

The Culture and Reality of Monitoring at Stack Overflow



Stack Overflow, Bosun, and Me

- Been with the company over 6 years
 - Although many core values have stayed the same, the company was ~30 people and is now ~300
- We started coding Bosun 3 years ago (Open Sourced 2 years ago)
 - Time series alerting was a somewhat novel idea (at least in the open source world)
 - Goal was build an alerting system that could create more informative alerts with a much better S/N ratio
 - With a bigger company and more complexity in the company - thinking about different problems today: More about how users interact with monitoring and communicate with it



Some of Stack's Monitoring Stack

- **Bosun**: Alerting IDE
- **scollector**: Cross platform collection agent
- **OpenTSDB**: Current time series database
- **Grafana**: User crafted dashboard
- **Elastic, Kibana**: Can also be queried via Grafana and Bosun
- **BosunRepoter.NET**: .NET Library for Bosun / OpenTSDB
- **Opserver**: Monitoring dashboard, control of Redis, Elastic, in depth SQL monitoring
- **Client Timings**: Sampled web browser Performance
- **PRIZM**: In house A/B testing and Events
- **Looker**: Business Analytics
- **Solarwinds**: Old system, retired after we switch Observer

More accurately...

The **Cultures** and **Realities** of Monitoring at Stack Overflow

because reality is far more
complex than our imagination
of it and it is a mix of successes
and failures

I think of the difference between the reality of a system, and how we imagine and reason about it as our **mental model** of a system

Mental models are the **abstractions** we create to cope with the complexity of reality

Monitoring a medium of communication that we use to share these mental models or aspects of them

Effectiveness of monitoring can analyzed as the effectiveness of end to end communication

Monitoring is Hard Medium to Communicate with

End to end communication with monitoring involves
Humans communicating via machines to Humans

The model in a reduced form is:

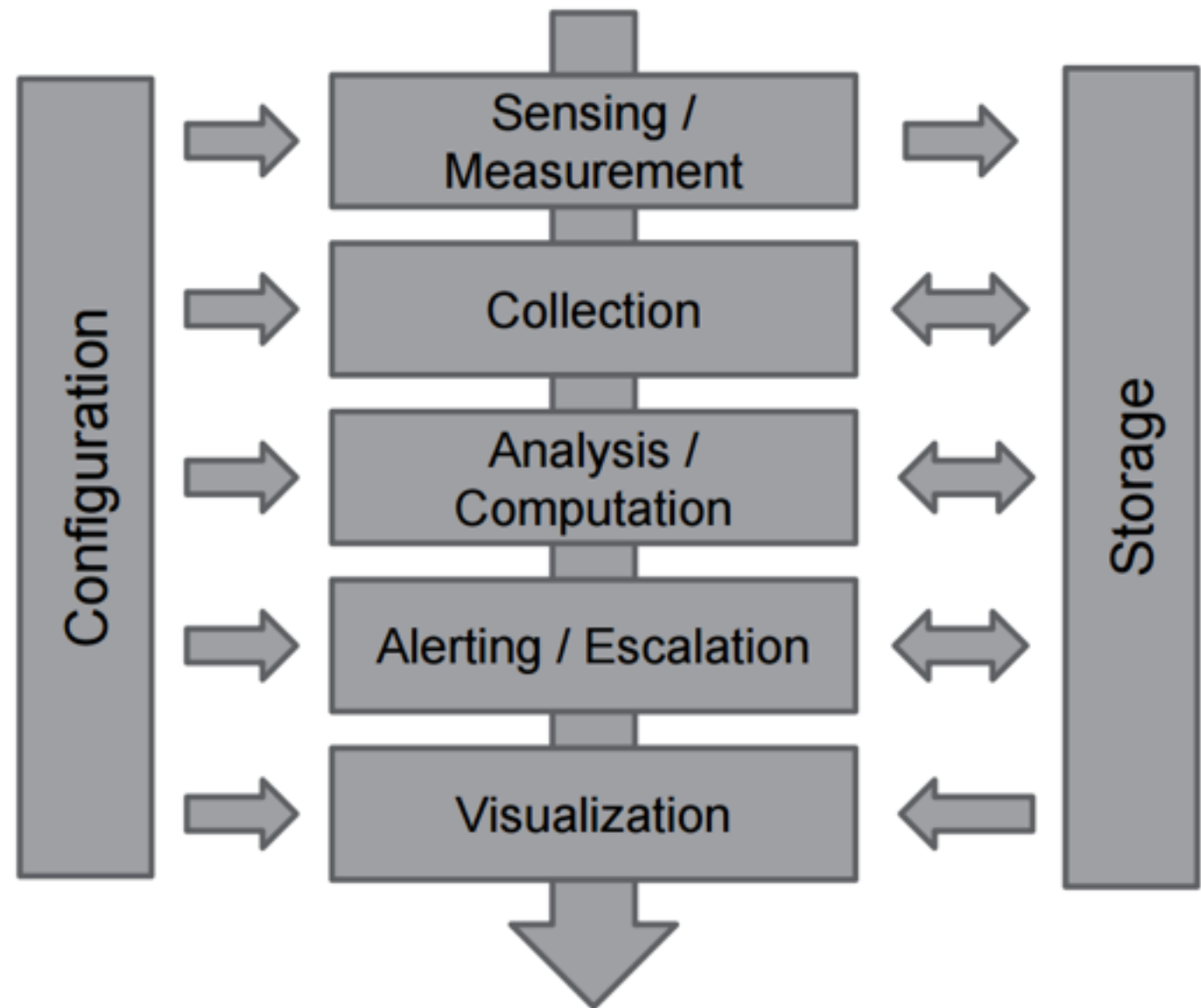
Human => Machine (Software) => Human

Tom creates metric in linux kernel

Jane writes collection agent to gather the data and send it a TSDB

Alfred writes alerts in Bosun to query the TSDB.

