elastic Meetup Night Class: Centralized Logging

2017-03-16#9b6c82e

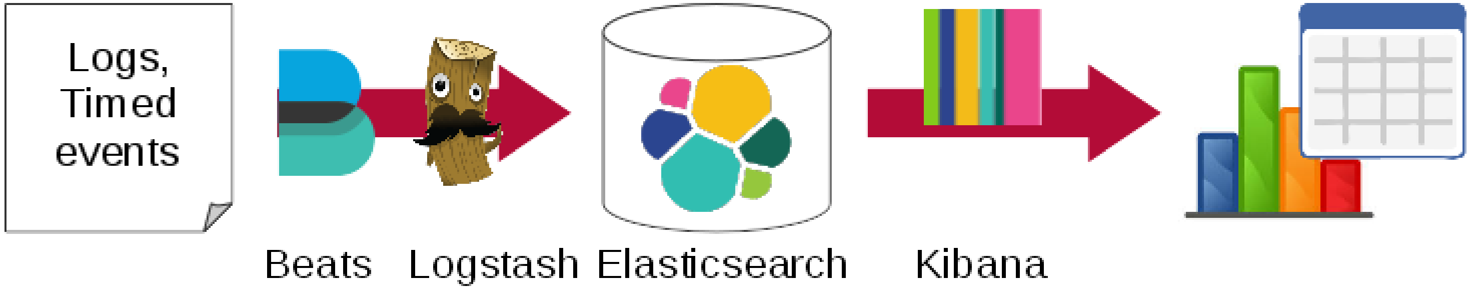
Motivation

- Aggregate log events
  - across applications
  - across processes
  - across machines
- Combine and visualize data

Use Cases

- Business
  - amount and duration of transactions
  - history of operations
  - correlation between applications
- Performance and health
  - amount and duration of requests, request times
  - errors
  - correlation between systems, data bases, applications
- Security
  - fraud detection, intrusion detection
  - tracing

Overview



Beats

- Lightweight agent that pushes data to another destination
- Common for centralized logging: Filebeat
- Implemented in Go
- Other beats:
  - Metricbeat
  - Winlogbeat
  - Packetbeat

Logstash

- Kind of ETL system for transforming event data
- Often used for logs
- Written in JRuby
- Configured in Ruby Syntax
  - Input
  - Filter
  - Output

Elasticsearch

- Stores the log events
- Query DSL used for accessing the data
- Often time based indexing
  - filebeat-2017-03-23
  - Index pattern filebeat-*

Kibana

- Web interface for analyzing data
- Explore data
- Generates requests to Elasticsearch
- Visualizes results: diagrams, maps
- Written in JavaScript

Ingest Node

- Logstash can consume quite some ressources
- Common transformations available in Elasticsearch as well
- Ingestion can happen on separate node

Reading files: Filebeat

- Installation as service or standalone
- Configuration in filebeat.yml
- Prospectors define where to read data from
- Output determines where to send data to

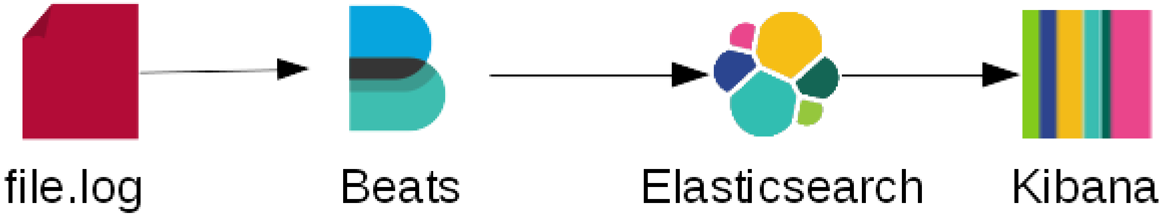
Filebeat: Prospector configuration

```
filebeat.prospectors:

- input_type: log

  paths:
    - /var/log/*.log
    #- c:\programdata\elasticsearch\logs\*
```

