

The ELK Stack

Elastic Logging



Content

- | | |
|---------------------------------|-------|
| 1. Log analysis | |
| 2. The ELK stack | |
| 3. Elasticsearch | Lab 1 |
| 4. Kibana phase 1 | Lab 2 |
| 5. Beats | Lab 3 |
| 6. Kibana | Lab 4 |
| 7. Logstash & Filebeat | Lab 5 |
| 8. Enhanced Logstash | Lab 6 |
| 9. Kibana Custom Visualisations | Lab 7 |



Log analysis

What does it mean to analyse logs?



Log analysis

- Taken from Wikipedia
 - https://en.wikipedia.org/wiki/Log_analysis

"In computer log management and intelligence, log analysis (or system and network log analysis) is an art and science seeking to make sense out of computer-generated records (also called log or audit trail records). The process of creating such records is called data logging."



Reasons

- Compliance with security policies
- Compliance with audit or regulation
- System troubleshooting
- Forensics (during investigations or in response to subpoena)
- Security incident response
- Understanding online user behavior
- Debugging production application issues



Many formats, many locations

- Logs come in many shapes and sizes
 - System logs
 - Application logs
 - Audit logs
 - Data feeds
 - Other sources
- Formats
 - Single line
 - Multiline
 - Mixed data
 - Organised



Disparate logging

- Logs are everywhere
- Why are they not centralised?
- One view to rule them all!
- Ability to obfuscate the data
 - Enable Devs to view production logs for live debugging
 - Enables support to perform more meaningful tasks



The ELK Stack

Centralise your view of logs and get what you need

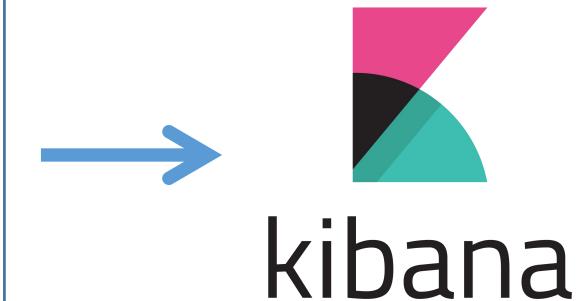
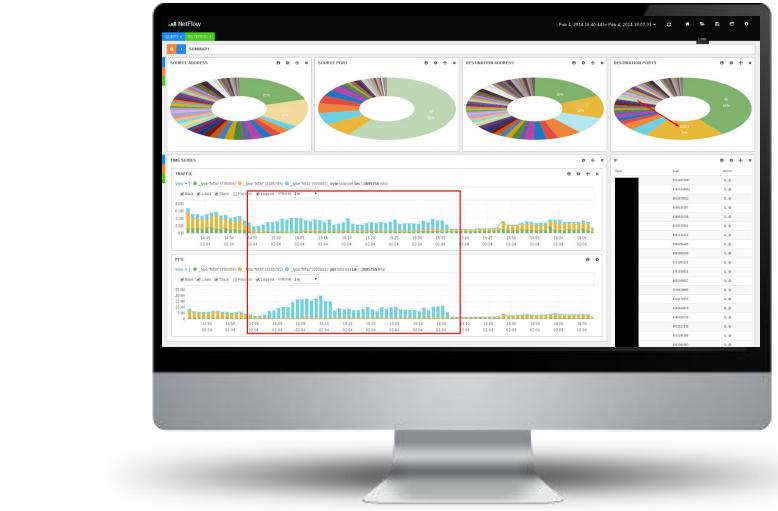
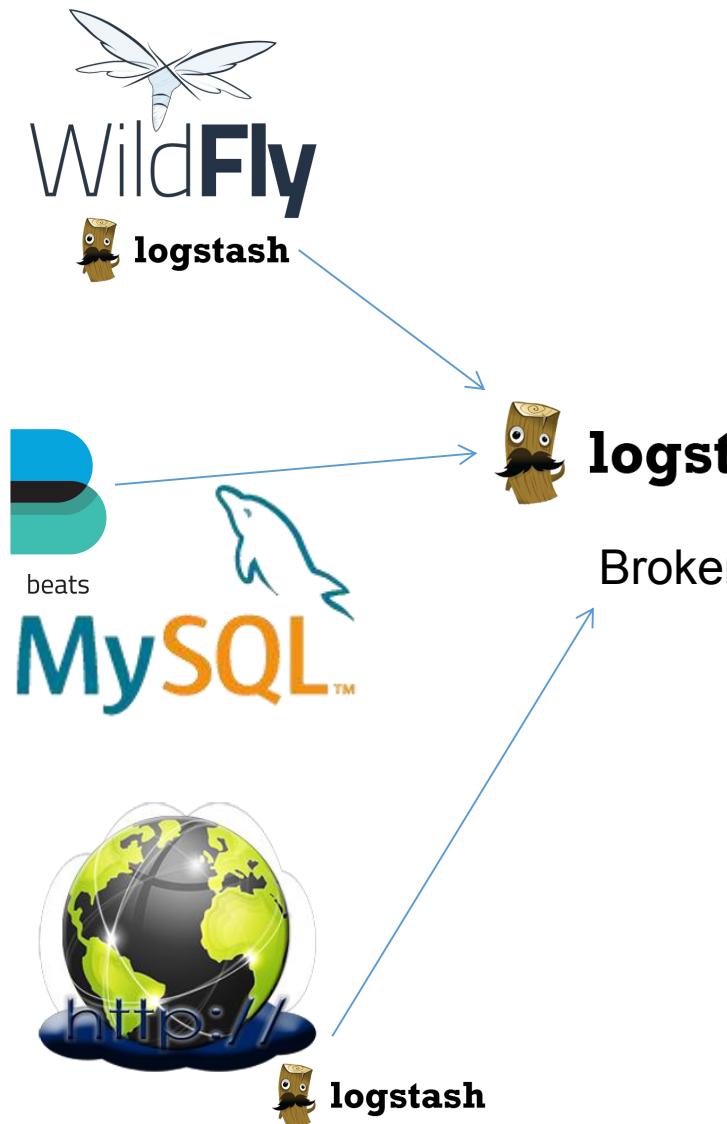


What is ELK?