Alfred and Sally also create dashboards in Grafana against the TSDB



Jill and Co Consume Alerts and Dashboards

Human (Author) =>

Machine (Software) =>

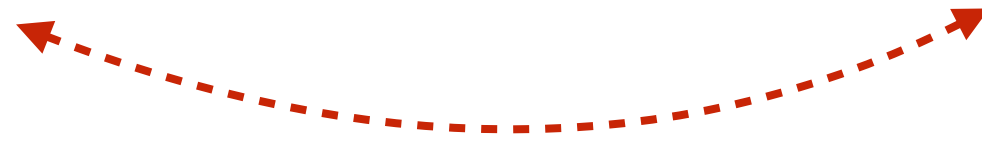
(Consumer) Human

Some of these people may be the member of the same organization or none of them may be.

So often we are communicating *serially* through software

So when one light goes out, the rest follow

Human => Machine => Human



Human <=> Machine

If there is human => human communication in a medium other than documentation, it does not scale and is error prone

Looking at Bosun and Grafana at Stack Overflow in Terms of Human Interaction

Three Personas

1. Authors
2. Consumers
3. Some Degree of Author and Consumer

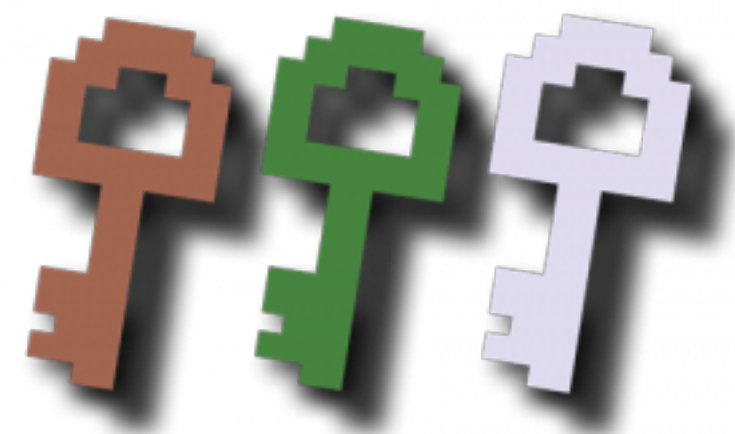
Bosun Alerting Author Workflow

Workflow is all in same UI

- Level 1: Graph, which becomes an expression
- Level 2: Expression: Manipulate the query you graphed
- Level 3: Rule: Refine expression, create template, preview with historical testing. Save Config (New!)

