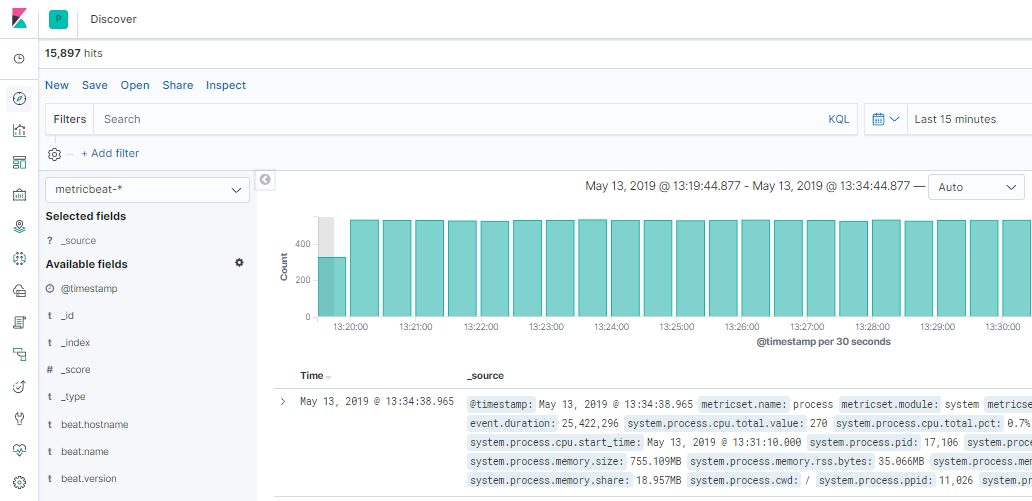
In this document, we are mainly focusing on how to form a query to do log analysis on different production logs. Before that, we need some basic idea on kibana query language.

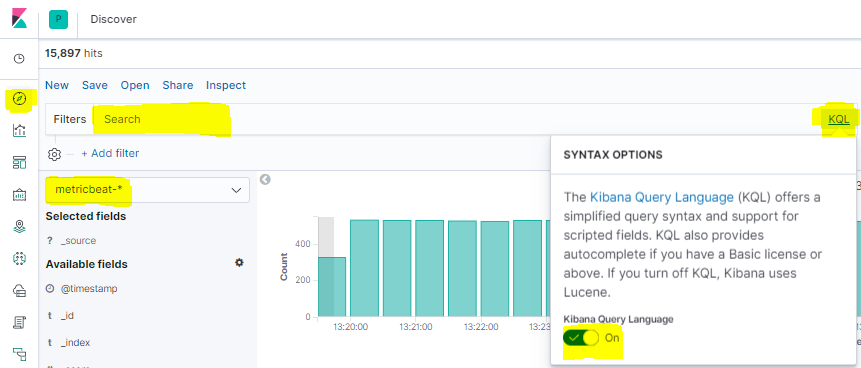
KQL will follow the basic syntaxes to form a query. Thon this query will be converted into JSON format and request elasticsearch to provide the information during that timing.

**Analyzing the logs:**

This screenshot refers to how normal kibana dashboard appears on the home page.