- Open source log and data viewing system
- Used for data analytics
 - So not just log files but any data you like
 - Can take inputs from many source
- Designed for big data storage and searching
 - Yes it is a noSQL database
- Further detail at
 - <https://logz.io/learn/complete-guide-elk-stack/>
 - <https://qbox.io/blog/welcome-to-the-elk-stack-elasticsearch-logstash-kibana>



The ELK stack



The heart of the system

- Elasticsearch
 - Big data store
 - Clusters for very large data and fast searching
- JSON API
- Adding data to Elasticsearch is called "Indexing"
 - POST or PUT methods
- Data can be searched using the GET method
- Data removal using the DELETE method



Indexing using the API

```
elk:~> curl -X PUT http://localhost:9200/app/instructors/2 \
-d '{
    "id":2,
    "firstname":"Steve",
    "lastname":"Shilling",
    "speciality":"Unix",
    "change_date":"2017-07-22 16:22:00"
}

{
    "_index":"app",
    "_type":"instructors",
    "_id":"2",
    "_version":1,
    "result":"created",
    "_shards":{
        "total":2,
        "successful":1,
        "failed":0
    },
    "created":true
}'
```



Querying Elasticsearch

```
elk:~> curl -X GET http://localhost:9200/instructors/2

{
    "_index":"app",
    "_type":"instructors",
    "_id":"2",
    "_version":2,
    "found":true,
    "_source":{
        "id":2,
        "firstname":"Steve",
        "lastname":"Shilling",
        "speciality":"Unix",
        "change_date":"2017-07-22 16:22:00"
    }
}
```



Using _search API

- curl -X GET http://localhost:9200/_search?q=steve
 - curl -X GET http://localhost:9200/_search?q=firstname:steve
 - curl -X GET http://localhost:9200/_search?q=change_date:2017*
 - curl -X GET http://localhost:9200/_search?q=speciality:Uni?
-
- took: search time in milliseconds
 - timed_out: If the search timed out
 - shards: Lucene shards searched and success/fail rate
 - hits: Actual results found and metadata

```
elk:~> curl -X GET http://localhost:9200/_search?q=steve
```

```
{  
  "took":18,  
  "timed_out":false,  
  "_shards":{  
    "total":11,  
    "successful":11,  
    "failed":0  
  },  
  "hits":{  
    "total":1,  
    "max_score":0.28488502,  
    "hits":[{  
      "_index":"app",  
      "_type":"instructors",  
      "_id":"2",  
      "_score":0.28488502,  
      "_source":{  
        "id":2,  
        "firstname":"Steve",  
        "lastname":"Shilling",  
        "speciality":"Unix",  
        "change_date":"2017-07-22 16:22:00"  
      }  
    }]  
  }  
}
```



`_search` API with JSON

```
curl -X GET http://localhost:9200/_search \  
-d '  
{  
  "query":{  
    "match_phrase":{  
      "speciality":"Unix"  
    }  
  }  
}'
```



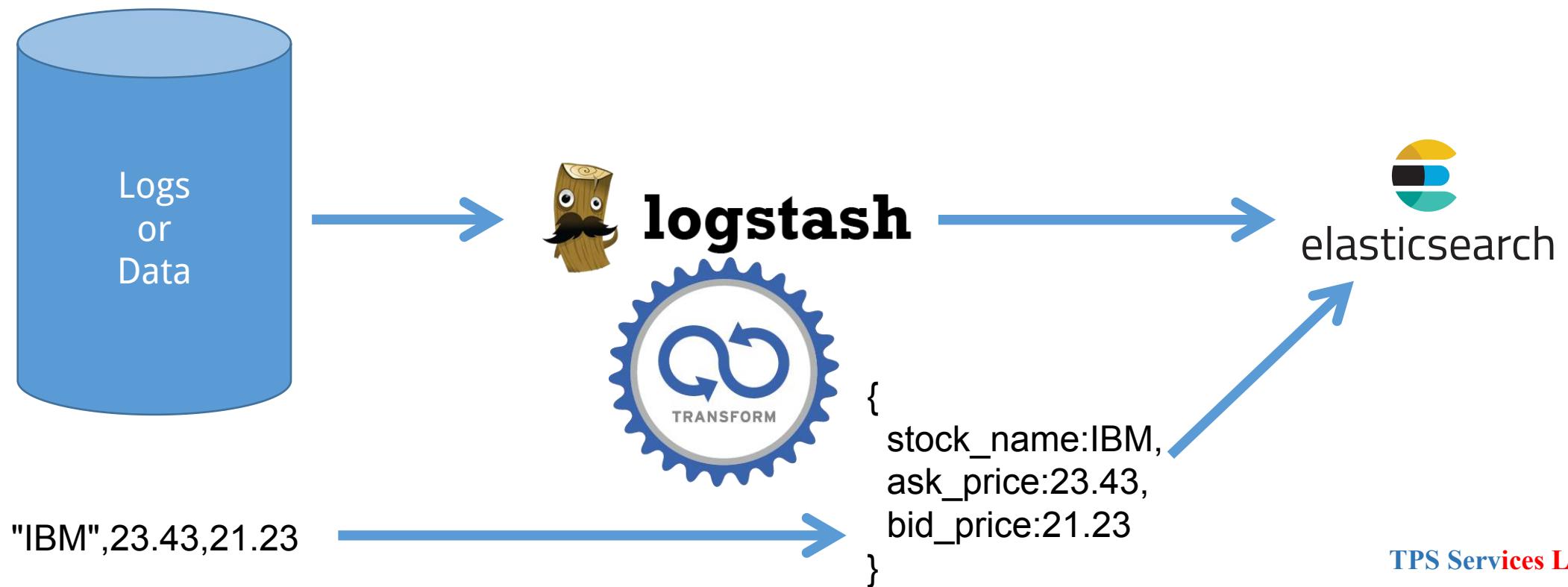
Data shipping

- Elasticsearch needs to be fed data
- Any JSON output can be collected by Elasticsearch
- Elastic make 2 products to help with logs and other data
 - Logstash
 - Beats



