



Logging

“that makes DevOps happy”

By

Reid Morrison

Software Architect

Clarity Services, Inc.

Stock Rails

Started GET "/" for 127.0.0.1 at 2016-02-16 12:13:20 -0500

Processing by StoreController#index as HTML

Cart Load (0.1ms) SELECT "carts".* FROM "carts" WHERE "carts"."id" = ? LIMIT 1 [["id", 9]]

Product Load (0.2ms) SELECT "products".* FROM "products" ORDER BY "products".updated_at DESC LIMIT 1

Product Load (0.1ms) SELECT "products".* FROM "products" ORDER BY "products".title ASC

Rendered store/index.html.haml within layouts/application (3.3ms)

LineItem Exists (0.1ms) SELECT 1 AS one FROM "line_items" WHERE "line_items"."cart_id" = ? LIMIT 1 [["cart_id", 9]]

LineItem Load (0.1ms) SELECT "line_items".* FROM "line_items" WHERE "line_items"."cart_id" = ? [["cart_id", 9]]

Rendered collection (0.0ms)

Rendered carts/_cart.html.erb (1.4ms)

Completed 200 OK in 18ms (Views: 15.4ms | ActiveRecord: 0.5ms)

Started GET "/assets/application.css?body=1" for 127.0.0.1 at 2016-02-16 12:13:20 -0500

rocketjob.io

```
Started GET "/assets/application.css?body=1" for 127.0.0.1 at 2016-02-16 12:13:20 -0500
```

```
Started GET "/assets/application.css?body=1" for 127.0.0.1 at 2016-02-16 12:13:20 -0500
```

Add context

```
2016-02-16 12:19:07.781921 I [26202:70288050797680] Rails -- Started GET "/" for 127.0.0.1 at 2016-02-16 12:19:07 -0500
2016-02-16 12:19:07.783712 I [26202:70288050797680] ActionController::Base -- Processing by StoreController#index as HTML
2016-02-16 12:19:07.784378 D [26202:70288050797680] ActiveRecord::Base -- Cart Load (0.1ms) SELECT "carts".* FROM "carts" WHERE "carts"."id" = ? LIMIT 1 [["id", 9]]
2016-02-16 12:19:07.789272 D [26202:70288050797680] ActiveRecord::Base -- Product Load (0.2ms) SELECT "products".* FROM "products" ORDER BY "products"."updated_at" DESC LIMIT 1
2016-02-16 12:19:07.789643 D [26202:70288050797680] ActiveRecord::Base -- Product Load (0.2ms) SELECT "products".* FROM "products" ORDER BY "products"."title" ASC
2016-02-16 12:19:07.792261 I [26202:70288050797680] ActionView::Base -- Rendered store/index.html.haml within layouts/application (4.0ms)
2016-02-16 12:19:07.800406 D [26202:70288050797680] ActiveRecord::Base -- LineItem Exists (0.2ms) SELECT 1 AS one FROM "line_items" WHERE "line_items"."cart_id" = ? LIMIT 1 [["cart_id", 9]]
2016-02-16 12:19:07.801725 D [26202:70288050797680] ActiveRecord::Base -- LineItem Load (0.1ms) SELECT "line_items".* FROM "line_items" WHERE "line_items"."cart_id" = ? [["cart_id", 9]]
2016-02-16 12:19:07.801787 I [26202:70288050797680] ActionView::Base -- Rendered collection (0.0ms)
2016-02-16 12:19:07.802581 I [26202:70288050797680] ActionView::Base -- Rendered carts/_cart.html.erb (1.2ms)
2016-02-16 12:19:07.802845 I [26202:70288050797680] ActionController::Base -- Completed 200 OK in 19ms (Views: 17.6ms | ActiveRecord: 0.7ms)
2016-02-16 12:19:07.839937 D [26202:70288050797680] Rails --
2016-02-16 12:19:07.839970 D [26202:70288050797680] Rails --
2016-02-16 12:19:07.840087 I [26202:70288050797680] Rails -- Started GET "/assets/application.css?body=1" for 127.0.0.1 at 2016-02-16 12:19:07 -0500
```