But you pass the trials to level up to Level 3

READY PLAYER 1

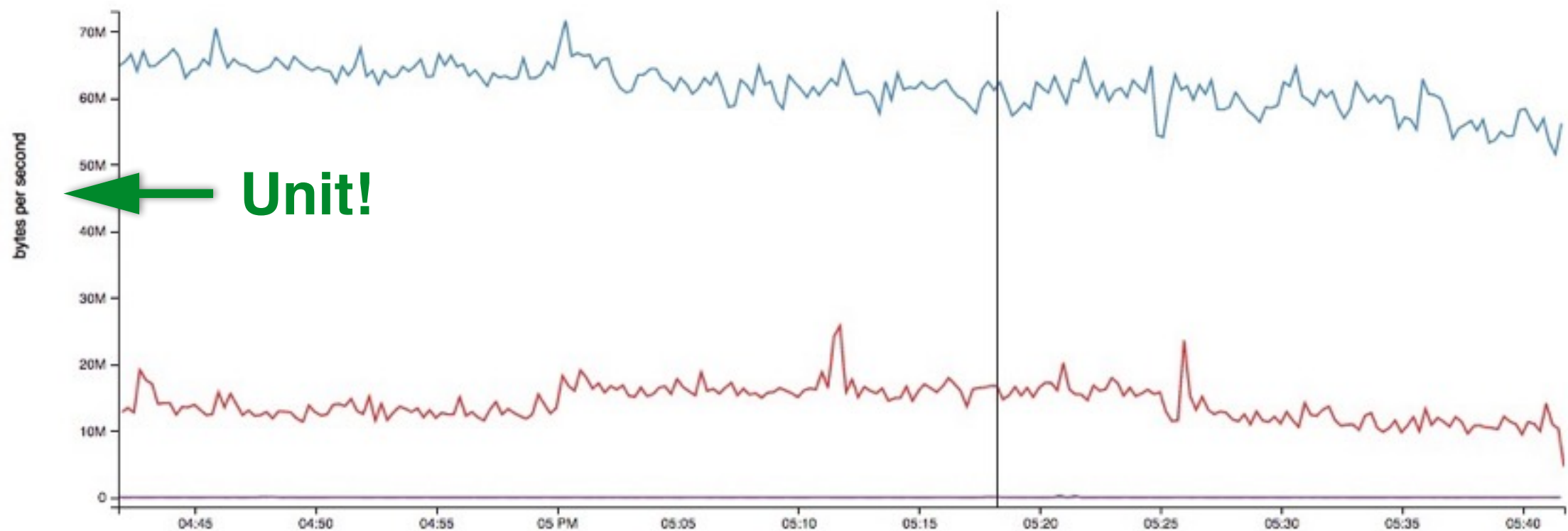


1h-ago End Query Switch Time ☒ Auto Downsample ☐ Refresh ☐ Normalize ☐ Annotations embed image

Query 1 +

Metric	ps.net.bytes	host (566)	auto	ny-lb*	auto	y min		y max	
Aggregator	sum	direction (2)	auto		auto				
Type	auto	inname (2606)	auto		auto				

Auto Gauge / Counter!



Unit!

Time: 2016/11/25-17:18:14 (23m34s ago)

os.net.bytes(host=ny-lb05): 14.84245M

os.net.bytes(host=ny-lb03): 62.49483M

os.net.bytes(host=ny-lb04): 152.1172k

os.net.bytes(host=ny-lb06): 168.4934k

Builds Query Expression!

Queries

q("sum:rate{counter,,1}:os.net.bytes{host=iwildcard(ny-lb*)}", "1h", "")

Expression

Rule

Metric Description

os.net.bytes

The rate at which bytes are sent or received over each network adapter.

Description!

GREAT!

Except for Level 1 is deceptively simple...

1h-ago

End

Query

Switch Time

☒ Auto Downsample☐ Refresh☐ Normalize☐ Annotations

embed

image

Query 1

+

Metric

ps.net.bytes

Aggregator

sum

Type

auto

host (566)

auto

ny-lb*

auto

direction (2)

auto

iface (1742)

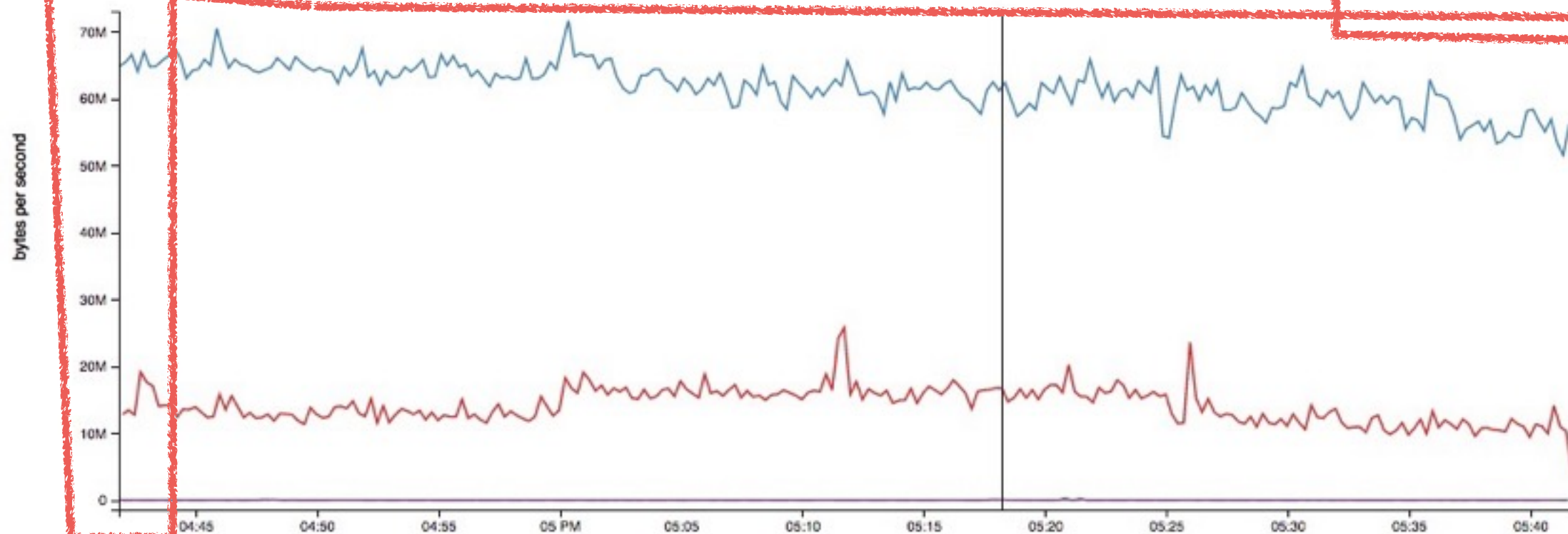
auto

iname (2606)

auto

y min

y max



Time: 2016/11/25-17:18:14 (23m34s ago)

os.net.bytes(host=ny-lb05): 14.84245M

os.net.bytes(host=ny-lb03): 62.49483M

os.net.bytes(host=ny-lb04): 152.1172k

os.net.bytes(host=ny-lb06): 168.4934k

Queries

q("sum:rate(counter,,1):os.net.bytes{host=iwildcard(ny-lb*)}", "1h", "")