Logstash

- Server side data processing pipeline
 - <https://www.elastic.co/products/logstash>



Beats

- Lightweight processes for gathering common system data
 - Known applications and data formats
 - Log shipper for large enterprises
 - Sends to logstash server
 - Reduce load on Logstash servers and Elasticsearch cluster
- Extensible framework
 - Write your own
 - Use another from the community
 - <https://www.elastic.co/guide/en/beats/libbeat/current/community-beats.html>
- From Elastic
 - Filebeat, Metricbeat, Packetbeat, Winlogbeat, Heartbeat



Kibana

- The front end to the stack
- A web application for viewing, querying and analysis
- Capabilities
 - View raw data
 - Search raw data
 - Create charts for analysis



On with the show

- Now you know what ELK does
- Let's build and use it



Elastic Stack: Logging Lab 1

Installing, configuring and starting Elasticsearch



Installing Elasticsearch: Steps

- Install Java
- Download Elasticsearch
- Install Elasticsearch
- Start Elasticsearch
- Test
- Configuring Elasticsearch

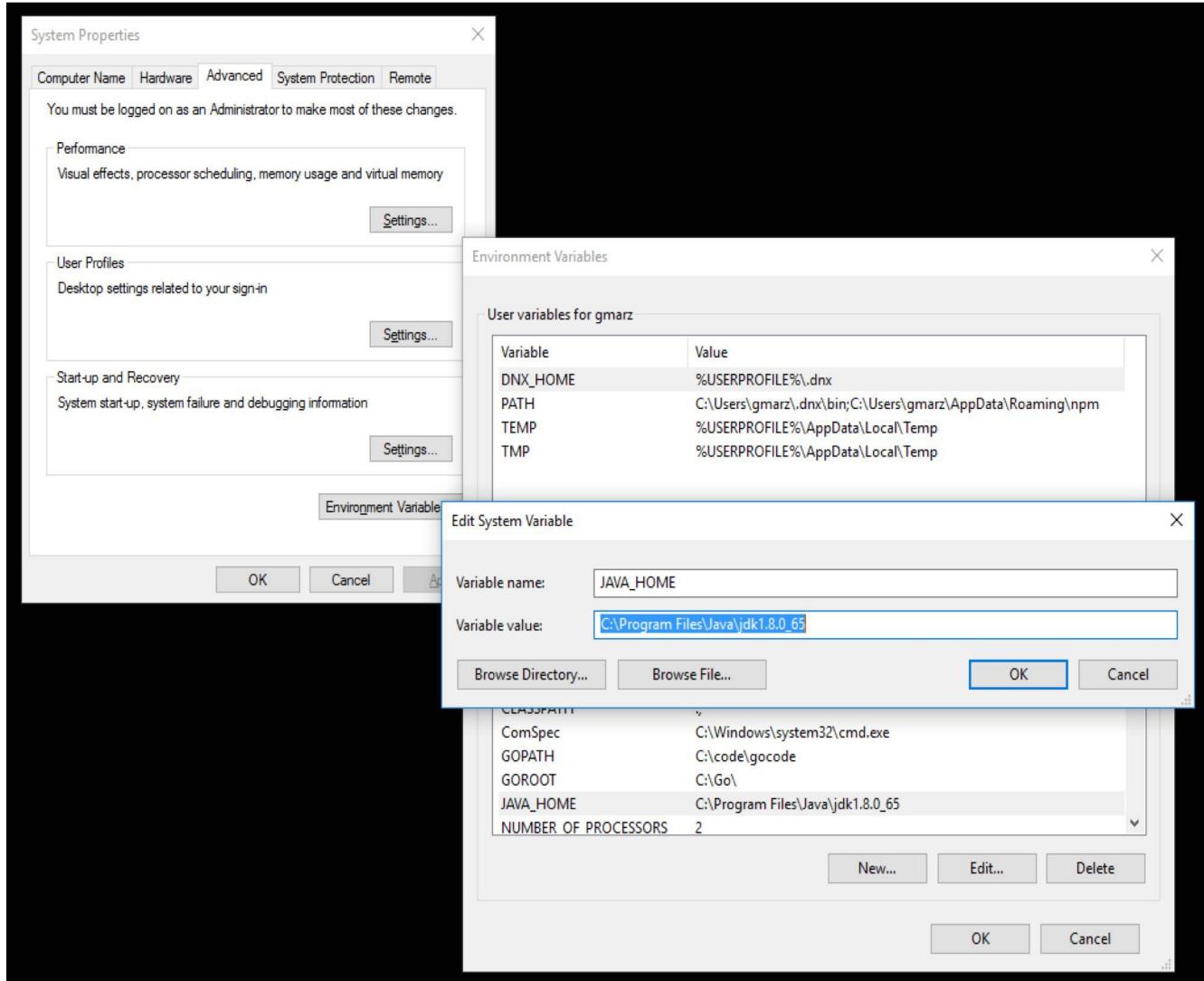


Installing Java

- Elasticsearch is written in this language
 - <http://www.oracle.com/technetwork/java/javase/downloads/index.html>
- Linux can use the native Java packages (must be 1.8 or higher)
 - yum -y install java-1.8.0-openjdk
- Ensure JAVA_HOME variable is set
 - Most installations will do this
 - Although you may need to log out and back in for Windows
 - If it does not then see the following slide
 - On Linux if it does not set, or the wrong version;
 - /usr/bin/alternatives --config java



