Machine Learning and Anomaly Detection in Splunk IT Service Intelligence

Alex Cruise

Sr. Dev. Manager/Architect, Splunk

Fred Zhang

Sr. Data Scientist, Splunk

.conf2016

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

- Introductions/History
- Axioms – Problem Domain
- Axioms – Solution Domain
- Time Series Feature Engineering
- Spatial vs. Temporal Analysis
- Other Approaches
- MAD Service Engineering
- ITSI Context

Introductions/History

- Key team members
  - Shang
  - Mihai
  - Jacob
  - Iman
  - Touf
- Presenters
  - Fred – Data scientist
  - Alex – Architect/Dev Manager

Axioms – Problem Domain

- THE UNIVERSE OF DATA

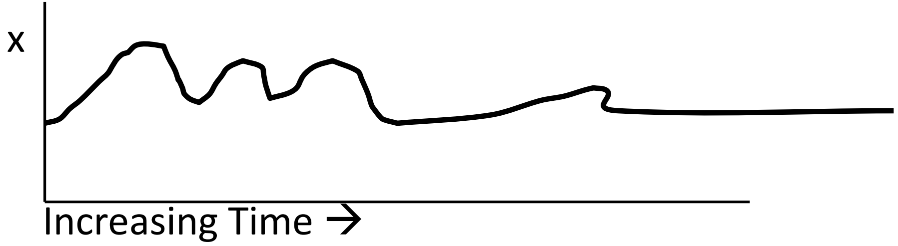
Axioms – Problem Domain

- THE UNIVERSE OF DATA

Axioms – Problem Domain

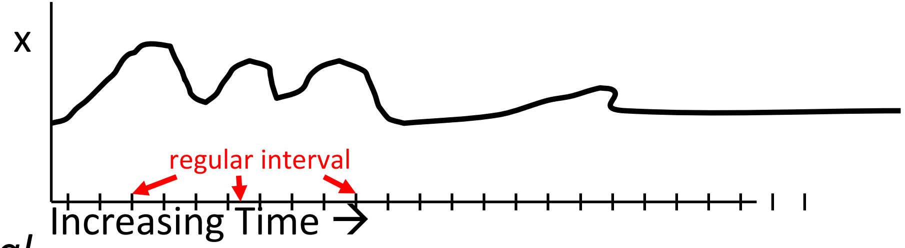
- ***Detecting anomalies*** in this narrow subset of the universe of data:
- Time series

  Numeric variables that change over time



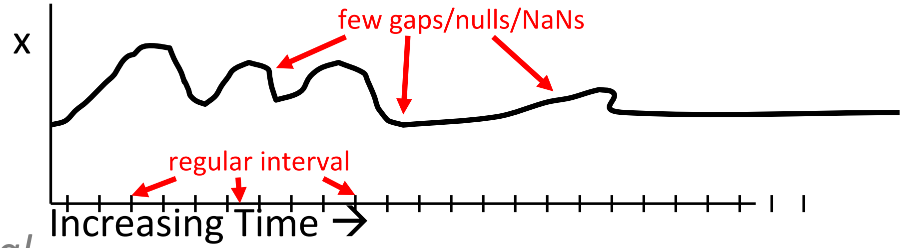
Axioms – Problem Domain

- ***Detecting anomalies*** in this narrow subset of the universe of data:
- Time series
  Numeric variables that change over time
- **Regular** time series
  The new values arrive *on a regular interval (e.g. every five seconds)*



Axioms – Problem Domain

- ***Detecting anomalies*** in this narrow subset of the universe of data:
- Time series
  Numeric variables that change over time



