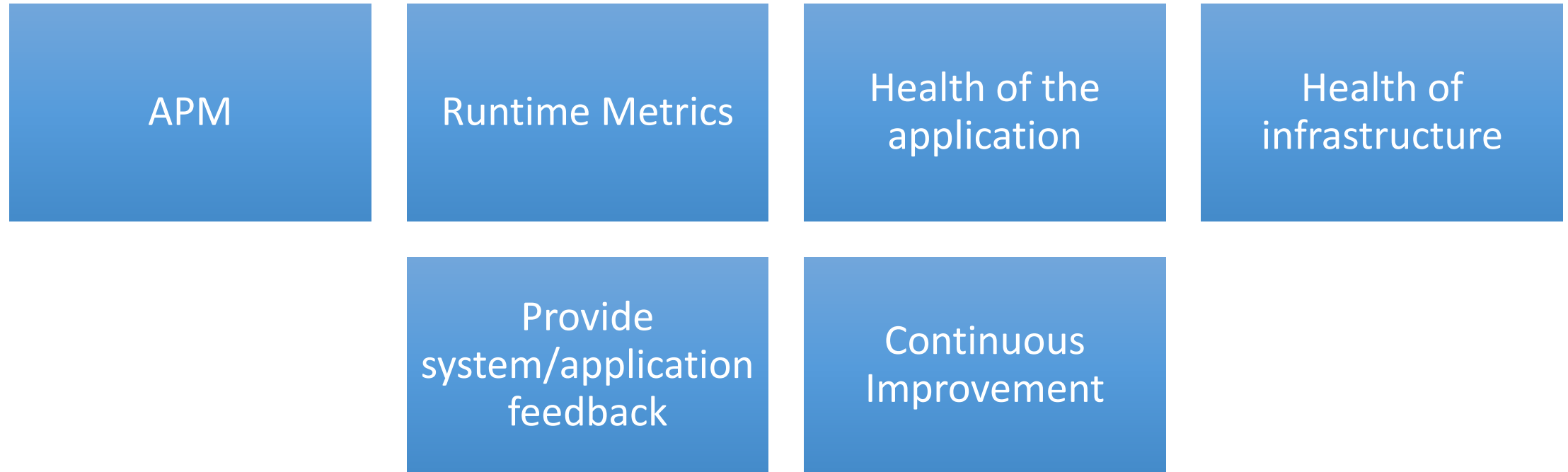


# Application Performance Monitoring using DataDog

Case Study of Performance monitoring for a client in the Media and  
Mobile domain

- Ensure that a software application processes and performs in an expected manner and scope



# What is Datadog?

- Monitoring as a Service
- Agent Based
- Python
- Integrations
- Dashboards
- Tagging
- Alerts
- Checks





redis



docker



Jenkins



mongoDB



loggly








ubuntu


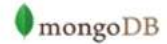





# Challenge

- Monitor System Health
- Monitor Redis, MongoDB
- Application (API Server) runs as docker container
- *Monitor application performance*
- Logging slowed down performance
- *Rapid development, No feedback mechanism about the application performance*
- *Insight in application needed by management*
- *Insight needed by support, devops and developers*

## Integration Dashboards

Add Integration +

↓	Name ↑
☆ 	Amazon – ElastiCache
☆ 	Amazon SNS
☆ 	Amazon SQS
☆ 	AWS
☆ 	AWS – EBS

☆ 	Docker
☆ 	MongoDB – Overview
☆ 	New Relic – Overview
☆ 	Redis – Overview
☆ 	System – Disk I/O
☆ 	System – Networking
☆ 	System – Overview

☆ AWS Edit Board **BETA**

Template Variables [How does this work?](#)

Szone   Template variables

Selb

Latest status updates

- a** Service is operating normally: [RESOLVED] Increased API error rates 1 day ago
- a** Informational message: Increased API error rates 1 day ago
- a** Service is operating normally: Increased API Error Rate 6 days ago