Set the Windows JAVA_HOME variable



Make sure you set the path that your JDK is at



Create A Workspace

- You will want to create a place to work out for this course.

```
# Mac or Linux:
```

```
elk:~> mkdir elkCourse  
elk:~> cd elkCourse  
elk:~/elkCourse>
```

```
# Windows
```

```
C:\Users\elk > mkdir elkCourse  
C:\Users\elk > cd elkCourse  
C:\Users\elk\elkCourse >
```



Download Elasticsearch

- For the latest version go to the elastic web site
 - www.elastic.co/downloads/elasticsearch
- Download the tar or zip files for better control
 - Location, and start up for different operating systems
- If you have administrator rights you can download the package
 - rpm = RedHat
 - deb = Debian/Ubuntu
 - msi = Microsoft



Installing Elasticsearch

- Package installations will do everything
 - Require Administrator rights to install
 - Will install as a service
 - RHEL = yum -y localinstall elasticsearch-5.5.0-x86_64.rpm
 - DEBIAN = dpkg -i elasticsearch-5.5.0-amd64.deb
- Using tar or zip will allow you to run it as a normal user
 - **Warning: It will fill up your disk quota!**
- Windows use a suitable unzip tool
- Linux use the tar command
 - tar xvf elasticsearch-5.5.0.tar.gz



Installing Elasticsearch

- For this course extract the file inside your elkCourse directory
 - e.g.

```
# Mac or Linux:
```

```
elk:~/elkCourse> tar xvf elasticsearch-5.5.0.tar.gz
elk:~/elkCourse> ls
elasticsearch-5.5.0
```



Starting Elasticsearch

- To start Elasticsearch as a user (and for this course)

```
# Windows:
```

```
C:\Users\elk\elkCourse> dir  
elasticsearch-5.5.0
```

```
C:\Users\elk\elkCourse> cd elasticsearch-5.5.0
```

```
C:\Users\elk\elkCourse\elasticsearch-5.5.0> bin\elasticsearch.bat
```

```
# to quit Elasticsearch type ctrl-c, but don't quit now!
```

```
# Mac or Linux:
```

```
elk:~/elkCourse> ls  
elasticsearch-5.5.0
```

```
elk:~/elkCourse> cd elasticsearch-5.5.0
```

```
elk:~/elkCourse/elasticsearch-5.5.0> bin/elasticsearch
```

```
# to quit Elasticsearch type ctrl-c, but don't quit now!
```

```
# Linux as a service:
```

```
elk:~/elkCourse> sudo service elasticsearch start
```



Testing Elasticsearch

- Check Elasticsearch is running
 - Point your web browser or curl `http://localhost:9200`

```
← → C ⌂ ⓘ localhost:9200
{
  "name" : "logstash",
  "cluster_name" : "elasticsearch",
  "version" : {
    "number" : "2.0.0",
    "build_hash" : "de54438d6af8f9340d50c5c786151783ce7d6be5",
    "build_timestamp" : "2015-10-22T08:09:48Z",
    "build_snapshot" : false,
    "lucene_version" : "5.2.1"
  },
  "tagline" : "You Know, for Search"
}
```

```
# Mac or Linux:
elk:~/elkCourse/elasticsearch-5.5.0> curl http://localhost:9200
{
  "name" : "logstash",
  "cluster_name" : "elasticsearch",
  "version" : {
    "number" : "2.0.0",
    "build_hash" : "de54438d6af8f9340d50c5c786151783ce7d6be5",
    "build_timestamp" : "2015-10-22T08:09:48Z",
    "build_snapshot" : false,
    "lucene_version" : "5.2.1"
  },
  "tagline" : "You Know, for Search"
}
```



Configuring Elasticsearch

- Configured through YAML files
 - `elasticsearch.yml`
 - Define cluster names and nodes
 - Define log location
 - Specify which port to run on if you cannot use 9200
 - `http.port: 9200`
 - Define which NIC to run on
 - `localhost` is default
 - `0.0.0.0` for any network
 - `network.host: 0.0.0.0`
 - For version up to 6
 - `network.host: _local_,_enp0s8:ipv4_`
 - Also requires `/etc/security/limits.conf` to be changed to increase file descriptors
 - `* hard nofile 65536`



Checking changes

- Use netstat to ensure process is on required IPs and Ports

Active Internet connections (only servers)					
Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	0.0.0.0:22	0.0.0.0:*	LISTEN
tcp	0	0	127.0.0.1:25	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:55804	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:5601	0.0.0.0:*	LISTEN
tcp6	0	0	:::9200	...:*	LISTEN
tcp6	0	0	:::9300	...:*	LISTEN
tcp6	0	0	:::22	...:*	LISTEN
tcp6	0	0	:::125	...:*	LISTEN

elk:~> netstat -tln

Elasticsearch running
All possible NICs
Ports 9200 and 9300



Congratulation

- You have completed Lab 1
- You have configured Elasticsearch and tested it is ready
- Elasticsearch is now available to receive data





Installing Kibana Lab 2

Installing, configuring and starting Kibana

Installing Kibana: Steps

- Downloading Kibana
- Install Kibana
 - Optionally install "Sense" if using Kibana 4.x
- Start Kibana
- Restart Kibana
- Test
- Configuring Kibana



kibana



Downloading Kibana

- Download Kibana.
 - You may choose to do this course using Kibana 5 or 4.x.
 - If you are using Elasticsearch 2.x then you must use Kibana 4.x.
 - We recommend Kibana 5
- www.elastic.co/downloads/kibana
- Like with Elasticsearch you should choose your download type
 - Administrators use the O/S packages
 - Users/Developers use the WINDOWS or LINUX downloads
 - ZIP or TAR.GZ files