Expression

Rule

Metric Description

os.net.bytes

The rate at which bytes are sent or received over each network adapter

Some Issues for the Graph Page

- If you don't understand the TSDB the graphs may not be what you expect. In our case, OpenTSDB it means you need to understand:
 - How tags work: Group By / Filter
 - Aggregation
 - Counter / Gauge (and that counters are not in correct place in OpenTSDB's order of operations)
 - Downsampling
 - Linear Interpolation
- The description could be better since it is specific to *physical* interfaces. Also does not describe how the tags work
- The Y-Axis is Si units, which means base 10. Sometimes people in base 2 with network, and even though it says bytes people think bits out of habit

How it fails in terms of communication

- Needing to understand the TSDB, the underlying metrics, and how they combine leads to data not meaning what authors think it means early in the pipeline

and/or

- Only accessible to authors that are willing to accept a steep learning curve

Congrats, you made it to:

Bosun Level 1



Expression Page

q("sum:rate{counter,,1}:os.net.bytes{host=iwildcard(ny-lb*)}", "1h", "")

Date

yyyy-mm-dd

Time

HH:MM:SS

Test

Rule

Image

Results

Graph

Queries

[end=2016/11/26-15:53:25&m=sum:rate{counter,,1}:os.net.bytes{host=iwildcard\(ny-lb*\)}&start=2016/11/26-14:53:25](#)

group	result	computations
{ host=ny-lb05 }	show	
{ host=ny-lb03 }	show	
{ host=ny-lb04 }	show	
{ host=ny-lb06 }	show	

Good

- A REPL for Bosun's expression language. Lets you incrementally craft expressions and see the results of intermediate stages
- Power ...

Raw Power



but no more buttons

Expression Language is Quick to Learn

If:

- This makes immediate sense to you: A functional set based and typed domain specific language (DSL) for querying time series databases and reducing time series into other sets
- You read all the documentation, look at examples on the internet, and watch some training youtube videos with careful attention

Example Incantation

```
$time = 5m
$warn_threshold = 10
$crit_threshold = 200
$metric = max:rate{counter,,1}:exceptional.exceptions.count
$error_count = sum(t(change("$metric{application=AdServer,machine=*,source=*}", "$time", ""), "application"))
$error_count > $warn_threshold
```




Bosun Level 2

Rule Page

- Create rich templates:
 - Tables, Graphs, Links, Elastic Logs, Results from HTTP calls etc
 - After learning the currently poorly documented template language
- Test the alert against history and view instances of the template
- Save the config



Jump to:

alert

template

lookup

notification

macro

template generic.graph

Download

Syntax Highlighting

Validate

Save Dialog

```
3937 # warnNotification = default
3938 #}
3939
3940 alert bosun.failed.emails {
3941     template = generic
3942     $generic_no_details = true
3943     ignoreUnknown = true
3944     $notes = This alert will fire if bosun has failed to send an email in the last 5 minutes. ignoreUnknown is set since
normally only one bosun instance will be trying to send emails.
3945     critNotification = sre-chat
3946     $q = min(q("sum:rate{counter,,1}:bosun.email.sent_failed{host=wildcard(*)}", "5m", ""))
3947     crit = $q > 0
3948 }
3949
3950 alert bosun.check.duration {
3951     template = generic.graph
3952     ignoreUnknown = true
3953     $notes = This alert will trigger if Bosun takes longer than expected to run all the rule checks. This usually means there
is an issue with reading data from opentsdb. The duration is usually less than 5 minutes for all checks to run, and the alert
will trigger if in the last 30 minutes the rule check has taken over 15 minutes to finish.
3954     warnNotification = default
3955     $q = avg(q("sum:bosun.check.duration{host=ny-bosun01|ny-bosun02|co-tsdb01}", "30m", "")) / 60
3956     $q_format = %.0f
3957     $unit_string = ` minutes is higher than expected`
3958     $generic_graph = q("sum:bosun.check.duration{host=ny-bosun01|ny-bosun02|co-tsdb01}", "8h", "") / 60
3959     $graph_unit = Time in minutes for all rule checks to finish
3960     warn = $q > 15
3961     runEvery = 30
3962 }
3963
```

From

yyyy-mm-dd

HH:MM

To

yyyy-mm-dd

HH:MM

Intervals

5

Step Duration (m)

Email

Template Group

Test bosun.check.duration

Results

Template

Timeline

Subject

normal: bosun check duration: 1 minutes is higher than expected on co-tsdb01

Body

[Acknowledge](#) [View Alert in Bosun's Rule Editor](#) [View co-tsdb01 in Opserver](#) [View co-tsdb01 in Kibana](#)