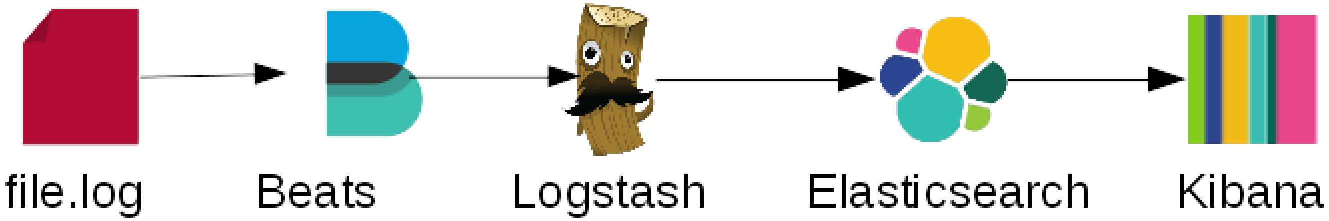
- Includes, excludes
- Multiline configuration

Filebeat: Setup options

- Using Ingest Node to process data



- Using Logstash to process data



Filebeat: Output configuration

- Sending data to Elasticsearch
  - for storing the message as a whole
  - or for processing with Ingest pipeline

```
output.elasticsearch:
  hosts: ["localhost:9200"]

  # Optional protocol and basic auth credentials.
  # username: "elastic"
  # password: "changeme"
```

Filebeat: Output configuration

- Sending data to Logstash
  - for processing and redistribution

```
output.logstash:
  # The Logstash hosts
  hosts: [”localhost:5044”]
```

Reading files: Logstash

- 3 major sections
  - Input: Specifies where to read data from
  - Filter: Different ways to massage the data
  - Output: Data sinks
- Countless plugins for all

Logstash: Input

- Server process for reading beat data

```
input {
  beats {
    port => 5044
  }
}
```

- or reading from file

```
input {
    file {
        path => "/var/log/apache2/access.log”
    }
}
```

Logstash: Grok filter

- Common filter: Grok
  - Regular expressions with predefined patterns

```
filter {
    grok {
        match => { message => "%{COMBINEDAPACHELOG}" }
    }
}
```

Logstash: Output

- Sending data to Elasticsearch

```
output {
    elasticsearch {
        host => ”localhost”
        protocol => ”http”
    }
}
```

- For debugging

```
    stdout {
      codec => rubydebug
    }
```

Ingest Node

- Implementation of some of the Logstash filters
- Pipeline defines processing order
- management via HTTP API
- Pipeline can be defined in Filebeat config

Ingest Node

- Pipeline has name and processors
  - simple case: number followed by a word

```
PUT _ingest/pipeline/simplest-log
{
  "processors" : [
    {
      "grok" : {
        "field": "message",
        "patterns": ["%{NUMBER:line} %{WORD:text}"]
      }
    }
  ]
}
```

- Result: documents with line and text field

Ingest Node

- Configure pipeline in Filebeat

```
output.elasticsearch:
  hosts: [”localhost:9200”]
  pipeline: simplest-log
```