ELBs

Requests	Healthy hosts
<b>9/s</b>	<b>9</b>
Latency	Unhealthy
<b>588ms</b>	<b>0</b>

ELB requests per second

ELB Latency (ms)

EC2 Instances

Active EC2 Instances	Max utilization
<b>29</b>	<b>28%</b>
Average Utilization	Min Utilization
<b>15%</b>	<b>12%</b>

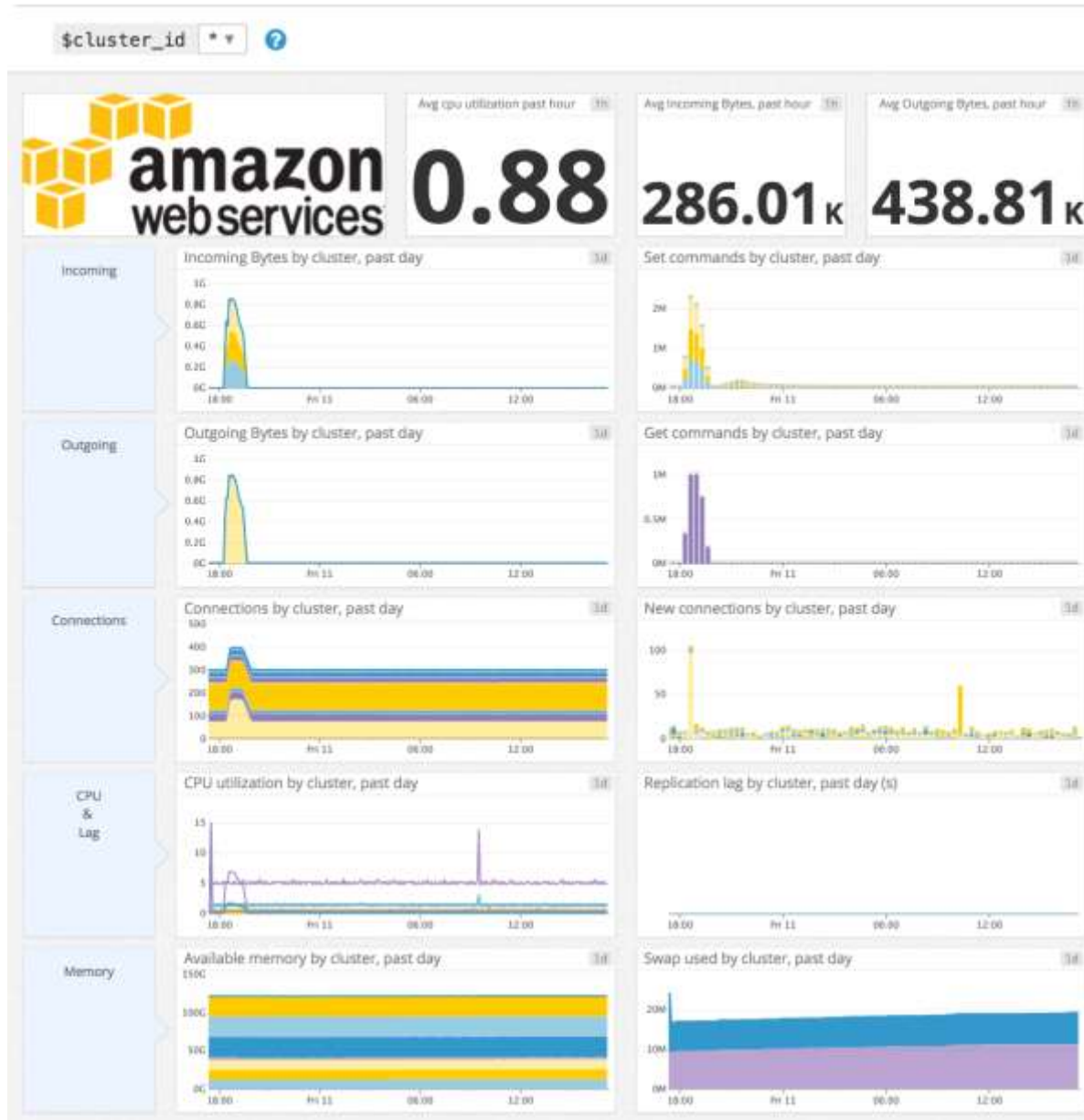
CPU Utilization by instance (%)