Key: bosun.check.duration{host=co-tsdb01}

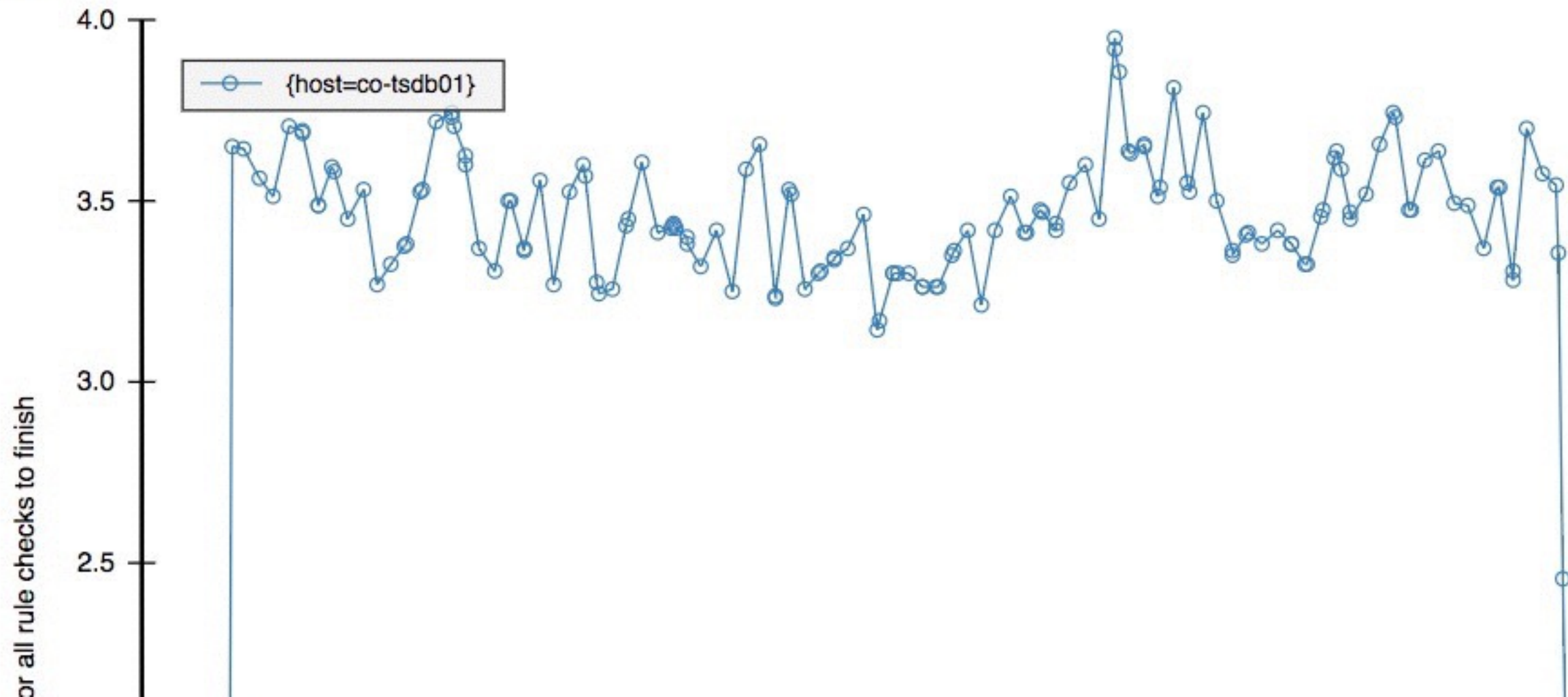
Incident: #0

Notes: This alert will trigger if Bosun takes longer than expected to run all the rule checks. This usually means there is an issue with reading data from opentsdb. The duration is usually less than 5 minutes for all checks to run, and the alert will trigger if in the last 30 minutes the rule check has taken over 15 minutes to finish.