Best Practices – Appropriate Log Levels

Verify correct log levels

logger.info '**Fatal error occurred**'

logger.error '**Started processing normally**'

logger.debug '**Important log message**'

```
2016-02-18 03:11:11.333214 I [38464:Main] User -- Fatal error occurred
2016-02-18 03:11:11.334677 E [38464:Main (irb):64] User -- Started processing normally
2016-02-18 03:11:11.335967 D [38464:Main] User -- Important log message
```

Best Practices – Use Blocks

Use blocks to prevent unnecessary evaluation

```
logger.debug do  
  count = results.inject(0) {|sum, i| i+sum }  
  "A total of #{count} were processed"  
end
```

Best Practices – Measure Everything

```
# Measure the duration of any block of code
count = logger.benchmark_info('Counting users') do

  User.where('created_at <= ?', date).count

end
```

```
2016-02-18 02:51:12.289139 I [38464:70242051291680] (314.7ms) Google — Counting users
```

Best Practices – Elastic Logging

```
# Only log when the block takes longer than 50ms
value = logger.benchmark_warning(
    'Memcache call was slow',
    min_duration: 50
) do
    memcache.get('key')
end
```


Best Practices - Class specific loggers

```
class Supplier
```

```
  include SemanticLogger::Loggable
```

```
  def self.some_class_method
```

```
    logger.debug 'logger is accessible from class methods'
```

```
  end
```

```
  def call_supplier
```

```
    logger.debug 'logger is accessible from instance methods'
```

```
  end
```

```
end
```

```
Supplier.some_class_method
```

```
Supplier.new.call_supplier
```

```
2016-02-18 04:00:37.036910 D [38464:Main] Supplier -- logger is accessible from class methods
2016-02-18 04:00:37.039196 D [38464:Main] Supplier -- logger is accessible from instance methods
```

Best Practices - Tagging

Add tags to any block of code

```
logger.tagged('127.0.0.1', 'jbloggs') do
  logger.info 'Started processing normally'
  logger.debug 'Calling supplier'
  logger.trace 'Received Response', response
end
```

```
2016-02-18 03:08:34.134733 I [38464:Main] [127.0.0.1] [jbloggs] User -- Started processing normally
2016-02-18 03:08:34.134754 D [38464:Main] [127.0.0.1] [jbloggs] User -- Calling supplier
2016-02-18 03:08:34.134763 T [38464:Main] [127.0.0.1] [jbloggs] User -- Received Response -- "123, 14543, data, valid"
```

Best Practices - Tagging

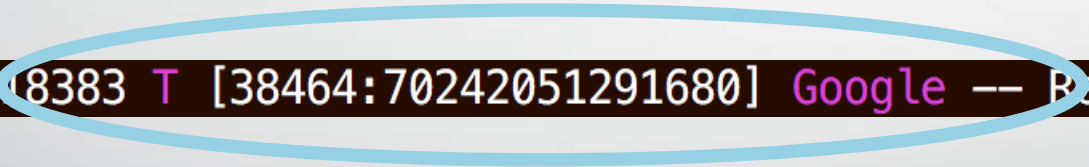
Add the ip-address and a tracking number to every Rails message:

```
config.log_tags = [  
  :remote_ip,  
  lambda do |request|  
    request.headers["Tracking-Number"] = random_tracking_number  
  end  
]
```

Best Practices – Trace level

Log low level messages as trace

```
logger.trace('Received', raw_response)
```



2016-02-18 02:35:40.408383 T [38464:70242051291680] Google -- Received -- "123, 14543, data, valid"

Best Practices – Semantic Messages



```
logger.error("Oops external call failed, result: #{result}, reason_code:  
#{reason_code}")
```

```
logger.error(  
  "Oops external call failed",  
  result: :failed,  
  reason_code: -10  
)
```

```
2016-02-18 02:45:58.355975 E [38464:70242051291680 (irb):9] Google -- Oops external call failed - { :reason_code => -10, :result => :failed }
```

Best Practices – Name Threads

```
# Name different threads
```

```
Thread.current.name = 'Main'
```

```
logger = SemanticLogger['Job']
```

```
logger.info "Handing work to processing thread"
```

```
Thread.new do
```

```
  Thread.current.name = 'Processor'
```

```
  logger = SemanticLogger['Processor']
```

```
  logger.info "Started processing"
```

```
end
```