Installing kibana

- Package installations will do everything
 - Require Administrator rights to install
 - Will install as a service
 - RHEL = yum -y localinstall kibana-5.5.0-x86_64.rpm
 - DEBIAN = dpkg -i kibana-5.5.0-amd64.deb
- Using tar or zip will allow you to run it as a normal user
- Windows use a suitable unzip tool
- Linux use the tar command
 - tar xvf kibana-5.5.0-linux-x86_64.tar.gz



Installing Kibana

- For this course extract the file inside your elkCourse directory
 - e.g.

```
# Mac or Linux:
```

```
elk:~/elkCourse> tar xvf kibana-5.5.0-linux-x86_64.tar.gz
elk:~/elkCourse> ls
kibana-5.5.0-linux-x86_64
```



Starting Kibana

- To start Elasticsearch as a user (and for this course)

```
# Windows:
```

```
C:\Users\elk\elkCourse> dir  
elasticsearch-5.5.0  kibana-5.5.0  
C:\Users\elk\elkCourse> cd kibana-5.5.0  
C:\Users\elk\elkCourse\kibana-5.5.0> bin\kibana.bat
```

```
# to quit Elasticsearch type ctrl-c, but don't quit now!
```

```
# Mac or Linux:
```

```
elk:~/elkCourse> ls  
elasticsearch-5.5.0  kibana-5.5.0-linux-x86_64  
elk:~/elkCourse> cd kibana-5.5.0-linux-x86_64  
elk:~/elkCourse/kibana-5.5.0-linux-x86_64> bin/kibana
```

```
# to quit Elasticsearch type ctrl-c, but don't quit now!
```

```
# Linux as a service:
```

```
elk:~/elkCourse> sudo service kibana start
```



Kibana view

- Point your browser at your Kibana hosts port 5601
- <http://localhost:5601>



Version 4 configuration error

Server Status

Status: Red 

Installed Plugins

Name	Status
plugin:kibana	 Ready
plugin:elasticsearch	 [illegal_argument_exception] [field_sort] unknown field [ignore_unmapped], parser not found
plugin:kbn_vislib_vis_types	 Ready
plugin:markdown_vis	 Ready
plugin:metric_vis	 Ready
plugin:spyModes	 Ready
plugin:statusPage	 Ready
plugin:table_vis	 Ready



Version 4 working

The screenshot shows the Kibana 4 interface. At the top, there is a navigation bar with the Kibana logo and links for Discover, Visualize, Dashboard, and Settings. Below the navigation bar, there is a secondary navigation bar with links for Indices, Advanced, Objects, Status, and About. The main content area is titled "Index Patterns". A warning message states: "Warning No default index pattern. You must select or create one to continue." The main section is titled "Configure an index pattern" and contains instructions: "In order to use Kibana you must configure at least one index pattern. Index patterns are used to identify the Elasticsearch index to run search and analytics against. They are also used to configure fields." There are two checkboxes: "Index contains time-based events" (checked) and "Use event times to create index names" (unchecked). A section for "Index name or pattern" shows the input field "logstash-*". A "Time-field name" dropdown is set to "@timestamp". At the bottom is a green "Create" button.



Version 5 configuration error

kibana

- Discover
- Visualize
- Dashboard
- Timelion
- Dev Tools
- Management

Collapse

Status: **Yellow**

elk

Heap Total (MB)

101.71

Heap Used (MB)

70.17

Load

0.46, 0.14, 0.08

Response Time Avg (ms)

0.00

Response Time Max (ms)

0.00

Requests Per Second

0.00

Status Breakdown

ID	Status
ui settings	⚠ Elasticsearch plugin is yellow
plugin:kibana@5.5.0	✓ Ready
plugin:elasticsearch@5.5.0	⚠ Waiting for Elasticsearch
plugin:console@5.5.0	✓ Ready
plugin:metrics@5.5.0	✓ Ready
plugin:timelion@5.5.0	✓ Ready



Version 5 working

Management / Kibana

Index Patterns Saved Objects Advanced Settings

Discover

Visualize

Dashboard

Timelion

Dev Tools

Management

Collapse

Warning
No default index pattern. You must select or create one to continue.

Configure an index pattern

In order to use Kibana you must configure at least one index pattern. Index patterns are used to identify the Elasticsearch index to run search and analytics against. They are also used to configure fields.

Index name or pattern

logstash-*

Patterns allow you to define dynamic index names using * as a wildcard. Example: logstash-*

Time Filter field name refresh fields

@timestamp

Expand index pattern when searching [DEPRECATED]

With this option selected, searches against any time-based index pattern that contains a wildcard will automatically be expanded to query only the indices that contain data within the currently selected time range.

Searching against the index pattern `logstash-*` will actually query Elasticsearch for the specific matching indices (e.g. `logstash-2015.12.21`) that fall within the current time range.

With recent changes to Elasticsearch, this option should no longer be necessary and will likely be removed in future versions of Kibana.