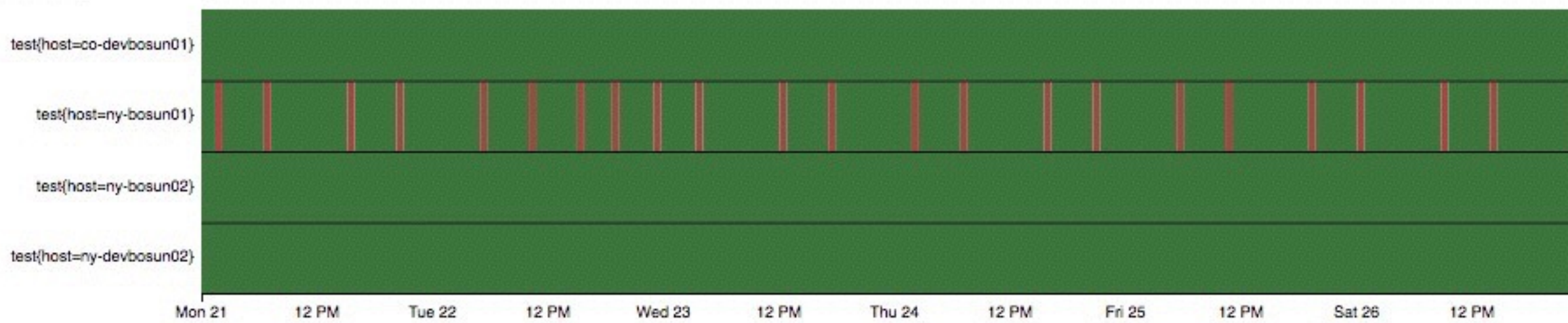
Tags

host co-tsdb01

Graph



Results Template Timeline



2016/11/21-08:09:49

test{host=co-devbosun01}

test{host=co-devbosun01}

1 eve

test{host=ny-bosun01}

47 eve

test{host=ny-bosun01}: normal

2016/11/21-00:00:00 (6d15h44m55s ago) for 1h26m50s

test{host=ny-bosun01}: critical

2016/11/21-01:26:50 (6d14h18m05s ago) for 43m25s

test{host=ny-bosun01}: normal

2016/11/21-02:10:15 (6d13h34m40s ago) for 4h20m30s

test{host=ny-bosun01}: critical

2016/11/21-06:30:45 (6d09h14m10s ago) for 43m25s

test{host=ny-bosun01}: normal

2016/11/21-07:14:10 (6d08h30m45s ago) for 7h57m35s

test{host=ny-bosun01}: critical

2016/11/21-15:11:45 (6d00h33m09s ago) for 43m25s

Subject

Reached Bosun Level 3

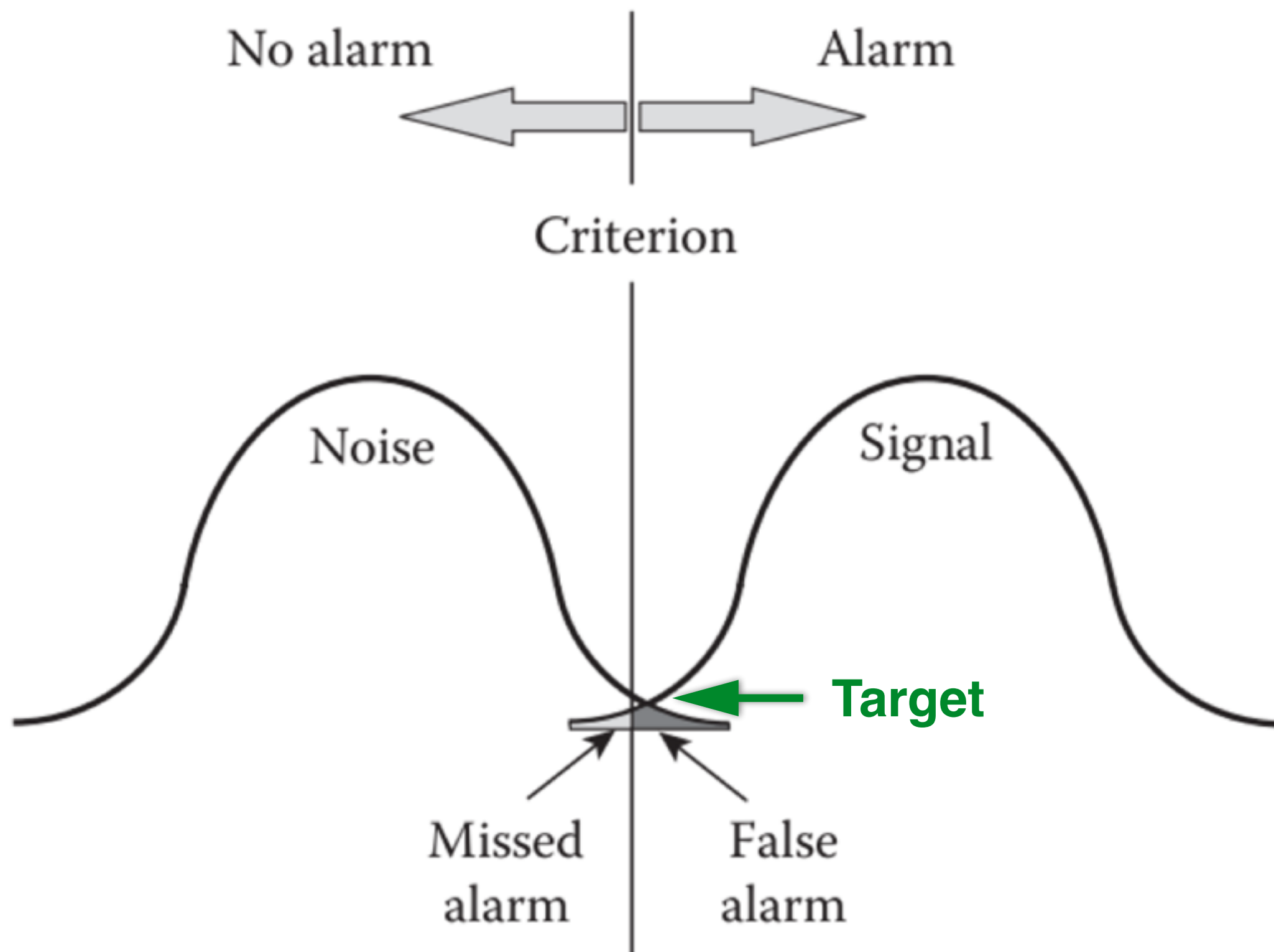


Why the complexity?