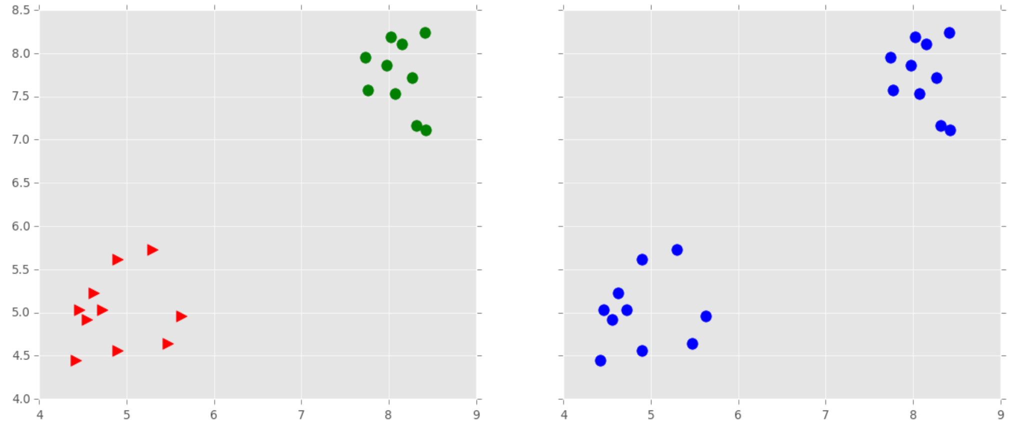
- **Regular** time series
  The new values arrive *on a regular interval (e.g. every five seconds)*
- **Dense**, Regular time series
  New values are *fairly likely* to arrive and *not be null*

Axioms – Solution Domain

- Unsupervised
- Non-Parametric
- Robust
- Streaming
- Adaptive
- Domain-agnostic

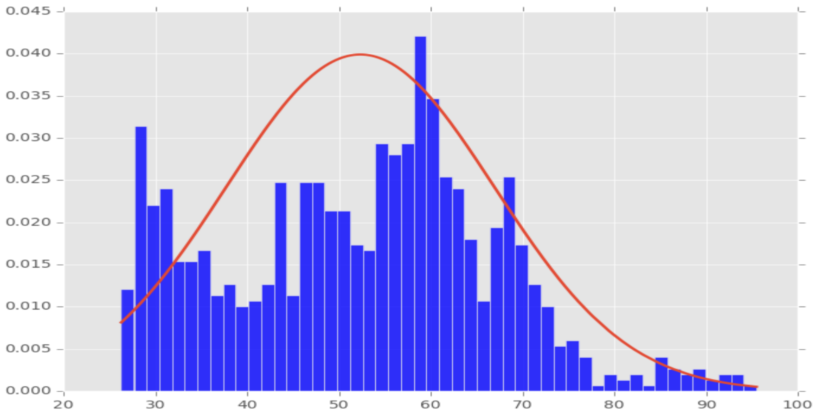
Axioms – Solution Domain

- Unsupervised
  - No labelled anomalies
  - What's normal is learned from observing the data itself, not defined by an expert
- Non-Parametric
- Robust
- Streaming
- Adaptive
- Domain-agnostic



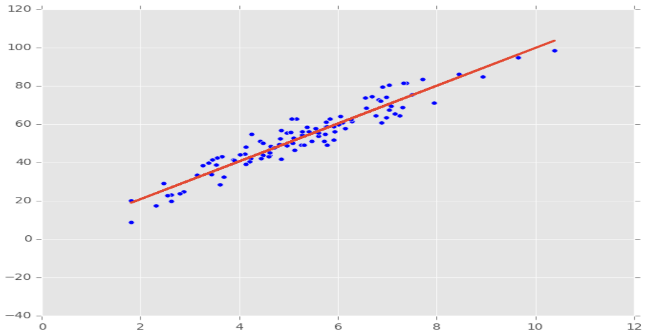
Axioms – Solution Domain

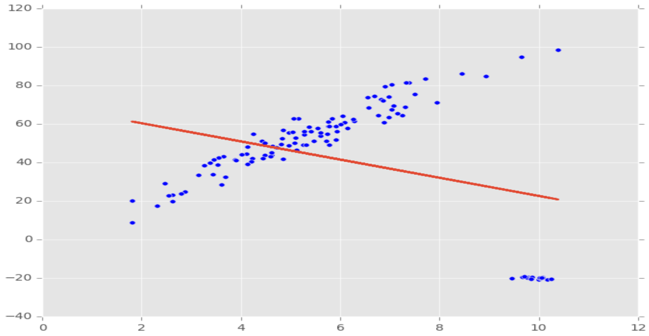
- Unsupervised
- Non-Parametric
  - We make no assumptions about the probability distribution of the values (e.g. Gaussian or stationary)
- Robust
- Streaming
- Adaptive
- Domain-agnostic