For threading:

Parallel Minion

```
2016-02-18 02:58:51.272253 I [38464:Main] Job — Handing work to processing thread
2016-02-18 02:58:51.275849 I [38464:Processor] User — Started processing
```

Best Practices – Log Rescued Exceptions

```
# Log rescued exceptions
```

```
begin
```

```
  google.search('semantic logger')
```

```
rescue StandardError => exc
```

```
  logger.error 'Failed calling supplier', exc  
  'No data available'
```

```
end
```

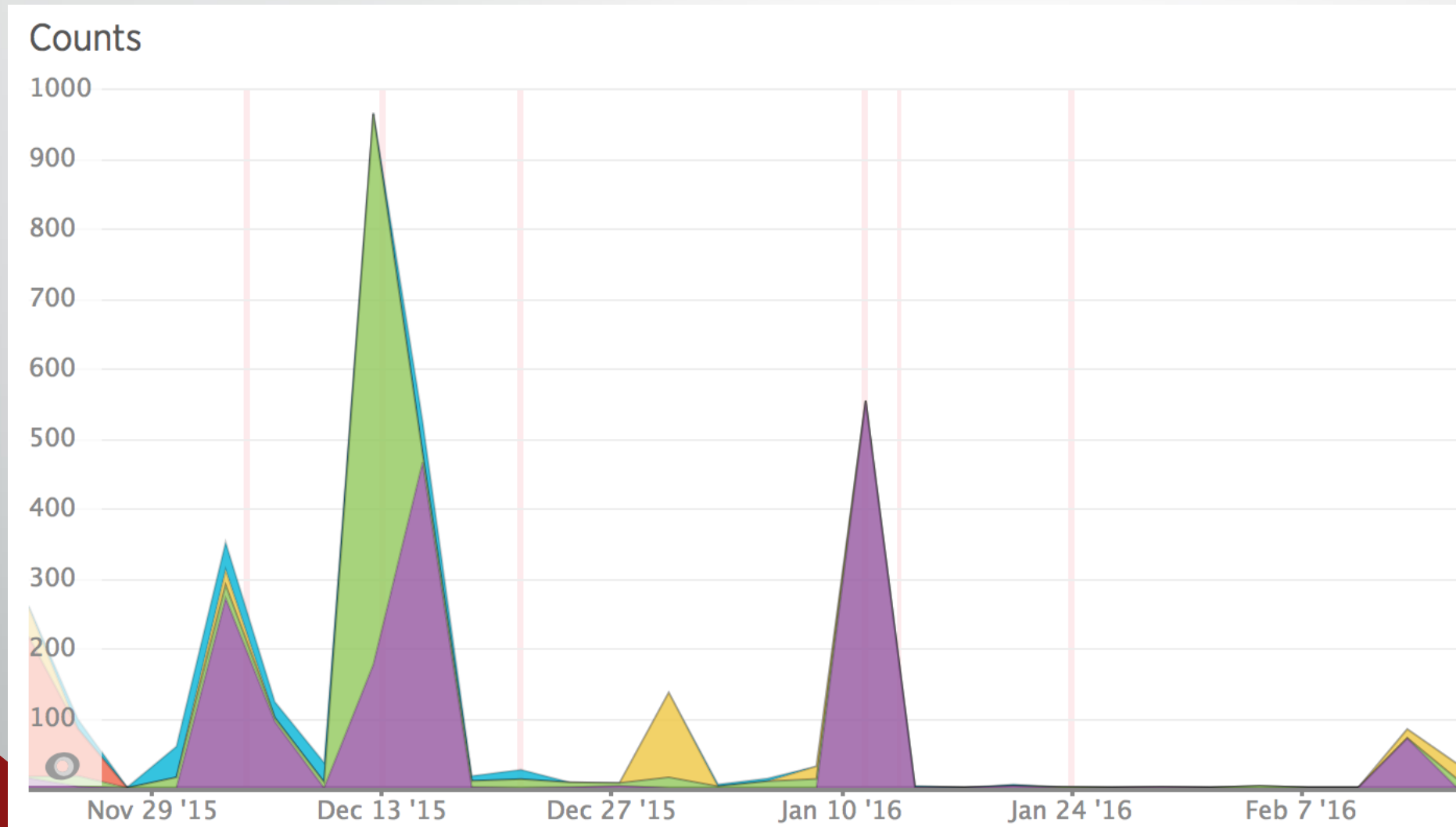
```
2016-02-18 04:16:24.377592 E [38464:Main (irb):122] User -- Failed calling supplier -- Exception: Timeout::Error: Timeout::Error  
(irb):120:in `irb_binding'  
/Users/rmorrison/.rvm/rubies/ruby-2.3.0/lib/ruby/2.3.0/irb/workspace.rb:87:in `eval'  
/Users/rmorrison/.rvm/rubies/ruby-2.3.0/lib/ruby/2.3.0/irb/workspace.rb:87:in `evaluate'  
/Users/rmorrison/.rvm/rubies/ruby-2.3.0/lib/ruby/2.3.0/irb/context.rb:380:in `evaluate'  
/Users/rmorrison/.rvm/rubies/ruby-2.3.0/lib/ruby/2.3.0/irb.rb:489:in `block (2 levels) in eval_input'
```

