Blueprints for Actionable Alerts
...while you get settled...

- Latest Slides:
  - [https://splunk.box.com/v/blueprints-alerts](https://splunk.box.com/v/blueprints-alerts)

- Collaborate: #alerting
  - Sign Up @ [http://splk.it/slack](http://splk.it/slack)

- Load Feedback
Blueprints for Actionable Alerts

Presented by Splunk Blueprints

Burch | Senior Best Practices Engineer

.conf2017 | Version 0.0
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.

Splunk, Splunk>, Listen to Your Data, 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. © 2017 Splunk Inc. All rights reserved.
“Scale customer success through the automation of adoption services and best practices”

Blueprint’s Mission
What’s a “Burch”? 

Senior Best Practices Engineer

- Was a Senior Sales Engineer
- Before that, Splunk Customer
- Before that, Middleware Eng
- Before that, Computer Science
- Before that, an idea of my parents
“From spam to glam with Splunk Alerts”

Should you be here?
1. Stage 1: Message of Concern
2. Stage 2: Thresholds
3. Stage 3: Relative Percentages
4. Stage 4: Average Errors
5. Stage 5: Percentiles
6. Bonus Stage 6: IT Service Intelligence
7. Stage 7: Actionable Alerts
Phase 1: Message of Concern
Attempted Solution

Basic Search => Spammy Alert

[Spam]
action.email = true
action.email.to = welovespam@spam.com
counttype = number of events
cron_schedule = */15 * * * *
dispatch.earliest_time = -15min
dispatch.latest_time = now
enableSched = true
quantity = 0
relation = greater than
search = index=_internal error
Attempted Solution

Basic Search => Spammy Alert

[Spam]
action.email = true
action.email.to = welovespam@spam.com
counttype = number of events
cron_schedule = */15 * * * *
dispatch.earliest_time = -15min
dispatch.latest_time = now
enableSched = true
quantity = 0
relation = greater than
search = index=_internal error
Result

6,500 errors over last 15min

#unrealistic
Obvious Improvements

- Scope of problem is large
  - Solution: indexed fields (index, source, sourcetype, and/or pattern)

- Problem: “error” matches more than desired
  - Solution: bind with fields like log_level=“error”

- Result: Stronger search ignores benign results

```bash
index=_internal sourcetype=splunkd source!="*splunkforwarder*" log_level=ERROR
```
Phase 2: Thresholds
Attempted Solution

- Only alert if more than “arbitrary” # occurrences / time
  - Arbitrary = perception of healthy

```
index=_internal sourcetype=splunkd source=!="*splunkforwarder*" log_level=ERROR
| stats count
| where count>20
```

- or...

```
if number of events
```

20
Result & Obvious Improvements

- Ignores variances of different types of errors
  - Web errors rarely happen but server errors happen often

- Fluctuations relative to usage
  - Threshold too small or large during peak or minimal usage, respectively
  - Static thresholds not adjusting with business growth or decline
Phase 3:
Relative Percentages
What 2 Clean?
New Concept

eval goal_attacking = coalesce( spam, system )

Spam
- Normalize against # of errors
- Ignore non error events
- log_level=ERROR
- Good for clean up
- Bad for permanent

System
- Normalize to all events
- Include all error + non error events
- log_level=* 
- Good for permanent
- Bad for clean up
Attempted Solution
Large % Items

```
index=_internal sourcetype=splunkd source!="*/splunkforwarder/*" log_level=*  
| stats count, count(eval(log_level=="ERROR")) AS error_count by component  
| where ( error_count / count ) > .50
```
Result & Obvious Improvements

- Huge improvement
  - Less spam
  - Adjusts because normalized to volume

- What if that’s normal?
  - Then persistent alerts that should be ignored = spam + noise!

- Percentage => Static => Arbitrary?!
Phase 4: Average Errors
### Attempted Solution

**Current period vs historical average**

```plaintext
index=_internal sourcetype=splunkd source=!"*/splunkforwarder/" log_level=ERROR
| bin span=5min _time
| stats count by _time, component
| stats latest(count) as current_count, avg(count) as historical_count by component
| where current_count > historical_count
```

<table>
<thead>
<tr>
<th>component</th>
<th>current_count</th>
<th>historical_count</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdminHandler:PersistMessages</td>
<td>16</td>
<td>8.500000</td>
</tr>
<tr>
<td>Application</td>
<td>30</td>
<td>10.000000</td>
</tr>
<tr>
<td>ApplicationUpdater</td>
<td>77</td>
<td>27.333333</td>
</tr>
<tr>
<td>ArchiveContext</td>
<td>6</td>
<td>4.666667</td>
</tr>
<tr>
<td>CMSlave</td>
<td>3</td>
<td>2.333333</td>
</tr>
<tr>
<td>DistributedBundleReplicationManager</td>
<td>60</td>
<td>12.000000</td>
</tr>
<tr>
<td>ExecProcessor</td>
<td>63</td>
<td>30.735828</td>
</tr>
<tr>
<td>FilesystemChangeWatcher</td>
<td>2</td>
<td>1.772727</td>
</tr>
<tr>
<td>HttpClientRequest</td>
<td>204</td>
<td>82.500000</td>
</tr>
<tr>
<td>IndexAdminHandler</td>
<td>64</td>
<td>33.000000</td>
</tr>
</tbody>
</table>

- **Events:** 619,072 events (7/19/16 4:00:00.000 PM to 7/26/16 4:58:23.000 PM)
- **No Event Sampling**
- **Job**
- **Visualization**
Result

▶ Adjusts with changes in environment!