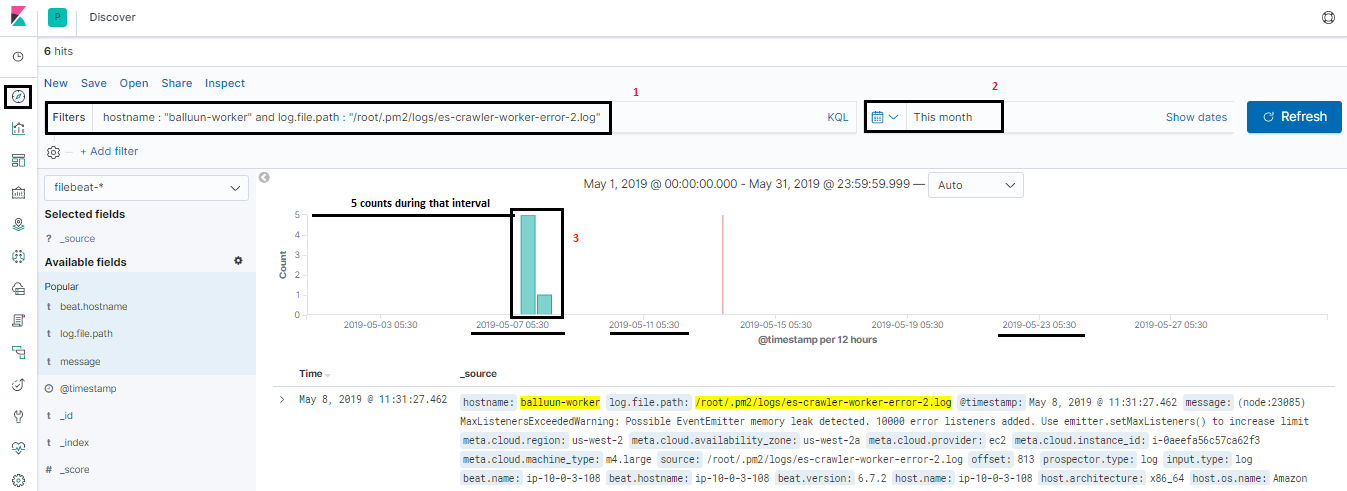
****

Let’s have a deep look into all fields and available options.



1. Discover option used to search the logs
2. We need to select the source. Here we have 2 sources metricbeat and filebeat
   1. **Metricbeat:** This will store all prod server metrics (CPU, Memory, Disk utilization, N/W In&Out)
   2. **Filebeat:** This will store application logs like Web, API, Worker, Rabbit-MQ and Redis server logs.
3. Once you selected the log source, click on KQL to get the suggestions before forming queries. If we enable this feature it will provide syntax on searching any kind of log.
4. **SEARCH BAR:** This is the area we will enter our query.

Below is the example to do some basic analysis on worker server logs.



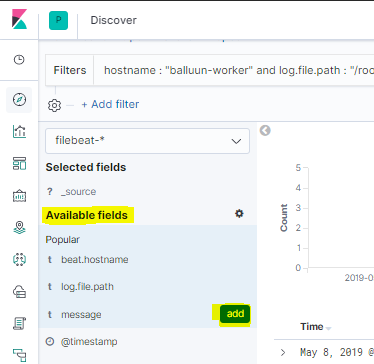
1. Form a query
2. Select time range

When clicking on search it will get the logs by making a query to elasticsearch engine with that time period.

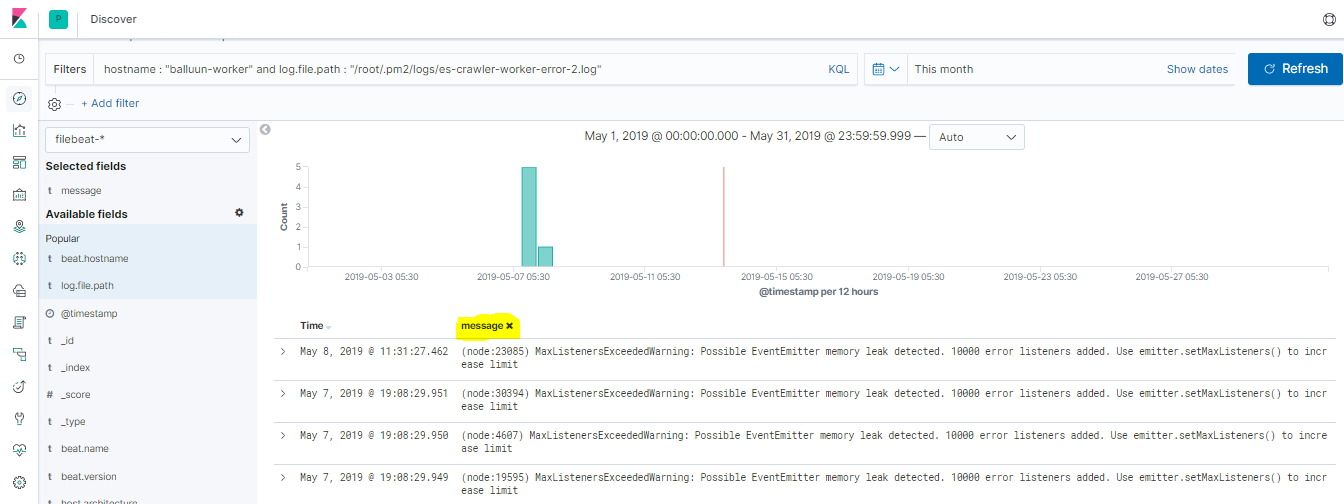
1. Bars indicate no. of times that query matched and count. Counts, we can see the left side and timing we can see the below to bars.