Best Practices – Count Metrics

Increment google search count metric

```
logger.info(  
    message: 'Called Google',  
    metric: 'google/search'  
)
```


Best Practices - Dashboard for count based metrics



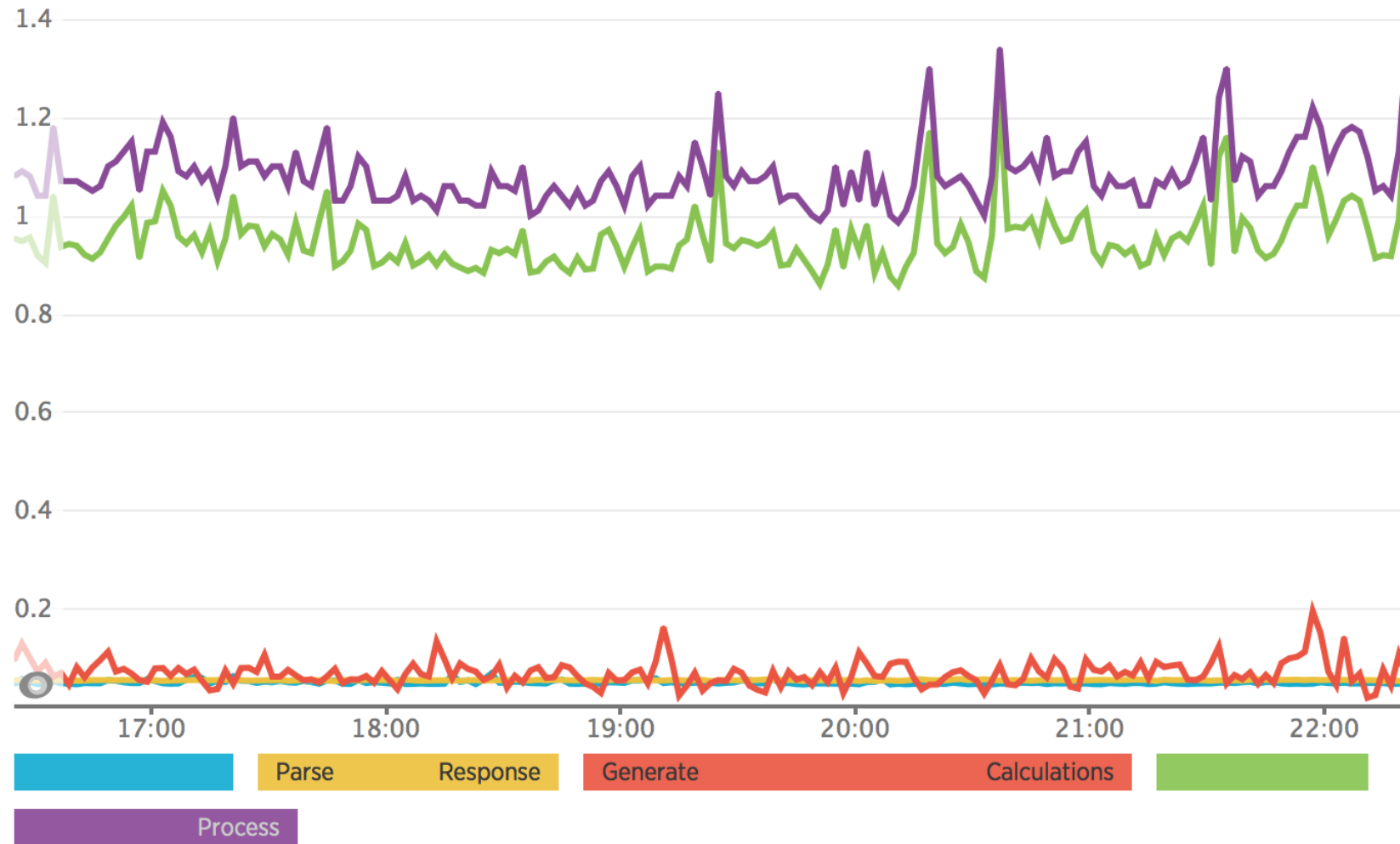
Best Practices – Duration Metrics

```
# Measure the duration of any block of code
count = logger.benchmark info('Counting users',
  metric: 'user/count') do
  User.where('created_at <= ?', date).count
end
```