Firstly

Give authors the power to achieve good communication with alerting:

1. Good signal to noise ratio
2. Informative alerts
3. Power user workflow (IDE)



Secondly

Bosun puts the burden
of communication on
the author

- Moves communication earlier in the pipeline
- Reality of alerts often has exceptions - so the power of the expression language is used a lot in practice
- Tune alerts up front and workflow for fast follow up tuning
- As a team scales, the author to consumer ratio for alerts and dashboards increases - so it scales better

however, there are consequences

Now authors are scarce

- Authorship requires training / time investment so there are less authors
- As the number of teams increases, we have a mix of
 1. Self-Service DevOps: Team uses Bosun independently
 2. Concierge: Bosun experts create things for people (or help)
 3. Missing: Team has observability and alerting gaps

Thirdly

Reasons.

- Time
- Uncharted territory
- Seemed like a good idea at the time
- and other excuses...

Complexity / Power

- Rule of thumb is that as something becomes more flexible and powerful it also becomes more complicated
- However, simplicity is a feature since it allows humans to reason about it
- Simplifying complexity for authors while maintaining power to handle reality is my new goal

Save Feature

- Mistakingly sacrificing usability for power
- Did not have saving in UI because we wanted version control - and integration with VCS was too much work
- Realized that if we could just have a save hook with some locking, and the program that runs on save hook could handle VCS as long as Bosun was the only entity pushing to the repo

Consumer

Alert Handling Workflow

- Bosun incident (alert) handling workflow is non-traditional:
 - Incidents only re-notify on severity escalation, or if escalations have been set up for alerts that have not been acknowledged by a human
 - Humans must close alerts
 - There are no “normal notifications”

Results of that design

- Requires discipline
- Works well in that case, not aware of incidents where we did not get an alert because it was not closed
- Lack of up notifications means the anti-pattern of “alert, but not going to look because got an up alert” is avoided. Also inverse of up condition does not always mean “okay”
- No need for flapping detection
- Results in far less incident notifications

Downsides

- Not suitable for all situations (we have another per alert option that is doesn't go through the state flow, but has no flapping prevention)
- Not suitable for all teams (i.e. small number of high level alerts)

Grafana at Stack

- Since Bosun's visualization is limited to alerts - needed to solution for self-service dashboards for SREs and Devs
- Installed it, pointed it at OpenTSDB
- People just started using it to create dashboards

Bosun Plugin for Grafana

- Grafana (or visualization in general) is more usable than bosun, so turn Grafana authors and consumers into Bosun consumers
- Show relevant alerts on dashboards, and handle them there (less places is better)
- Use the expression language to achieve visualization that Grafana can't do

Query Generator

V1: A Text box

Graph General Metrics Axes Legend Display Time range ×

▼ A Query: ?

q("avg:\$ds-avg:rate:os.cpu(host=*bosun*)", "\$start", "")

Alias: ?

≡

👁

🗑

V2: UI

OpenTSDB Query Function Helper

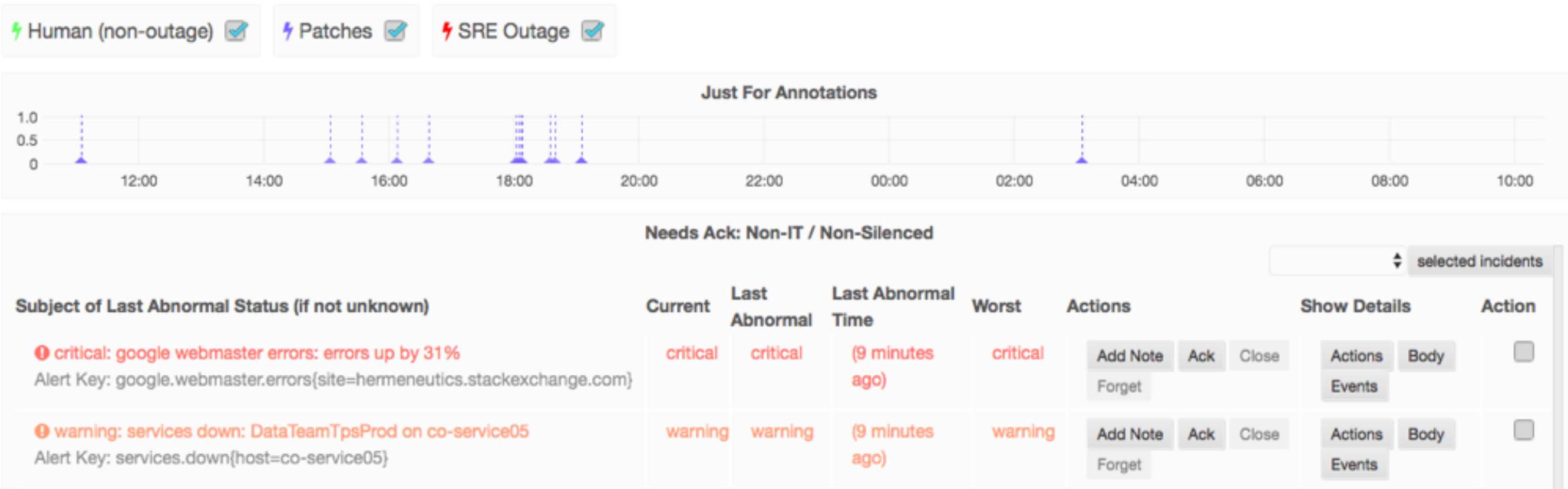
Metric	os.net.bytes	
direction		Group By ▾
host		Group By ▾
iface		Group By ▾
iname		Group By ▾
Generated Query	q("avg:\$ds-avg:rate{counter,,1}:os.net.bytes{}{}", "\$start'	Add To Query

Unit	bytes per second	Set Y Axis Label
Rate Type	counter	
Desc	The rate at which bytes are sent or received over the network interface.	