EBS Volumes

EBS Volumes	Worst Q Length
<b>35</b>	<b>1</b>
EBS Overall IOps	Avg Q Length
<b>60k</b>	<b>0</b>

EBS IOps by volume

EBS queue length by volume



## ☆ System - Overview



\$scope

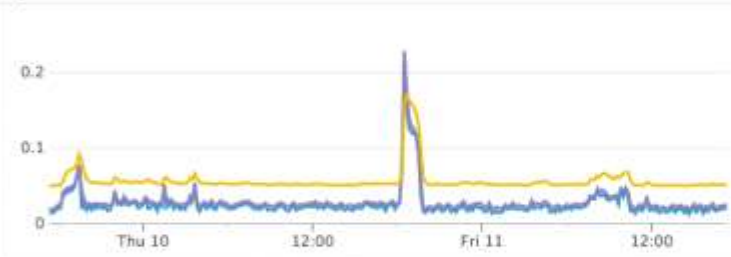
\$scope



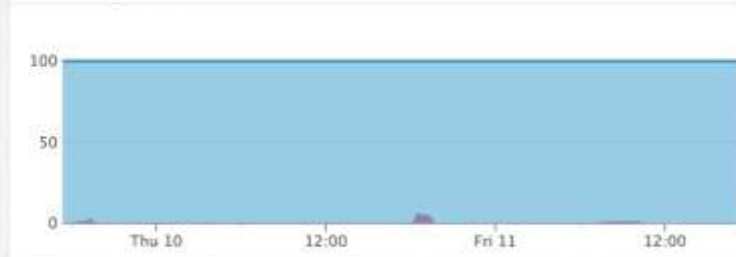
Show 2d The Past 2 Days



### System load



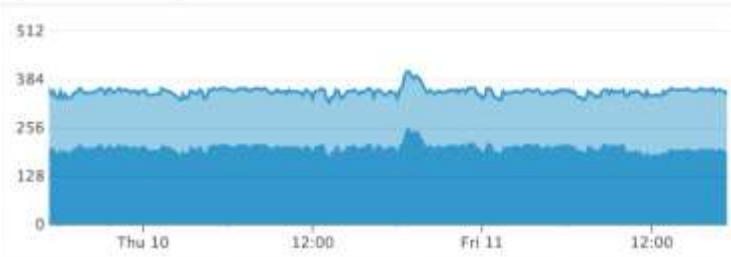
### CPU usage (%)