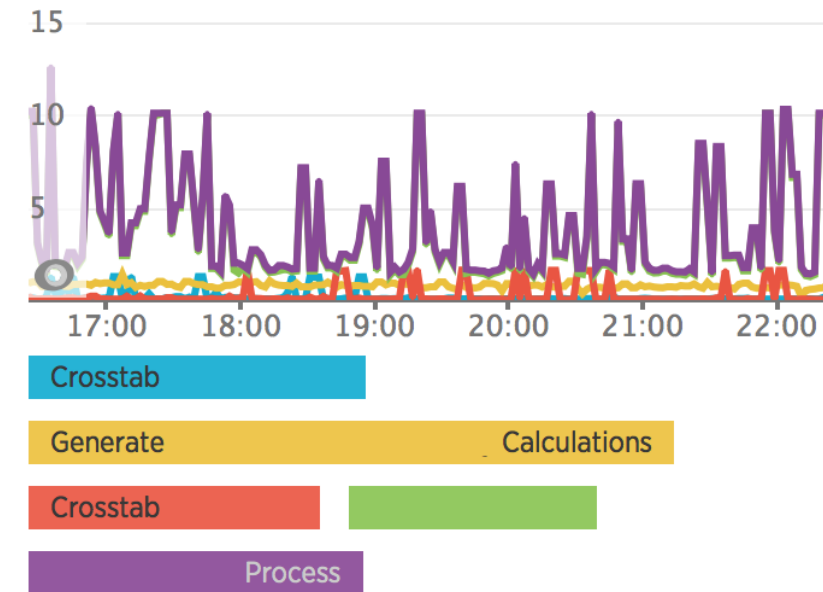
2016-02-18 02:51:12.289139 I [38464:70242051291680] (314.7ms) Google — Counting users

Best Practices - Dashboard from duration Metrics

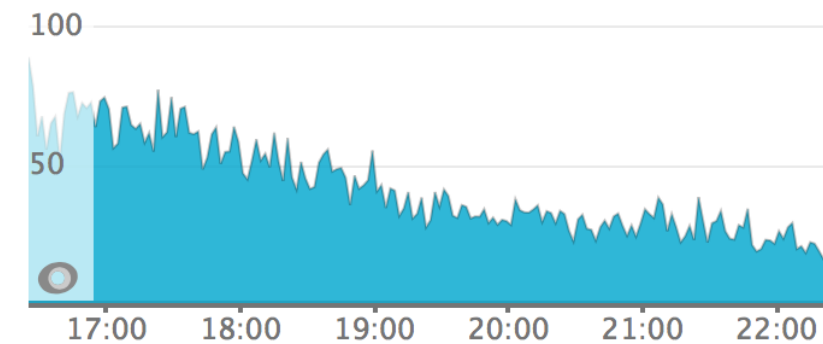
Itemized response times



Max Response Time



Calls per minute



Best Practices - Log Filtering

Configure sensitive parameters which will be filtered
from the log file.

```
Rails.application.config.filter_parameters += [password,  
social_security_number]
```