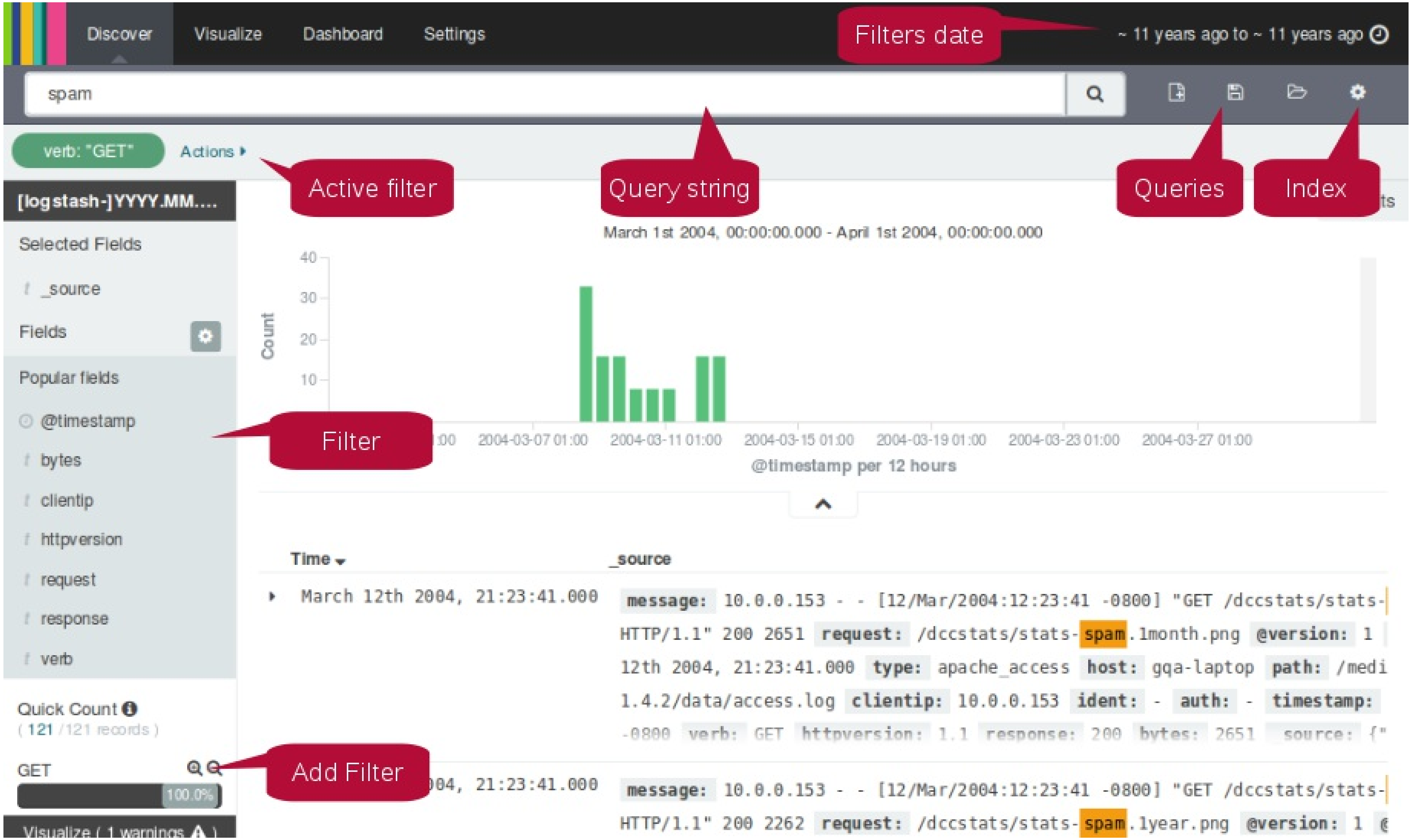
Ingest Node

- Simulation for debugging
- Online tools for creating grok expressions
  - grokdebug https://grokdebug.herokuapp.com/
  - Grok Constructor http://grokconstructor.appspot.com/
- Careful with expensive regular expressions