### I/O wait (%)



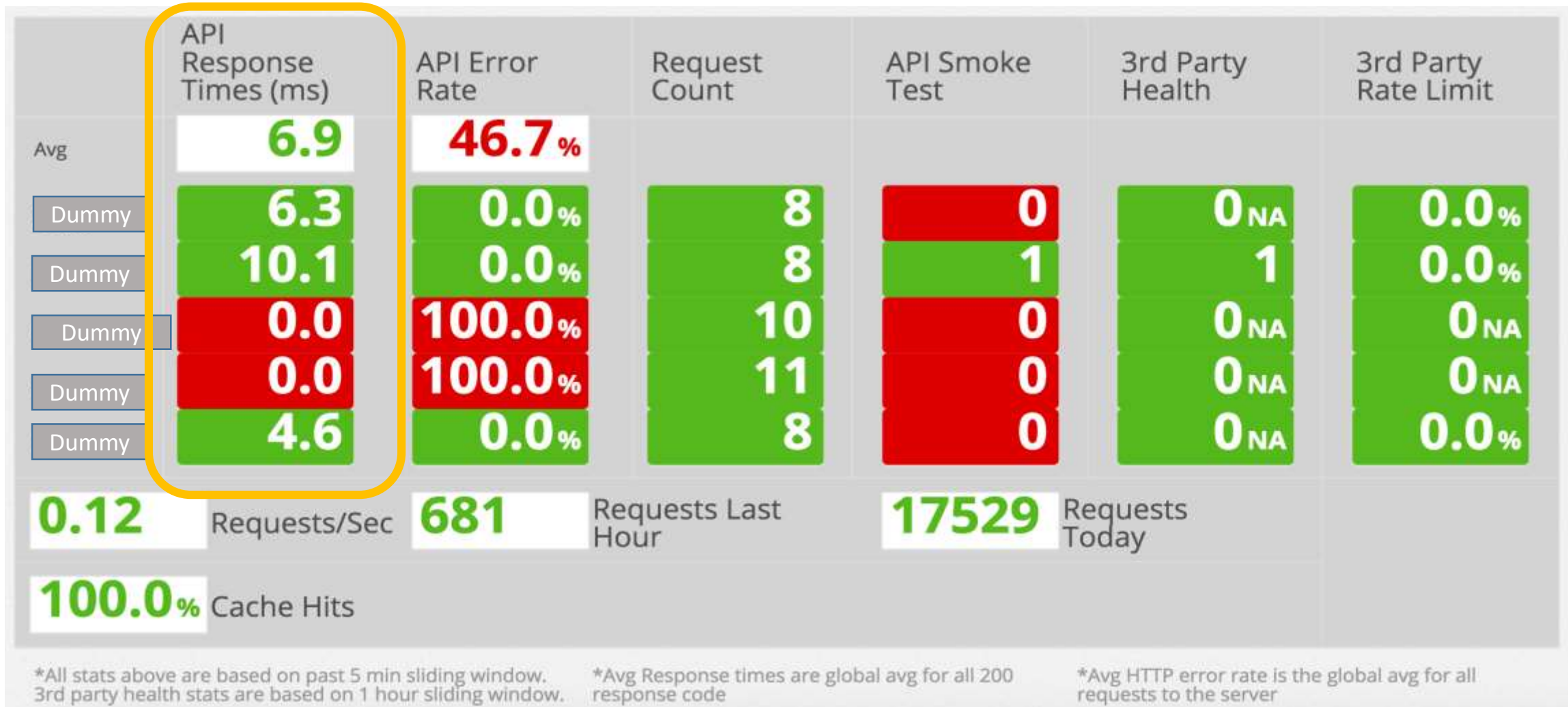
### System memory

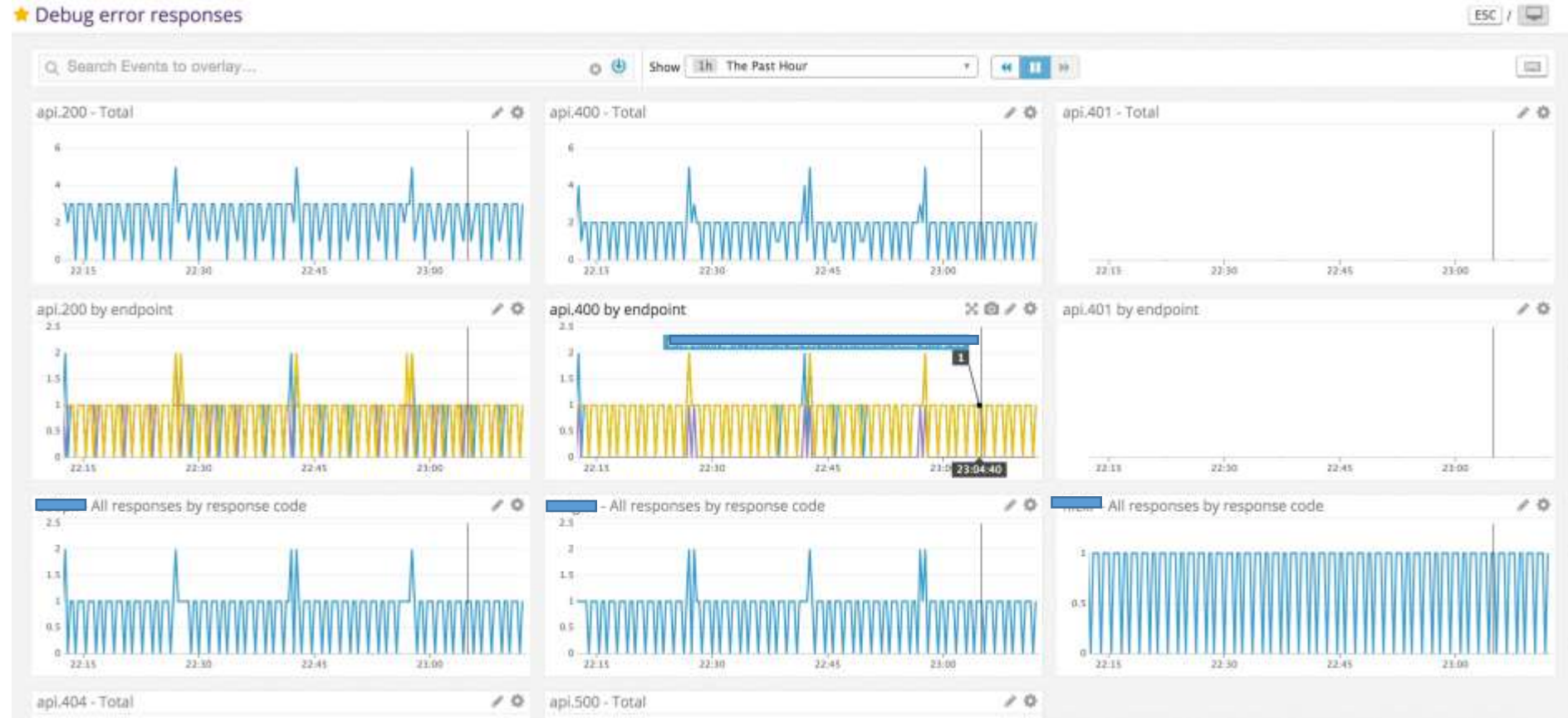


### Network traffic (kb/s)



- How to impress the management?! 😊
- How to deal with X environments and Y versions?