Best Practices – Centralized Exception & Error Management

- Services
 - Bugsnag
 - NewRelic
 - HoneyBadger
 - Etc...
- Features
 - Unhandled Exceptions
 - Log file errors
 - Alerting
 - First occurrence
 - Error spikes

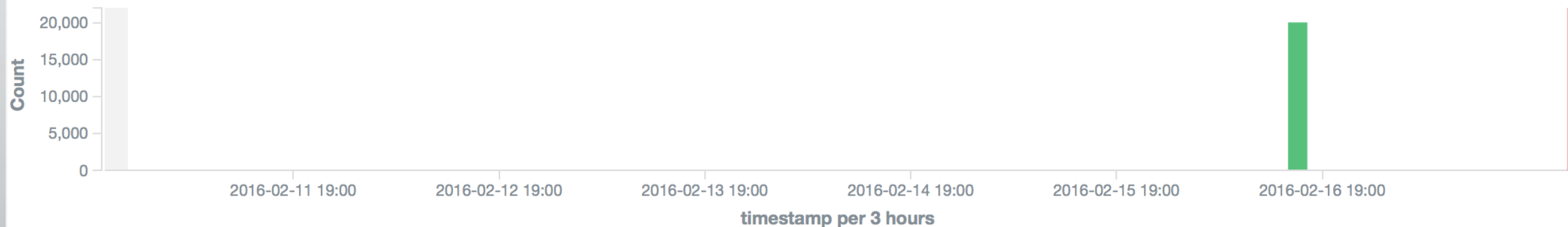


Best Practices – Centralized Logging

- Solutions
 - Elasticsearch & Kibana
 - Splunk
 - Graylog
 - Loggly
 - Logstash
 - MongoDB
 - Etc...
- Features
 - Search by any text keyword
 - Example: tracking number
 - Search by specific field
 - login = 'jbloggs'
 - Dashboards
 - Alerting



20,138 hits

February 10th 2016, 23:42:53.090 - February 17th 2016, 23:42:53.090 — [by 3 hours](#)


^					
Time ▾	duration	level	name	message	
▶ February 17th 2016, 11:50:10.628	0.644ms	info	RocketJob::Jobs::DirmonJob	Completed #perform	
▶ February 17th 2016, 11:45:10.362	1.153ms	info	RocketJob::Jobs::DirmonJob	Completed #perform	
▶ February 17th 2016, 11:40:10.067	1.078ms	info	RocketJob::Jobs::DirmonJob	Completed #perform	
▶ February 17th 2016, 11:35:09.502	1.491ms	info	RocketJob::Jobs::DirmonJob	Completed #perform	
▶ February 17th 2016, 11:30:08.848	0.912ms	info	RocketJob::Jobs::DirmonJob	Completed #perform	
▶ February 17th 2016, 11:25:08.155	0.855ms	info	RocketJob::Jobs::DirmonJob	Completed #perform	

Logging Framework Should Support:

- Multiple processes.
- Multiple threads.
- Measure everything.
- Metrics.
- Centralized Logging.
- Concurrent destinations.
- Many log destinations.
- Appender Neutral.
- Class specific log levels.
- Runtime log level change.
- Capture Ruby thread dumps.
- Performance.
- Battle tested.
- Standalone: semantic_logger.
- Rails: rails_semantic_logger.
- Common API across other loggers.

Questions?

- Semantic Logger
 - http://rocketjob.github.io/semantic_logger/
- Feedback?
 - @reidmorrison
 - <http://linkedin.com/in/reidmorrison>
 - reid@rocketjob.io
- Support
 - <https://gitter.im/rocketjob/support>
 - github.com/rocketjob/semantic_logger
 - Star project on github
- Projects using Semantic Logger:
 - Rocket Job
 - Parallel Minion
 - Data Cleansing
 - JRuby JMS