No place to expose to user



View Manage Incidents in Grafana



⚡ Human (non-outage) ☒⚡ Patches ☒⚡ SRE Outage ☒

Just For Annotations

Incident: #216674

Notes: This alert will watch for critical services and alert if they have been down for a few minutes.

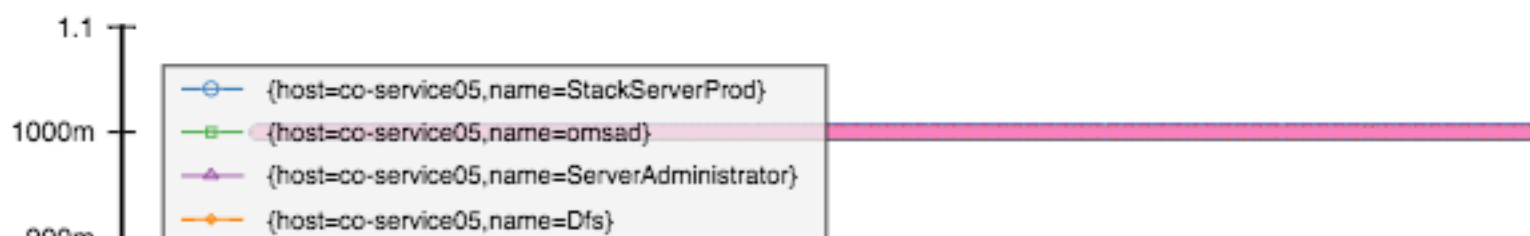
Tags

host co-service05

Services: (1 down, 7 total)

Host	Service	Status
co-service05	StackServerProd	up
co-service05	omsad	up
co-service05	ServerAdministrator	up
co-service05	Dfs	up
co-service05	DFSR	up
co-service05	TagServerProd	up
co-service05	DataTeamTpsProd	down

Graph



Subject of Last Abnormal Status (if any)

❗ critical: google webmaster errors

Alert Key: google.webmaster.errors

❗ warning: services down: DataTeamTpsProd

Alert Key: services.down{host=co-service05}

❗ warning: services down: DataTeamTpsProd

Alert Key: services.down{host=co-service05}

❗ warning: services down: DataTeamTpsProd

Alert Key: services.down{host=co-service05}

❗ warning: google webmaster errors

Alert Key: google.webmaster.errors

Do Crazy Stuff

(if power user)

route: Questions/Show ▾ TPS Source: ny-logsq101 ▾ Number Of Period Look Backs: 3 ▾ Bucket Size For Elastic Percentile: 10m ▾  Human ☐

Total Time (MS) in Section From Logs for Current Period

▼	Tr	ASP	SQL	Redis	HTTP Calls	Tag Engine
-	3.7 Mil	2.6 Mil	577.7 K	14.6 K	365.0	-


Table

General

Metrics

Options

Time range

▼ A Query: 

```
$index = esdaily("CreationDate", "trafficlogs-", "2006.01.02")
$filter = esand(esregexp("RouteName", "$route"), esquery("", "ResponseCode:>199 AND ResposneCode<300"))
$totalTr = sum(esstat($index, "", $filter, "Tr", "sum", "$ds", "$start", ""))
$totalASP = sum(esstat($index, "", $filter, "AspNetDurationMs", "sum", "$ds", "$start", ""))
$totalSQL = sum(esstat($index, "", $filter, "SqlDurationMs", "sum", "$ds", "$start", ""))
$totalRedis = sum(esstat($index, "", $filter, "RedisDurationMs", "sum", "$ds", "$start", ""))
$totalHTTP = sum(esstat($index, "", $filter, "HttpDurationMs", "sum", "$ds", "$start", ""))
$totalTag = sum(esstat($index, "", $filter, "TagEngineDurationMs", "sum", "$ds", "$start", ""))
leftjoin("", "Tr,ASP,SQL,Redis,HTTP Calls,Tag Engine", $totalTr, $totalASP, $totalSQL, $totalRedis, $totalHTTP, $totalTag)
```

Alias: 

series alias



Has it worked?

- Sort of - helps to create better dashboards and more eyes on alerts
- But hasn't really created significantly more bosun authors at Stack Overflow

Future Usability?

- Usability:
 - More Training, Documentation
 - Simple use case: Use Grafana alerting UI to create Bosun alerts?
 - IDE Usability: Autocompletion, integrated documentation?
 - Refactor state machine to smooth out alert handling workflow

What are the solutions?

Work backwards from what we are trying to communicate when creating dashboards and alerts

Over communicate early in the pipeline

Think of the audience, think of what they know and don't - and what you want them to know

Iterate on metrics, alerts, and dashboards in terms of how effectively they communicate

Focus on human users more when building monitoring

Questions

4616

###

4617

End of Alerts

4618

###

4619