**Adding custom fields to view:**

If we see above dashboards we will be having all unnecessary fields. We may get confused while making some analysis. If you just wanted to see the message you customize the view.



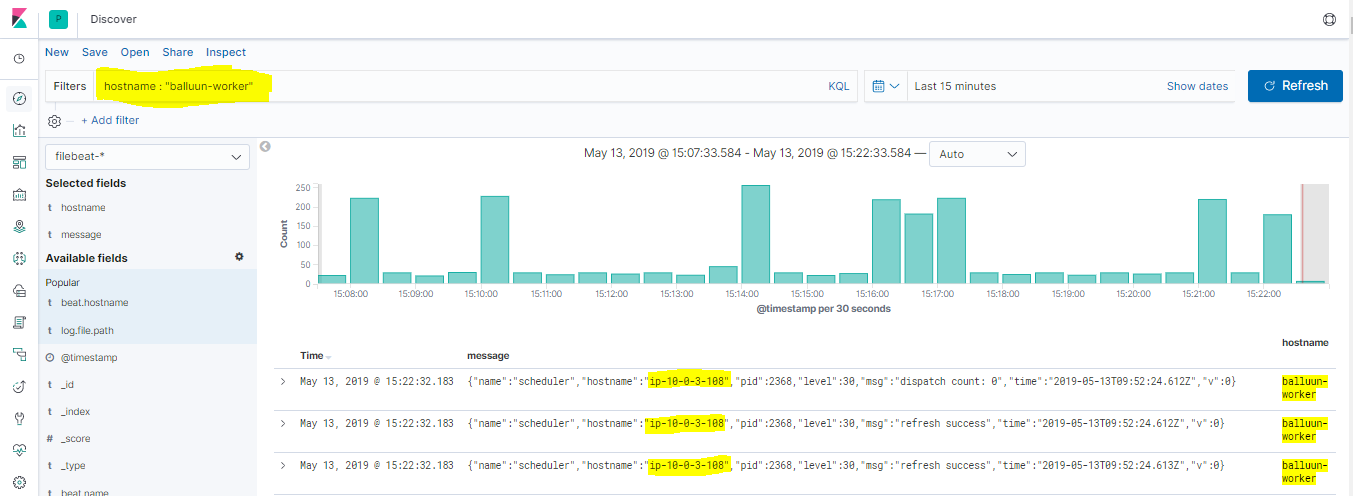
In the available fields, search your custom field and when you place the mouse over there it will show add button and just click on it. After clicking on it Kibana view will be changed to like below.



**Custom fields from filebeats:**

As we are dumping each server log files into elasticsearch we need a way to filter the server level. So we have added one custom field in each server and we can customize like this.