▶ Slow
  • Summary Indexing?
  • Acceleration?

▶ How often alert?
  • Definition of average!
Statistics Detour

- People who understand you can make statistics say whatever you want
- People who don't think twice about this chart

survivingtheworld.net

File under: Numbers Are Your Master Now!
Statistics Detour

Historical # of errors / 5 min period

11 87 5 56
19 5 67
21 18 77
31
54000
Statistics Detour

At what value does this become actionable?

Min
Average
Max

18
19
21
31
56
67
77
87

0 to 10 | 11 to 20 | 21 to 29 | 30 to 39 | 40 to 49 | 50 to 59 | 60 to 69 | 70 to 79 | 80 to 89 | 90 to 99
## Statistics Detour

What if we could skim off outliers?

Alert at *near* max?

<table>
<thead>
<tr>
<th>0 to 10</th>
<th>11 to 20</th>
<th>21 to 29</th>
<th>30 to 39</th>
<th>40 to 49</th>
<th>50 to 59</th>
<th>60 to 69</th>
<th>70 to 79</th>
<th>80 to 89</th>
<th>90 to 99</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>19</td>
<td>21</td>
<td>31</td>
<td>56</td>
<td>67</td>
<td>77</td>
<td>87</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

0 to 10 | 11 to 20 | 21 to 29 | 30 to 39 | 40 to 49 | 50 to 59 | 60 to 69 | 70 to 79 | 80 to 89 | 90 to 99 |
perc\(X\)(Y) = Returns the \(X\)-th percentile value of the numeric field Y, where \(X\) is an integer between 1 and 99. The percentile \(X\)-th function sorts the values of Y in an increasing order. Then, if you consider that 0% is the lowest and 100% the highest, the functions picks the value that corresponds to the position of the \(X\)% value.
Statistics Detour

<table>
<thead>
<tr>
<th>0 to 10</th>
<th>11 to 20</th>
<th>21 to 29</th>
<th>30 to 39</th>
<th>40 to 49</th>
<th>50 to 59</th>
<th>60 to 69</th>
<th>70 to 79</th>
<th>80 to 89</th>
<th>90 to 99</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5</td>
<td>11</td>
<td>21</td>
<td>31</td>
<td>56</td>
<td>67</td>
<td>77</td>
<td>87</td>
<td></td>
</tr>
</tbody>
</table>

perc90(this_result_set) = ?
Statistics Detour


1 result (7/31/16 2:46:51.000 PM to 7/31/16 3:01:51.000 PM) No Event Sampling Job

Events Patterns Statistics (1) Visualization

10 Per Page Format Preview

perc90(count)

87
Warning: Assumption

Shout out to Xander!
Warning: Heavy Tails

The Shape of a Fat-Tailed Distribution
Warning: Reality

What percentile is appropriate given this distribution?
index=_internal sourcetype=splunkd
source!="*/splunkforwarder/*"
| bin span=5min _time
| stats count AS group by _time
| bin span=1000 group
| stats count by group
| sort group
Phase 5: Percentiles
Attested Solution

▶ Current period’s error rate vs. historical error rate
  • by error category (component)

```snip
index=_internal sourcetype=splunkd source!="*/splunkforwarder/*"
log_level=ERROR
  | bin span=5min _time
  | stats count by _time, component
  | stats perc95(count) AS perc95_count, latest(count) AS current_count by component
  | where current_count > perc95_count
```

▶ Performance?
Generate malleable historical data

```sh
index=_internal sourcetype=splunkd source!="*/splunkforwarder/**" log_level=ERROR

    | bin span=5min _time
    | stats count by _time, component
```

Alert upon historical data

```sh
index=summary_internal sourcetype=stash source="my search name"

    | stats count by _time, component
    | stats perc95(count) AS perc95_count, latest(count) AS current_count by component
```
The Lasso Approach

- Triage Strategy
- Perimeter around errors
- Tighten lasso by reducing percentile
- Rinse & repeat
Alternatives

- Address most common errors first
  - Start at 5\textsuperscript{th} percentile and work up

- Normalization Frames:
  - Same errors
  - All errors
  - All events
  - Time windows (e.g. work hours)
Result

▶ Adjusts with changes in environment!

▶ Requires Maintenance
  • Power User skillz
  • Summary Indexing

▶ Not period time adjusted
  • Fluctuations in business day or period
Bonus Phase 6: IT Service Intelligence
“Make alerting accessible, usable and valuable to everyone!”

Why ITSI?
Quantile, Range, and STDDEV. Oh my!
Adaptive Thresholds
Anomaly Detection
Actionable Alerts Made Easy

I DID ABSOLUTELY NOTHING TODAY

AND IT WAS EVERYTHING I THOUGHT IT COULD BE

easymemes.com
1. Stage 1: Message of Concern
2. Stage 2: Thresholds
3. Stage 3: Relative Percentages
4. Stage 4: Average Errors
5. Stage 5: Percentiles
6. Bonus Stage 6: IT Service Intelligence
7. Stage 7: Actionable Alerts

Wrap Up
What Now?
Related breakout sessions and activities...

1. Rate this! (be honest)
2. Collaborate: #alerting
   • Sign Up @ http://splk.it/slack
3. Customer Success Studio
4. More talks, search for
   • Blueprints
   • Burch
   • Champagne
   • Delaney
   • Optimization
   • Best Practices
   • Veuve
Questions & Discussion?

Don't forget to rate this session in the .conf2017 mobile app