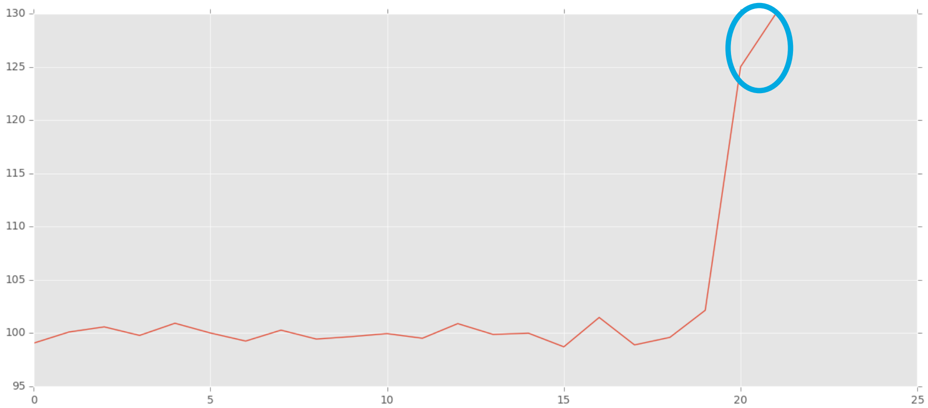
Axioms – Solution Domain

- Unsupervised
- Non-Parametric
- Robust
  - Outliers are detected as anomalies, but don't cause distortions in our expectations
- Streaming
- Adaptive
- Domain-agnostic



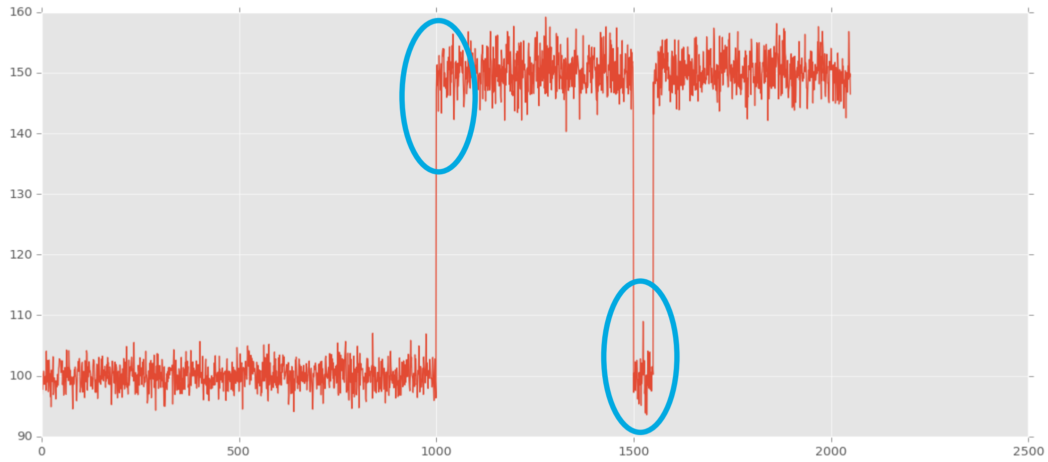


Axioms – Solution Domain



- Unsupervised
- Non-Parametric
- Robust
- Streaming
  - No separate training/test periods
  - Anomalies are detected and reported in (near-) real time
- Adaptive
- Domain-agnostic