- How to measure performance, without affecting performance?





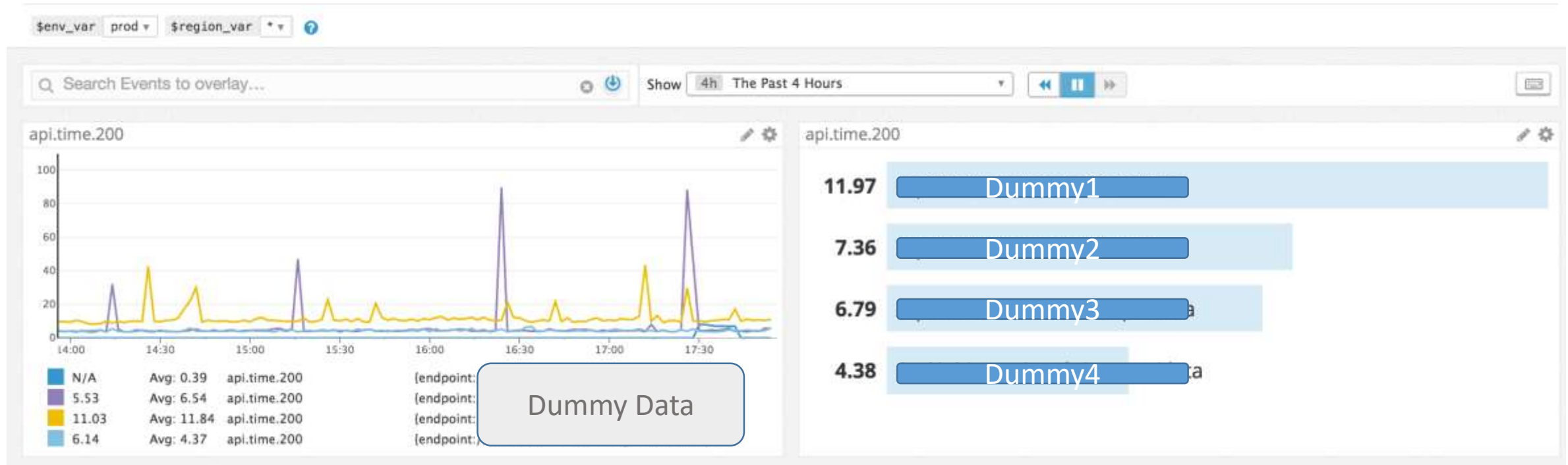
Which endpoint is having more errors? Which error codes are being thrown? Search logs in Loggly if needed.