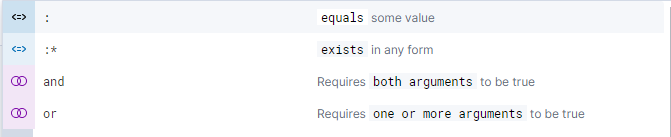
Query: hostname: “balluun-worker” / hostname: “balluun-mq”



It will apply the filter it will get the data for those specific fields only.

**Kibana filters:**

We have the basic filters in kibana to form the query and get the required data.



When you want to match field value exactly we will “:” or if it is similar “:\*”

Example:

* hostname : “balluun-worker”
* hostname :\* “balluun-worker”
* hostname : “balluun-worker” and log.file.path : “/var/log/rabbitmq/\*.log”
* hostname and log.file.path : "balluun-worker"

Note: AND can be used only when both perfectly match. OR can be used when any one of its matches.

It’s better to enable KQL to get the suggestions.

**Analyze Web server logs:**

For any server/service logs first, we need to need to enter the hostname then give the log file path. If you enable KQL it will give you log file location also.

* Open Kibana Discover section
* Select filebeat
* Enter the below query in the search bar to see Web logs.
  + hostname : "balluun-ui" and log.file.path : "/var/log/httpd/access\_log"
  + hostname : "balluun-ui" and log.file.path : "/var/log/httpd/error\_log"

**Analyze API logs:**

* Open Kibana Discover section
* Select filebeat
* Enter the below query in the search bar to see API logs by Nginx module.
  + hostname : “balluun-api” and log.file.path : “/var/log/nginx/access.log”
* To see error logs by API
  + hostname : “balluun-api” and log.file.path : “/var/log/nginx/error.log”

**API Server CRON logs:**

As we are running cron jobs in the API server we have also enabled cron logs to dump into elasticsearch. So that we can have a track of what are the scripts executed by the cron

* Open Kibana Discover section
* Select filebeat
* Enter the below query in the search bar to see API Cron logs .
  + hostname : "balluun-api" and log.file.path : "/var/log/cron"

**Analyze worker server logs:**

* Open Kibana Discover section
* Select filebeat
* All worker server logs stored under the /root/.pm2/logs/<\*>.log
  + /root/.pm2/logs/statistic-worker-out-14.log
  + /root/.pm2/logs/statistic-worker-error-14.log
* Enter the below pattern to see specific worker logs
  + hostname : “balluun-worker” and log.file.path : /root/.pm2/logs/statistic-worker-out-14.log
  + hostname : “balluun-worker” and log.file.path : /root/.pm2/logs/statistic-worker-error-14.log

**Analyze Rabbit-MQ server logs:**

* Open Kibana Discover section
* Select filebeat
* All the rabbit mq server logs stored under the /var/log/rabbitmq/
* Majorly it has 2 log files with server IP.
* Enter the below pattern to see specific worker logs
  + hostname : “balluun-mq” and log.file.path : /var/log/rabbitmq/rabbit@ip-10-0-1-164.log
  + hostname : “balluun-mq” and log.file.path : /var/log/rabbitmq/rabbit@ip-10-0-1-164-sasl.log
* Similar to this we have redis logs
  + hostname : "balluun-mq" and log.file.path : "/var/log/redis/redis.log"

**Getting RabbitMQ Crash Status:**

Basically RabbitMQ storing the crash report in a separate log file located under /var/log/rabbitmq

We have 2 files here.

* var/log/rabbitmq/rabbit@ip-10-0-1-164.log
* var/log/rabbitmq/rabbit@ip-10-0-1-164-sasl.log

The first log file will store **Info, warning** and **error** reports

The second log file will store **error** and **supervisor** reports.

Update the query:

hostname : "balluun-mq" and (log.file.path : "/var/log/rabbitmq/rabbit@ip-10-0-1-164.log" or "/var/log/rabbitmq/rabbit@ip-10-0-1-164-sasl.log")

This query will provide the logs in the combination of Info, warning, error, error and supervisor logs.

Updated kibana visualization with the above query to match paatern and display in the pattern whenever RabbitMQ crashed.