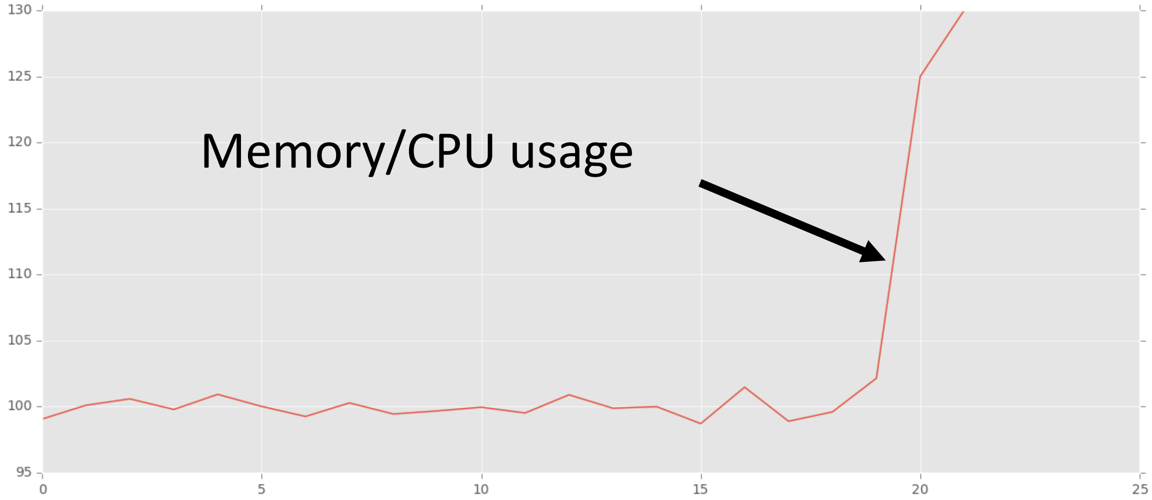
Axioms – Solution Domain



- Unsupervised
- Non-Parametric
- Robust
- Streaming
- Adaptive
  - No static thresholds, discover normal behaviour patterns automatically
  - Adapt to behavioral changes without end-user feedback
  - What was normal last week might be worrisome today
- Domain-agnostic

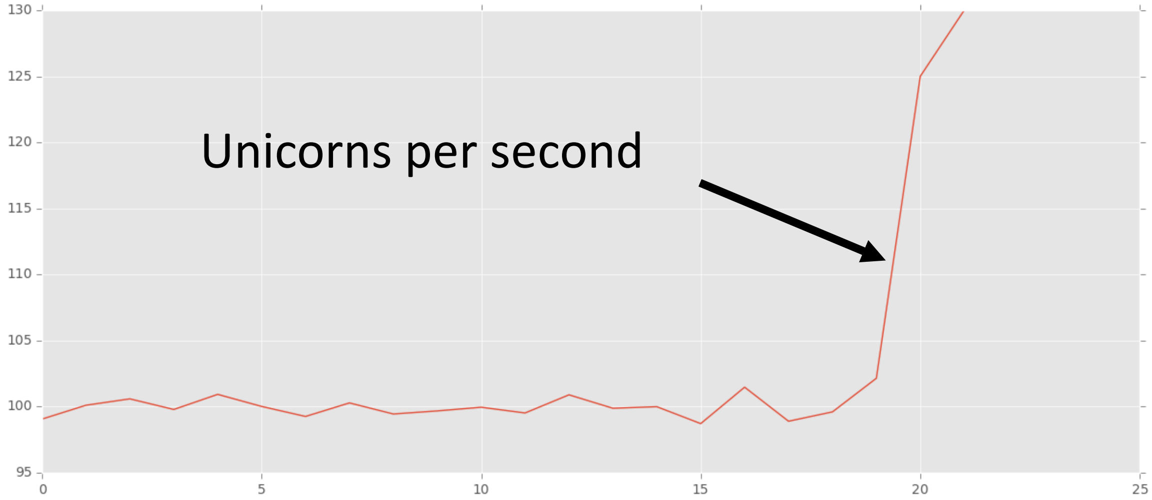
Axioms – Solution Domain

- Unsupervised
- Non-Parametric
- Robust
- Streaming
- Adaptive
- Domain-agnostic
  - Purely numeric
  - No information about underlying subjects or causes of the behaviour stream



Axioms – Solution Domain

- Unsupervised
- Non-Parametric
- Robust
- Streaming
- Adaptive
- Domain-agnostic
  – Purely numeric
  – No information about underlying subjects or causes of the behaviour stream



Getting Data In

Time Series Feature Engineering

- If you already have ***dense, regular, numeric*** time series (aka "metrics" or "KPIs") you're good to go

Getting Data In

Time Series Feature Engineering

- If you already have ***dense, regular, numeric*** time series (aka “metrics” or “KPIs”) you’re good to go
- If you have something else, now you have a ***time series feature engineering*** problem

Getting Data In

Time Series Feature Engineering

- If you already have ***dense, regular, numeric*** time series (aka "metrics" or "KPIs") you're good to go
- If you have something else, now you have a ***time series feature engineering*** problem
- There are *inescapable tradeoffs* between **density** and **precision**

Getting Data In

Time Series Feature Engineering

- If you already have ***dense, regular, numeric*** time series (aka "metrics" or "KPIs") you're good to go
- If you have something else, now you have a ***time series feature engineering*** problem
- There are *inescapable* tradeoffs between **density** and **precision**
- Increased precision implies sparser time series
  - Also increased memory and bandwidth usage!