Kibana

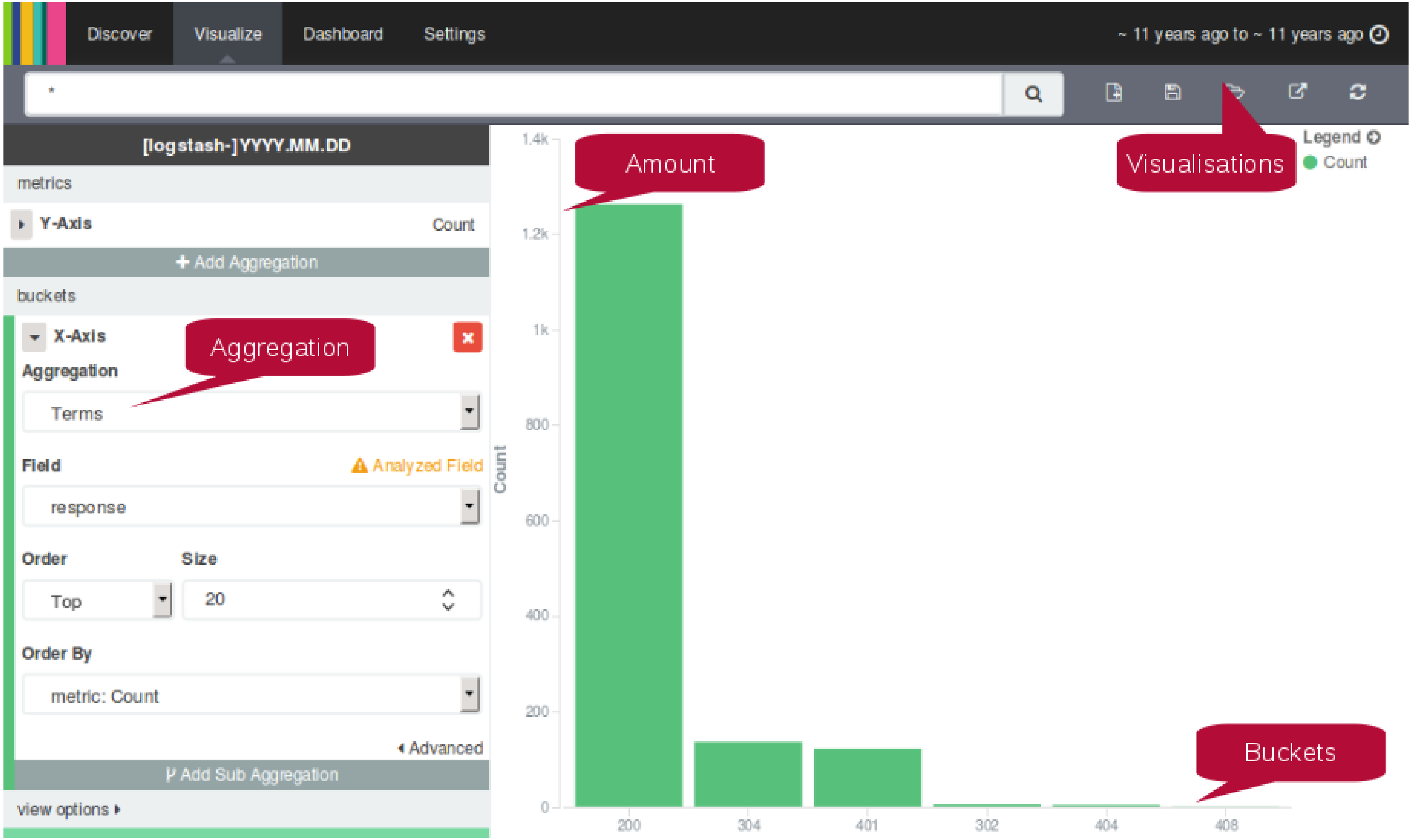
- Search in log events
- Visualize data

Kibana



```
▸ March 12th 2004, 21:23:41.000   message: 10.0.0.153 - - [12/Mar/2004:12:23:41 -0800] "GET /dccstats/stats-
                                  HTTP/1.1" 200 2651 request: /dccstats/stats-spam.1month.png @version: 1
                                  12th 2004, 21:23:41.000 type: apache_access host: gqa-laptop path: /medi
                                  1.4.2/data/access.log clientip: 10.0.0.153 ident: - auth: - timestamp:
                                  -0800 verb: GET httpversion: 1.1 response: 200 bytes: 2651 _source: {"

  )04, 21:23:41.000               message: 10.0.0.153 - - [12/Mar/2004:12:23:41 -0800] "GET /dccstats/stats-
                                  HTTP/1.1" 200 2262 request: /dccstats/stats-spam.1year.png @version: 1 @
```