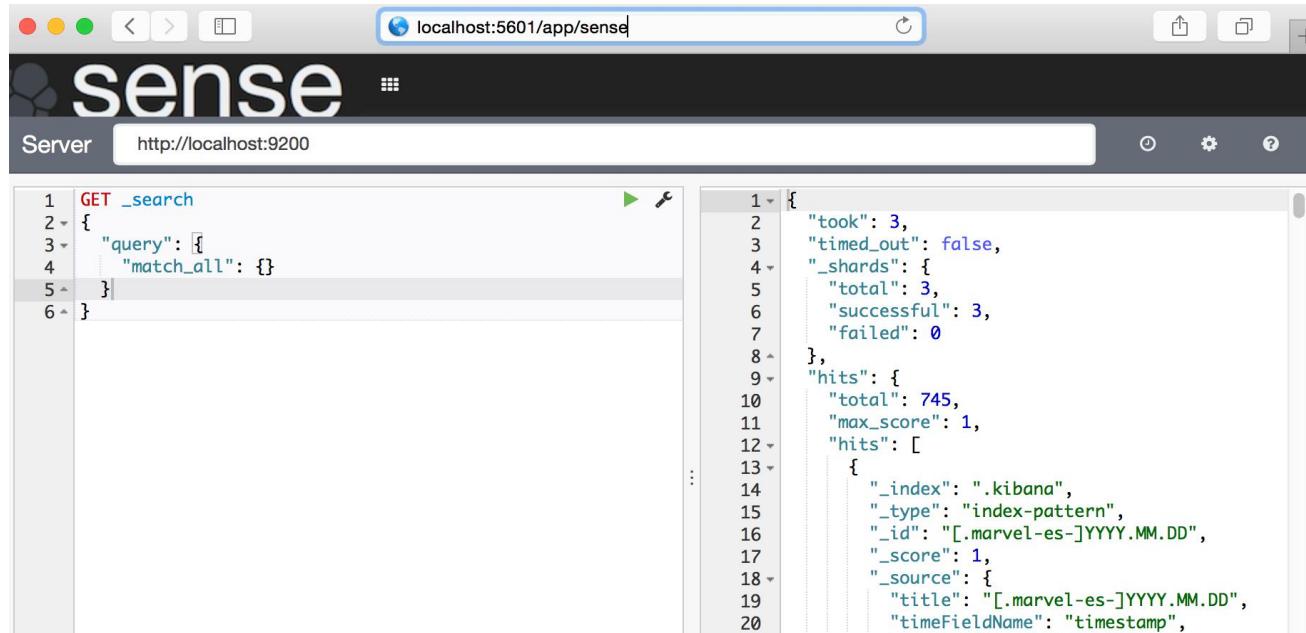
Use event times to create index names [DEPRECATED]

Create



Kibana AppSense

- App Sense allows you to view Elasticsearch API through Kibana
 - Make direct API JSON queries to Elasticsearch
 - It is a plugin so needs to be installed through Kibana
- Accessed by pointing web browser at;
 - <http://localhost:5601/app/sense>



The screenshot shows the Kibana AppSense interface in a web browser. The address bar displays `localhost:5601/app/sense`. The main window has a dark header with the word "sense" in white. Below the header, there's a "Server" dropdown set to `http://localhost:9200`. On the left, a code editor pane shows a JSON search query:

```
1 GET _search
2 {
3   "query": {
4     "match_all": {}
5   }
6 }
```

On the right, the results of the search are displayed as a JSON object:

```
1 {
2   "took": 3,
3   "timed_out": false,
4   "_shards": {
5     "total": 3,
6     "successful": 3,
7     "failed": 0
8   },
9   "hits": {
10    "total": 745,
11    "max_score": 1,
12    "hits": [
13      {
14        "_index": ".kibana",
15        "_type": "index-pattern",
16        "_id": "[.marvel-es-]YYYY.MM.DD",
17        "_score": 1,
18        "_source": {
19          "title": "[.marvel-es-]YYYY.MM.DD",
20          "timeFieldName": "timestamp",
21        }
22      }
23    ]
24  }
25 }
```



Installing AppSense v4 Kibana

- Must be done from the command line
- Requires internet access to download plugin
- Use the command used to start Kibana with options

```
# Mac/Linux:
```

```
elk:~/elkCourse/kibana-4.4.0-linux-x86_64> bin/kibana plugin --install elastic/sense
```

```
# Windows:
```

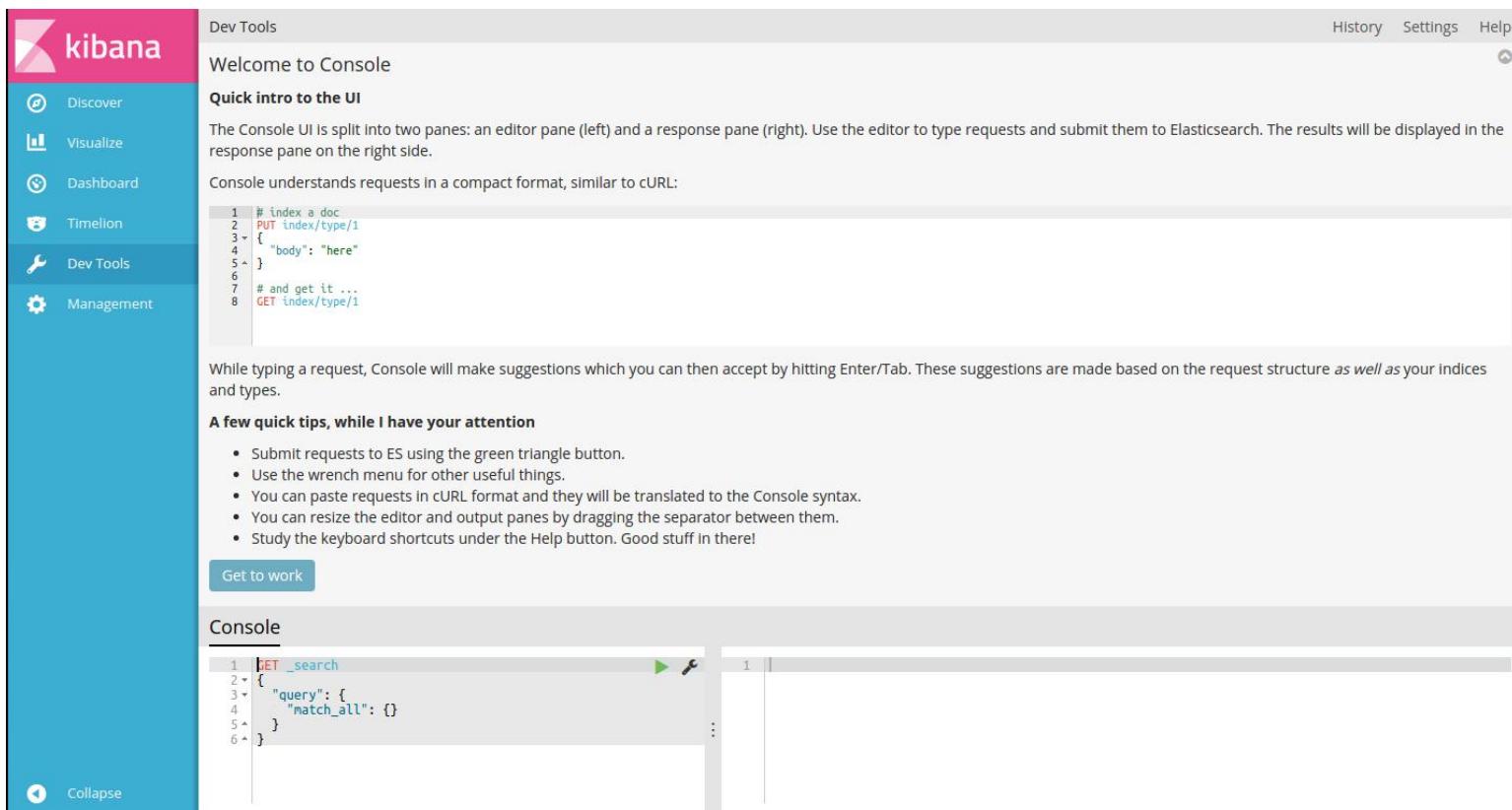
```
C:\Users\elkCourse\kibana-4.4.0> bin\kibana.bat plugin --install elastic/sense
```