Getting Data In

Time Series Feature Engineering

- If you already have ***dense, regular, numeric*** time series (aka "metrics" or "KPIs") you're good to go
- If you have something else, now you have a ***time series feature engineering*** problem
- There are *inescapable* tradeoffs between **density** and **precision**
- Increased precision implies sparser time series
  - Also increased memory and bandwidth usage!
- TSFE requires dealing with **Time**, **Space** and **Values**

Getting Data In

Time Series Feature Engineering

- **Time**
  - How *frequently* do new values arrive?
  - How *regularly* do new values arrive?
  - How *precisely* do we want to be able to record the time when the measurement was taken?
    - Finer time resolution **increases sparsity**: the probability that any event occurred during a particular time window is decreased
- Space
- Values

Getting Data In

Time Series Feature Engineering

- Time
- **Space** - how *precisely* do we want to be able to relate time series back to the underlying event stream?
  - *How many* dimensions? e.g. IP address, geo. coordinates, MIME type, HTTP response code
    - Adding dimensions increases precision, but also **magnifies the likelihood of sparsity**
  - Within a dimension, *how precise* do we need to be?
    - Full IP address or /24? Distinguish 400, 401, 403, 404 or just 4xx?
    - Country, state/province, city, neighbourhood, building, …?
    - Extra precision **increases the likelihood of sparsity**
- Values

Getting Data In

Time Series Feature Engineering

- Time
- Space
- **Values**
  - How do we generate a number?
    - Get a numeric field as-is (i.e. a “gauge”)
    - Increment a counter
  - How do we aggregate multiple values?
    - Min, max, mean, etc.
  - How should we handle missing values?
    - ”Replace null with zero” only makes sense for something we know is a counter
    - “Take the previous value” might make sense

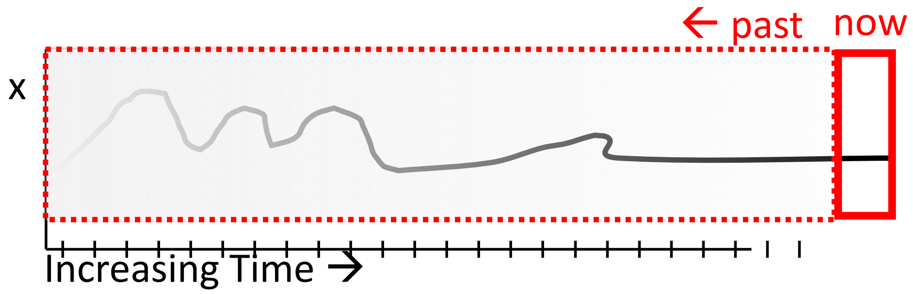
Metric Anomaly Detection Algorithms

- Proprietary! Not open source or off-the-shelf.
- Spatial and temporal algorithms
  - What do we mean by “spatial” and “temporal”?
  - Completely orthogonal, irreducible distinction
    - One cannot substitute for the other
    - Neither is always applicable to every time series

Metric Anomaly Detection Algorithms

Temporal Analysis (aka "Trending" algorithm)

- Analyze one time series at a time (embarrassingly parallel)
- Alerting when *present* behaviour is surprising compared to *past* behaviour



Metric Anomaly Detection Algorithms

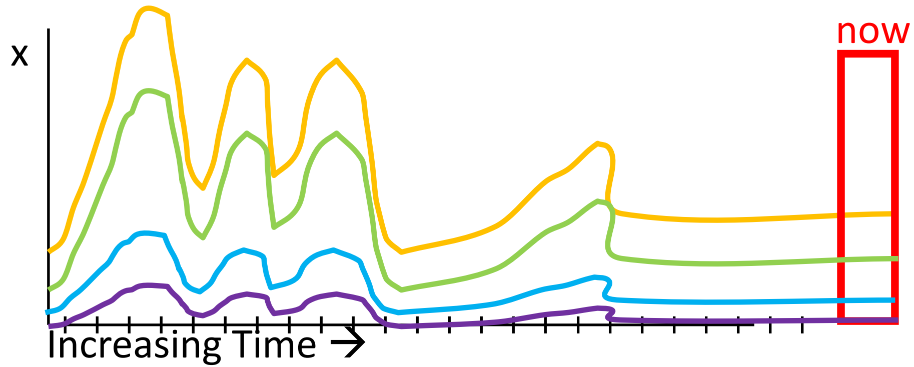
Trending Algorithm Constraints

- Good results only when there is a history of recurring patterns in the underlying event stream
  - Not necessarily periodic, just recurring
- How much history?
  - Preliminary (usually bad) results after ~2000 points
    - e.g. 1.5 days at 1-minute resolution
  - Great results after a “full period” has been observed (e.g. 7 days)
  - More is better! (modulo memory, storage…)