- We can code:
  - Instance configuration
  - Infrastructure
  - Deployments
- Why not monitoring?!
- Datadog has great API

```
1  var StatsD = require('node-dogstatsd').StatsD;  
2  var dogstatsd = new StatsD();  
3  
4  var tag1 = "env:" + process.env.NODE_ENV;  
5  var tag2 = "endpoint:dummy";  
6  
7  dogstatsd.increment("task.hits", 1, [ tag1, tag2 ]);|
```

```
1  if (!(api.config.env.isLocal || api.config.env.isTest)) {  
2      var tag1 = "response_code:" + raw.responseHttpCode;  
3      var tag2 = "env:" + process.env.NODE_ENV;  
4      var tag3 = "endpoint:" + raw.parsedURL.pathname;  
5  
6      var metric = raw.parsedURL.pathname;  
7  
8      // Send API metrics (count)  
9      metric = "api." + raw.responseHttpCode;  
10     dogstatsd.increment(metric, 1, [ tag3, tag2 ]);  
11  
12     // Send response times  
13     metric = "api.time." + raw.responseHttpCode;  
14     dogstatsd.gauge(metric, data.duration, [ tag2, tag3 ]);  
15 }
```

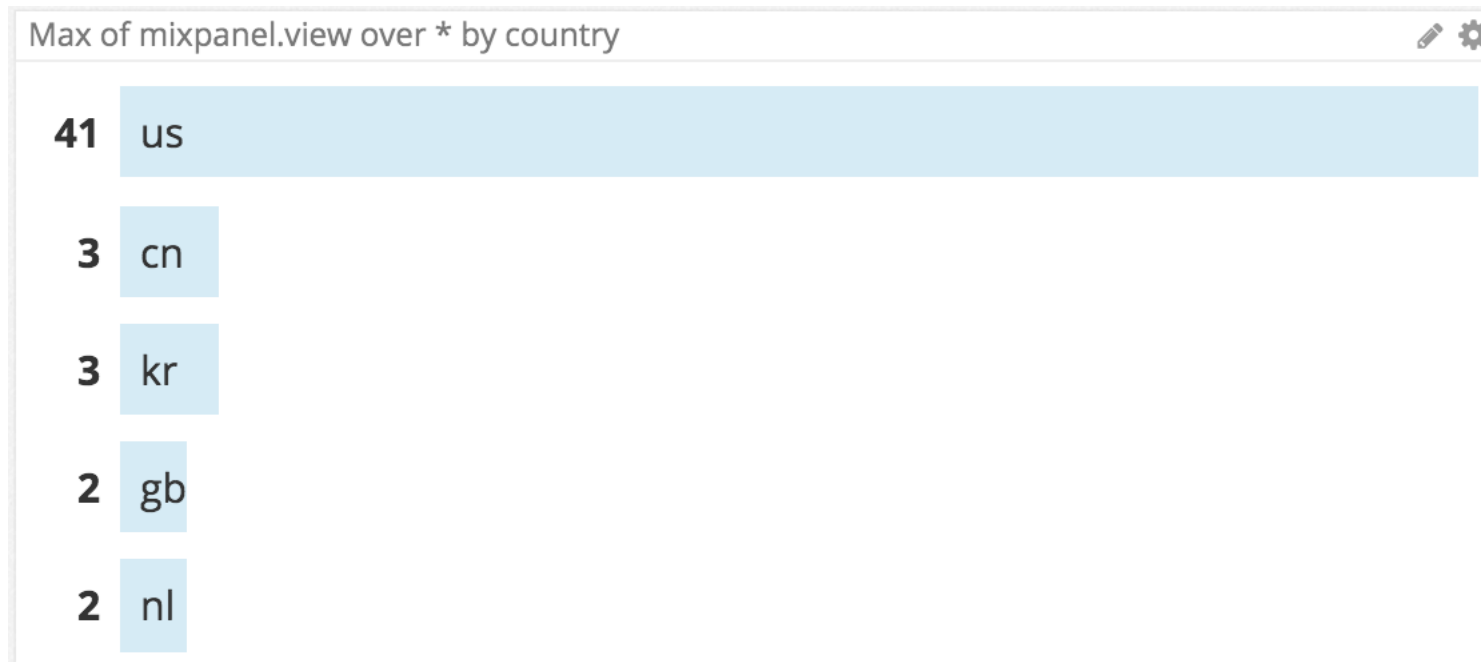
# Plot it!