Kibana

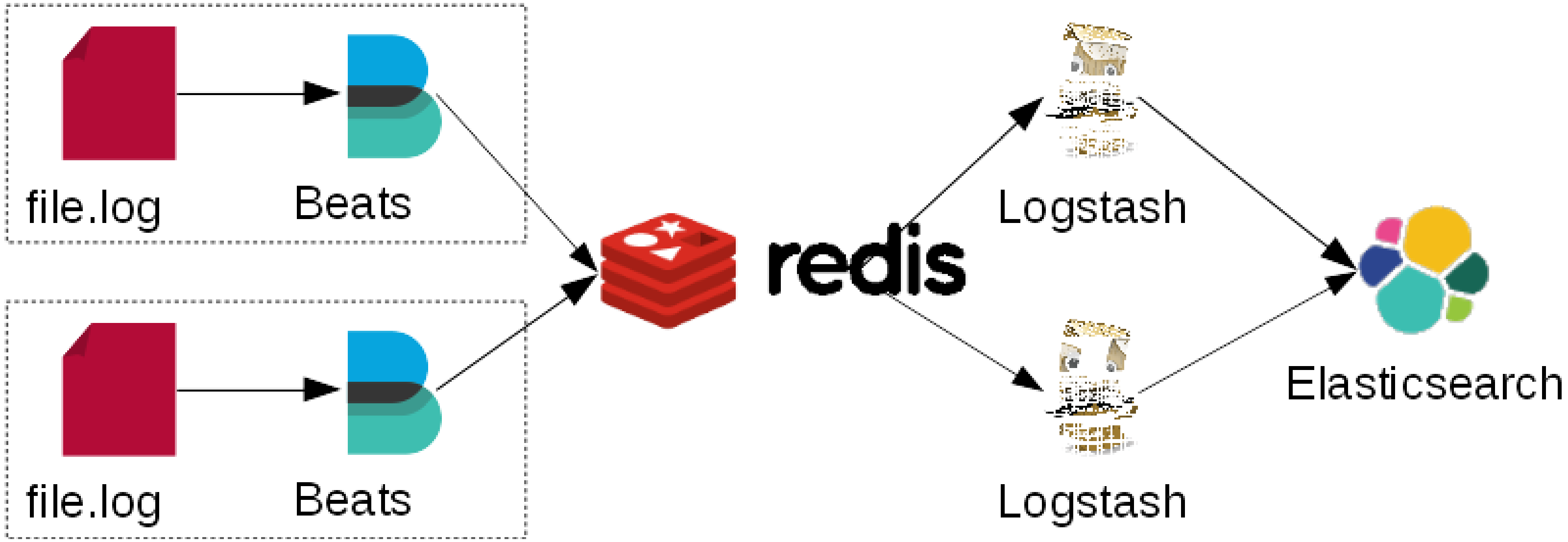


Discover
Visualize
Dashboard
Settings
~ 11 years ago to ~ 11 years ago
*
[logstash-]YYYY.MM.DD
metrics
Y-Axis
Count
+ Add Aggregation
buckets
X-Axis
Aggregation
Terms
Field
⚠ Analyzed Field
response
Order
Size
Top
20
Order By
metric: Count
Advanced
Add Sub Aggregation
view options
Count
1.4k
1.2k
1k
800
600
400
200
0
200
304
401
302
404
408
Legend
● Count
Amount
Visualisations
Aggregation
Buckets

Scaling options

- Backpressure sensitive when sending to Logstash and Elasticsearch
- Buffer for load spikes and for scaling Logstash
  - Redis
  - Kafka

Scaling options


file.log
Beats
file.log
Beats
redis
Logstash
Logstash
Elasticsearch