Metric Anomaly Detection Algorithms

Spatial (“Cohesive”) Algorithm

- Compare *present* behaviour of *multiple* metrics



Metric Anomaly Detection Algorithms

Cohesive Algorithm Constraints

- Given a *set** of time series that are *expected*† to behave *similarly*‡, detect when one or more of them departs from their peers

** set*

>= 3 members

† expected

by a human analyst or interesting ML process

‡ similarly

Roughly the same shape

Scale and magnitude invariant

Metric Anomaly Detection Algorithms

Cohesive Algorithm Characteristics

- No periodicity required
- History improves scale/magnitude invariance
- Performance relies on similarity within group
  - What if the group isn't inherently cohesive?
    - Lots of alerts early on
    - Then, the algorithm adapts to the chaos
    - If the group returns to cohesion, the algorithm will automatically adapt to the "new normal".

Metric Anomaly Detection Algorithms

Cohesive Algorithm: Example Use Case #1

- A cluster of servers performing *a similar role* for the *same application*, *behind the same load balancer*
- Assuming the load balancer is operating nominally, many server metrics should be roughly correlated, e.g.:
  - CPU usage (user, system, idle)
  - Disk usage (reads, writes, IOPS)
  - Network usage (bandwidth, # active sockets)
  - Application-specific metrics (requests handled per second, 500 errors, authentication failures, active sessions)

Metric Anomaly Detection Algorithms

Cohesive Algorithm: Example Use Case #2

- Imagine some wind turbines on the same hill
- We can’t predict wind direction and speed very well (yet?)
- But we expect every turbine should be roughly cohesive in several metrics:
  - rotation speed
  - power generation rate
  - vibration
  - direction
    - * actually, because this is a periodic metric (359° ≈ 1°), we don’t support it well right now
- If any metric for any turbine differs significantly from its peers, we should be notified, and maybe send a team to investigate

Other approaches we have tried

- ~~3-sigma~~
- Kolmogorov-Smirnov test over sliding windows
- Time-series forecasting methods
  - Holt-Winters (previous version of ITSI AD is based on its non-parametric version)
  - ARIMA, etc
- One-class SVM
- Clustering methods – DBSCAN, K-means, etc
- Various R, Python packages

MAD Service Engineering

- MAD = “Metafor Anomaly Detection”

MAD Service Engineering

- MAD = “~~Metafor~~ Anomaly Detection”

MAD Service Engineering

- MAD = “**Metric** Anomaly Detection”

MAD Service Engineering

- MAD = "**Metric** Anomaly Detection"
- Written in Scala
  - using Akka for concurrency

MAD Service Engineering

- MAD = “**Metric** Anomaly Detection”
- Written in Scala
  - using Akka for concurrency
- Uses Search Command Protocol v2 (available since Splunk 6.3)
  - Runs forever, doesn’t get restarted every 50k events
  - Receives data soon after it arrives at an indexer, no polling

MAD Service Engineering

- MAD = “**Metric** Anomaly Detection”
- Written in Scala
  - using Akka for concurrency
- Uses new Chunked External Command feature of Splunk 6.3
  - Runs forever, doesn’t get restarted every 50k events
  - Receives data soon after it arrives at an indexer, no polling
- Fast!

MAD Service Engineering

- MAD = “**Metric** Anomaly Detection”
- Written in Scala
  - using Akka for concurrency
- Uses new Chunked External Command feature of Splunk 6.3
  - Runs forever, doesn’t get restarted every 50k events
  - Receives data soon after it arrives at an indexer, no polling
- Fast!
- Designed for general-purpose use, no coupling to ITSI runtime