# What about performance while tracking performance?

- Metrics from code are sent to local datadog agent using UDP
- Local datadog agent syncs the metrics to the datadog server
- Datadog dashboard reflects the metrics with some delay
- Application performance does not get affected

- Metrics are sent from device (mobile/tablet/web) to MixPanel
- Real Data
- Integrate MixPanel with DataDog



Top 5 users of the application by Country

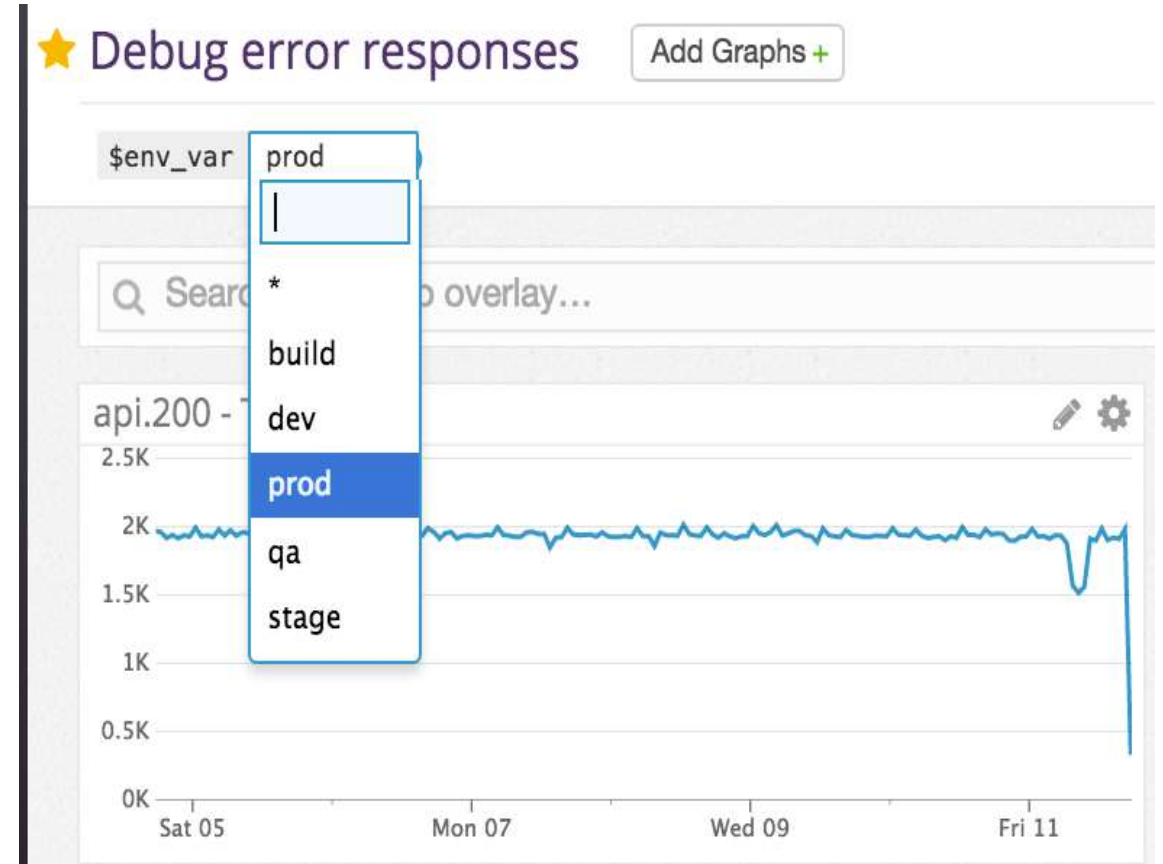
```
1  if (!(api.config.env.isLocal || api.config.env.isTest)) {  
2      var tag1 = "response_code:" + raw.responseHttpCode;  
3      var tag2 = "env:" + process.env.NODE_ENV;  
4      var tag3 = "endpoint:" + raw.parsedURL.pathname;  
5  
6      var metric = raw.parsedURL.pathname;  
7  
8      // Send API metrics (count)  
9      metric = "api." + raw.responseHttpCode;  
10     dogstatsd.increment(metric, 1, [ tag3, tag2 ]);  
11  
12     // Send response times  
13     metric = "api.time." + raw.responseHttpCode;  
14     dogstatsd.gauge(metric, data.duration, [ tag2, tag3 ]);  
15 }
```

