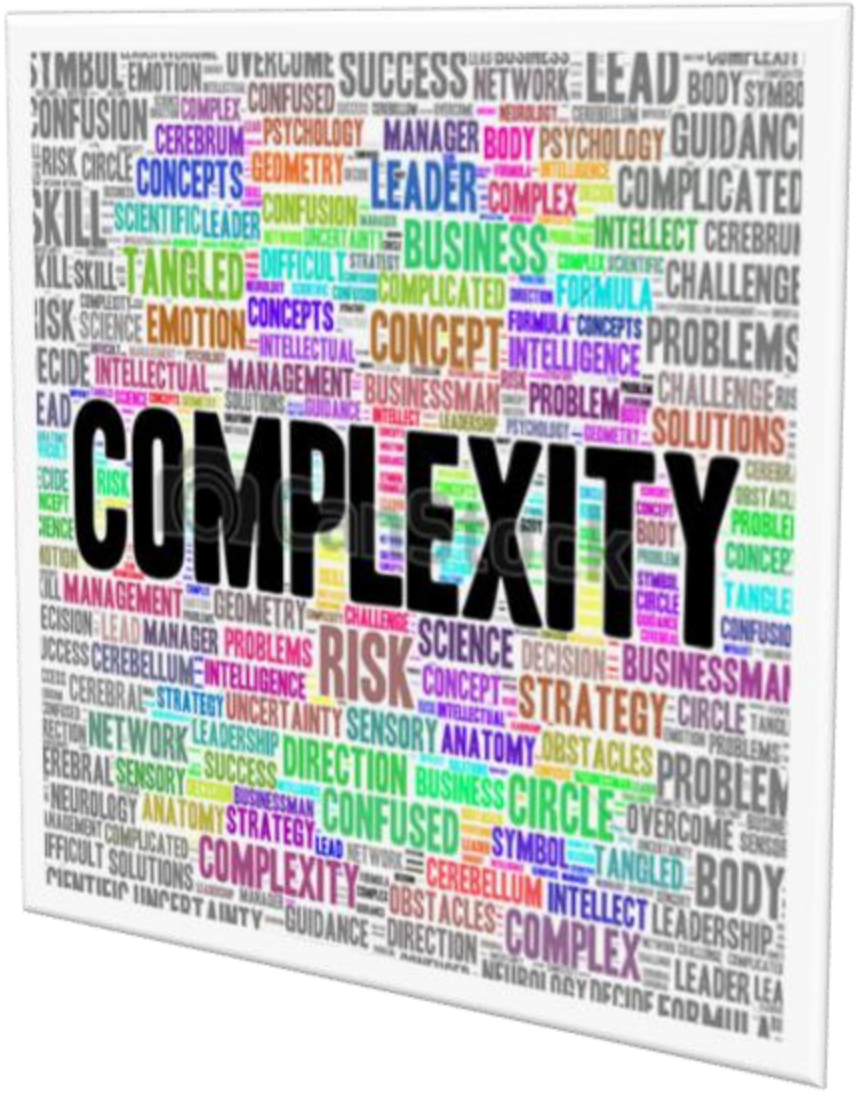


Benefits

.conf19
splunk>

BP (Before PCF)

- No elastic scaling
  - Hard to scale without
- Development time too long
- MTTR too high
- Development cost too high
- Two silo-ed development environments
- No way to have any insight past last 5 minutes



Why PCF?

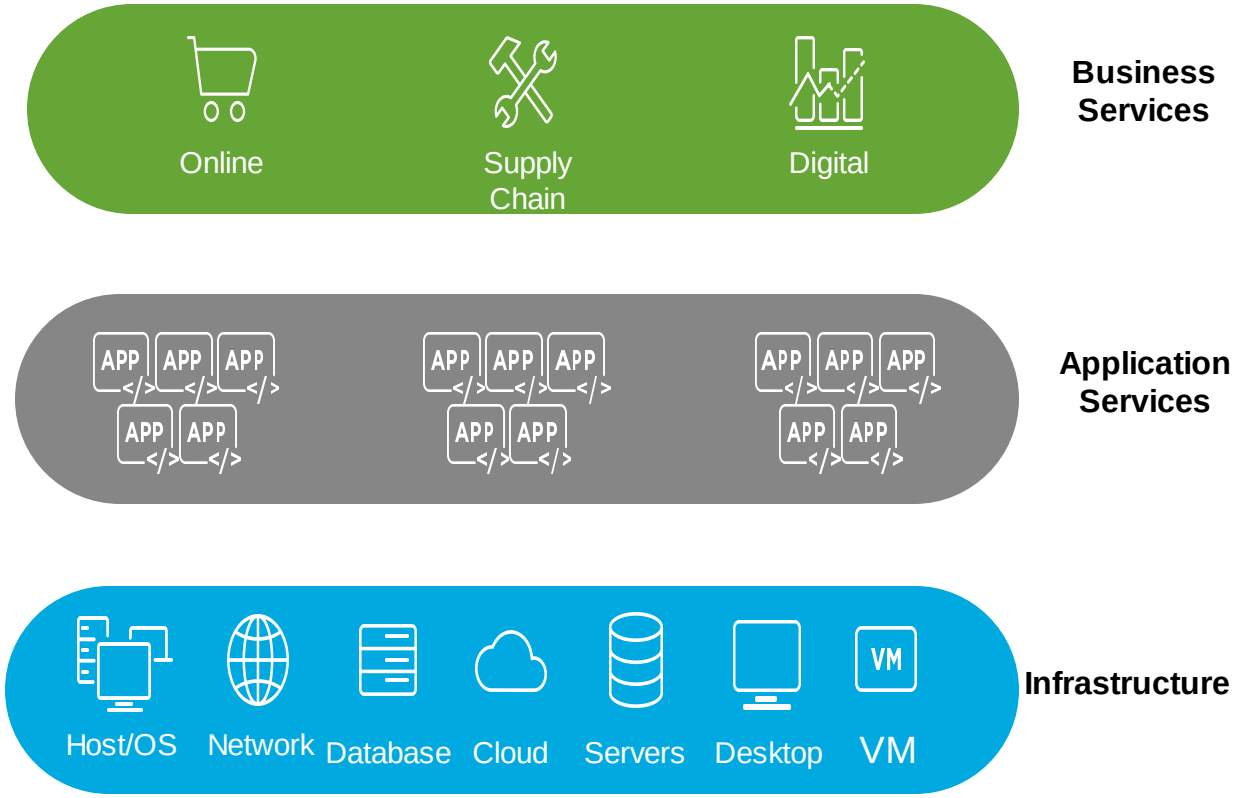
- Elastically Scale as the business requires
- Quick time to deploy
- Standardized deployment and testing
- Increased Operator efficiency
- Highly available architecture

Splunk + PCF

- Splunk is already used for monitoring
- Splunk is the logical choice
- Will use to help achieve 4x9's
- PCF is the future platform for development
- App will play a critical role in that

PCF + Splunk ITSI

- The main goal is a bottom up supportable process
- Not only inclusive of Infrastructure but App & Business Services
- A single source of truth both for development experience but operations as well
- A true devops monitoring platform



Roadmap and Future Plans

Q&A

.conf19

splunk>

splunk>

Thank You!