How to get it

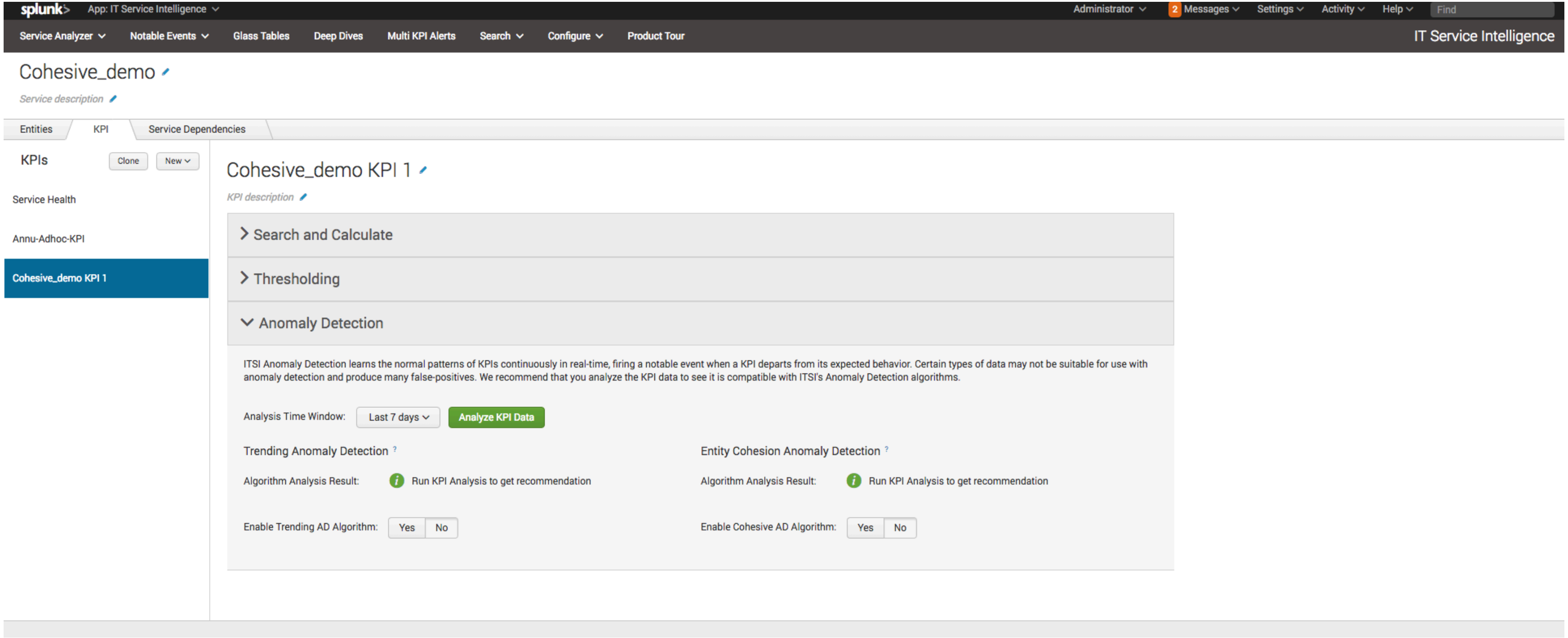
ITSI-AD

- ITSI 2.3 “Batman” (July 2016)
  – ITSI Anomaly Detection replaced with Trending algorithm

- ITSI 2.4 “Catwoman” (.conf 2016)
  – Cohesive algorithm added
  – Compares entities within a KPI