- Inherited from Integrations
- Custom tags

The screenshot displays the Datadog configuration interface for a time series graph. At the top, a line graph shows 'system.load.1' over a period from Saturday, Dec 05 to Friday, Dec 11. The y-axis ranges from 0 to 0.15. Below the graph, three numbered steps guide the user through configuration:

- 1 Select your visualization**: A row of tabs includes 'Timeseries' (selected), 'Heatmap', 'Distribution', 'Toplist', 'Change', and 'Hostmap'.
- 2 Choose metrics and events**: A configuration box shows 'Get system.load.1 from env:'. A dropdown menu is open, listing environment tags: 'env:qa', 'env:dev', 'env:prod', 'env:build', 'env:stage', and '\$env\_var' (highlighted). Buttons for '+ Add Metric' and '+ Overlay' are visible. The right side of the box includes 'Graph Primer', 'Share', 'JSON', 'Edit', and 'Advanced ...'.
- 3 Give your graph a title (or leave blank for suggested title)**: A text input field contains the suggested title 'Avg of system.load.1 over \*'. At the bottom right are 'Cancel', 'Preview', and 'Save' buttons.

- Dashboard **variables**
- Dynamically explore metrics



API Server is responding very slowly on PROD env on {{host.name}}

Edit Status

1m The Past Month

Original Data Monitor View



## 1 Define the metric

Source Edit

Define the monitor's query.

```
(avg:api.time.400{prod} + sum:api.time.401{prod} + avg:api.time.402{prod} + avg:api.time.409{prod} + avg:api.time.200{prod} + avg:api.time.500{prod} + 0) / 1000
```

## 2 Set alert conditions

**Threshold Alert** Change Alert An alert is triggered whenever a metric crosses a threshold.

Trigger when the metric is  the threshold  during the last

Alert threshold:  (3)

- Collect metrics from datadog agent check
- Out of the box agent checks
- Custom agent checks in Python
- Interesting use cases:
  - Keep alive check (Service is up)
  - Network Check (HTTP, TCP)
  - Validate response for expected data

## Using Datadog **we could...**

- Monitor dynamic infrastructure
- Monitor system health
- Monitor application availability
- Monitor application performance
- Show application & infra health graphically
- Provide feedback about health of system

# GET IN TOUCH WITH US

Go Ahead, Don't Hesitate!



<http://www.whitehedge.com/devops.html>

<http://www.whitehedge.com/docker-microservices/>