- Restart Kibana

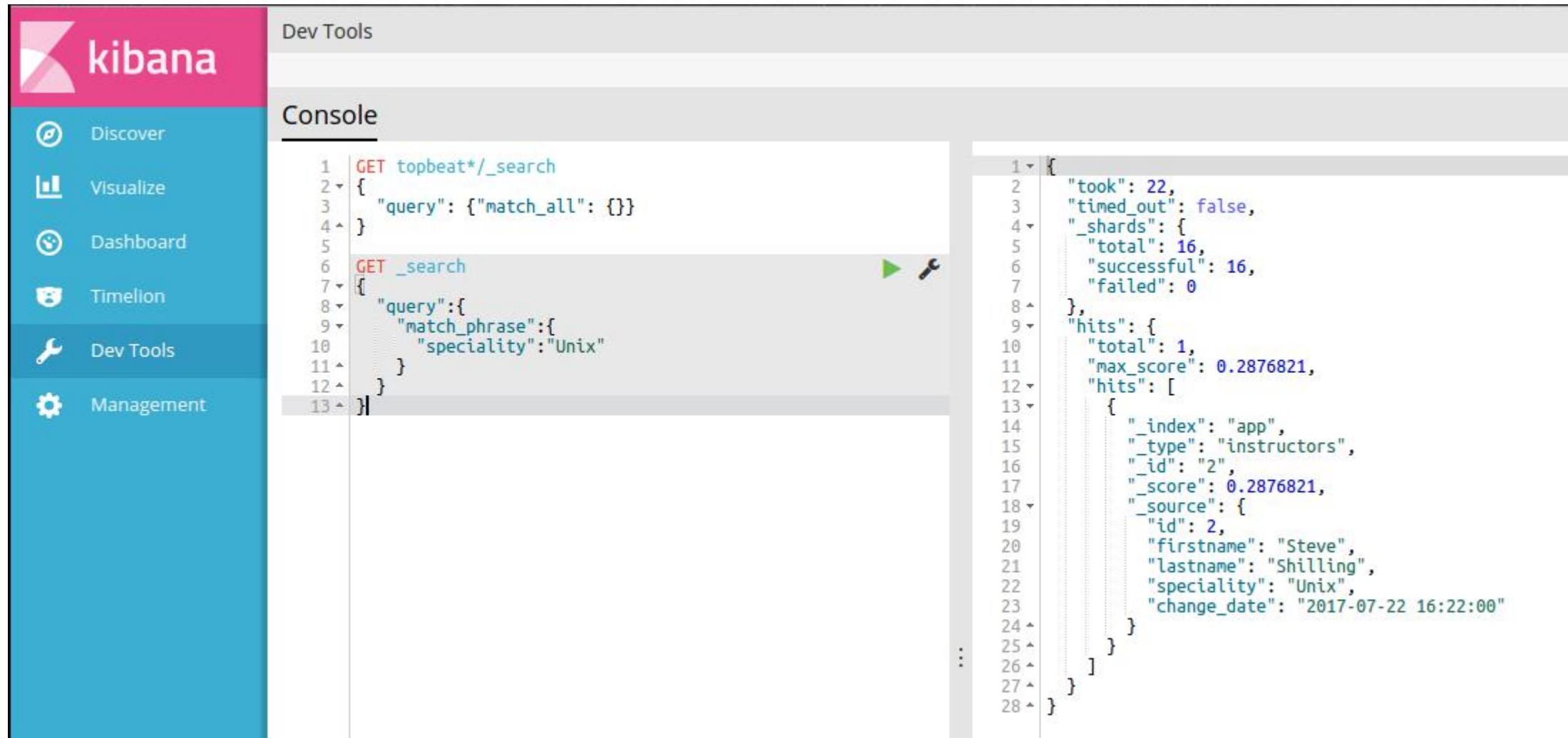


Kibana 5 AppSense equivalent

- Version 5 has an Elasticsearch console built in
- Use the **Dev Tools** link in the left menu