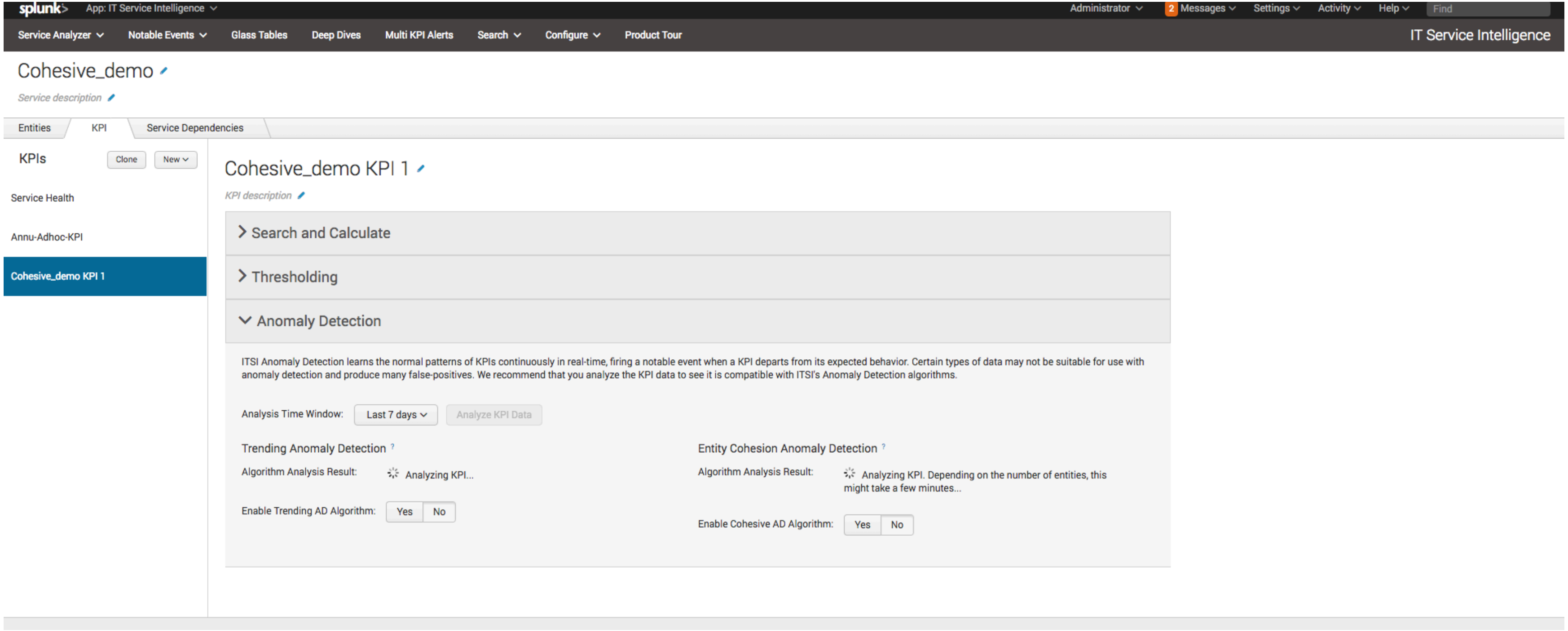
How to get it

ITSI-AD



How to get it

ITSI-AD



How to get it

Annu-Adhoc-KPI

Cohesive_demo KPI 1 ×

> Search and Calculate

> Thresholding

> Anomaly Detection

ITSI Anomaly Detection learns the normal patterns of KPIs continuously in real-time, firing a notable event when a KPI departs from its expected behavior. Certain types of data may not be suitable for use with anomaly detection and produce many false-positives. We recommend that you analyze the KPI data to see it is compatible with ITSI's Anomaly Detection algorithms.

Analysis Time Window: Last 7 days ⌄ Analyze KPI Data

Trending Anomaly Detection ?

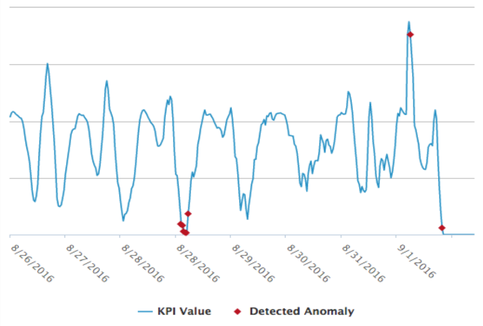
Algorithm Analysis Result: ✓ Recommended ?

Enable Trending AD Algorithm: Yes No

⌄ Analysis Breakdown

Percentage of Data Points with Anomalies: 2% (Expected < 10%)

KPI Value for Last 7 days



Entity Cohesion Anomaly Detection ?

Algorithm Analysis Result: ✓ Recommended ?

Enable Cohesive AD Algorithm: Yes No

⌄ Analysis Breakdown

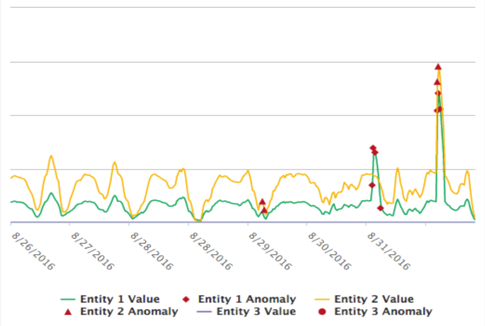
Entities Analyzed: 6

Entities with Detected Anomalies: 2

Average Anomalies Per Entity: 89.5

Percentage of Data Points with Anomalies: <1% (Expected < 10%)

3 Anomalous Entity KPI Values for Last 7 days



THANK YOU

.conf2016

splunk>