Checking Kibana against Elasticsearch



The screenshot shows the Kibana interface with the 'Dev Tools' tab selected in the sidebar. The 'Console' section contains the following Elasticsearch search query:

```
1 GET topbeat*/_search
2 {
3   "query": {"match_all": {}}
4 }
5
6 GET _search
7 {
8   "query":{
9     "match_phrase":{
10       "speciality": "Unix"
11     }
12   }
13 }
```

The results panel displays the response from Elasticsearch, which includes the execution time, shard information, total hits, and a single hit object containing the document's index, type, ID, score, source, and specific fields like first name, last name, speciality, and change date.

```
1 {
2   "took": 22,
3   "timed_out": false,
4   "_shards": {
5     "total": 16,
6     "successful": 16,
7     "failed": 0
8   },
9   "hits": {
10     "total": 1,
11     "max_score": 0.2876821,
12     "hits": [
13       {
14         "_index": "app",
15         "_type": "instructors",
16         "_id": "2",
17         "_score": 0.2876821,
18         "_source": {
19           "id": 2,
20           "firstname": "Steve",
21           "lastname": "Shilling",
22           "speciality": "Unix",
23           "change_date": "2017-07-22 16:22:00"
24         }
25       }
26     ]
27   }
28 }
```



Congratulation

- You have completed Lab 2
- Kibana will allow us to view data when it comes in



Beats Lab 3

Install, configure and start beats



Working with Beats

- Where to download
- Install Topbeat
- Configure Topbeat to send to Elasticsearch
- Start Topbeat
- Test
- View in Kibana



Downloading Beats

- Elastic created Beats
 - <https://www.elastic.co/downloads/beats>
- From other developers
 - <https://www.elastic.co/guide/en/beats/libbeat/current/community-beats.html>
 - Mostly Git repos
- Beats are written in GO
 - <https://www.elastic.co/guide/en/beats/libbeat/current/new-beat.html>



Install Topbeat

- Download from
 - www.elastic.co/downloads/beats/topbeat
- Unzip topbeat and place it in your ELK stack directory
- Before starting Topbeat we will need to configure it
 - Before doing that make a backup copy

```
# Mac/Linux:
```

```
elk:~/elkCourse/topbeat-1.3.1-x86_64> cp topbeat.yml topbeat.yml.orig
```

```
# Windows:
```

```
C:\Users\elkCourse\topbeat-1.3.1> copy topbeat.yml topbeat.yml.bak
```



Configure topbeat

- Open the **topbeat.yml** file in your favourite editor
- Edit the following entries in that file
 - Some of the items may need the # removed, some already set

input:

```
# In seconds, defines how often to read server statistics
period: 10

# Regular expression to match the processes that are monitored
procs: [".*"]

# Statistics to collect (we are collecting all the explained  statistics)
stats:
  system: true
  process: true
  filesystem: true

shipper:
# Tags make it easy  properties.
tags: ["front-end","web-tier", "linux", "desktop"]
```



Configure topbeat shipper

- Name of the shipper
 - Default is the hostname
- Tags used to identify information from shippers
 - A way to group information together from servers
 - Or to identify a particular server

```
shipper:  
# Tags make it easy properties.  
tags: ["front-end", "web-tier", "linux", "desktop"]
```



Configure topbeat where to go?

- Beats need to be told where to send their data
- Here we will send it directly to Elasticsearch

```
output: elasticsearch:  
  # add your ES host address  
  hosts: ["localhost:9200"]  
  
  # Optional protocol and basic auth credentials.  
  #protocol: "https"  
  #username: "admin"  
  #password: "s3cr3t"
```



Configure topbeat to use template

- Use the template it ships with
- Defines the JSON to send as output



Starting topbeat

- Start the beat - let the data flow.....

```
# Mac OS, Linux  
elk:~/elkCourse/topbeat-1.3.1-x86_64> ./topbeat -e -c topbeat.yml -d "elasticsearch"
```

```
# or if it must run with administrator privileges  
elk:~/elkCourse/topbeat-1.3.1-x86_64> sudo ./topbeat -e -c topbeat.yml -d "elasticsearch"
```

```
# Windows  
C:\Users\elk\elkCourse\topbeat> .\topbeat.exe -e -c topbeat.yml -d "elasticsearch"
```



Starting as a service

- Windows
 - Beats file comes with PowerShell installation script
 - install-service-topbeat.ps1
 - Also uninstall script too
 - uninstall-service-topbeat.ps1
- Linux
 - Create the relevant SVR4 init.d script or systemd control file



Example Linux /etc/init.d/topbeat script

```
#!/bin/bash
# description: topbeats service
# chkconfig: 35 99 99
case $1 in
    start)
        cd /opt/beats/topbeat*/bin
        nohup ./topbeat -e -c topbeat.yml -d "elasticsearch" >/var/log/topbeat.log 2>/var/log/topbeat.err &
        topbeatpid=$(ps -ef | grep topbeat | grep -v grep | awk '{print $2}')
        echo $topbeatpid >/var/run/topbeat.pid
        ;;
    stop)
        kill $(cat /var/run/topbeat.pid)
        ;;
    status)
        if ! ps -ef | grep topbeat | grep $(cat /var/run/topbeat.pid) | grep -v grep >/dev/null 2>&1
        then
            echo "Topbeat is running"
        else
            echo "Topbeat is not running"
        fi
        ;;
esac
```



Systemd topbeat.service

```
# Filename: /usr/lib/systemd/system/topbeat.service

[Unit]
Description=Topbeat service script
After=syslog.target network.target

[Service]
Type=forking
PIDFile=/var/run/topbeat.pid
ExecStart=/opt/beats/topbeat/bin/topbeat -e -c /opt/beats/topbeat/bin/topbeat.yml -d "elasticsearch"
>/var/log/topbeat.log 2>/var/log/topbeat.err
ExecStop=kill $(cat /var/run/topbeat.pid)

[Install]
WantedBy=multi-user.target
```

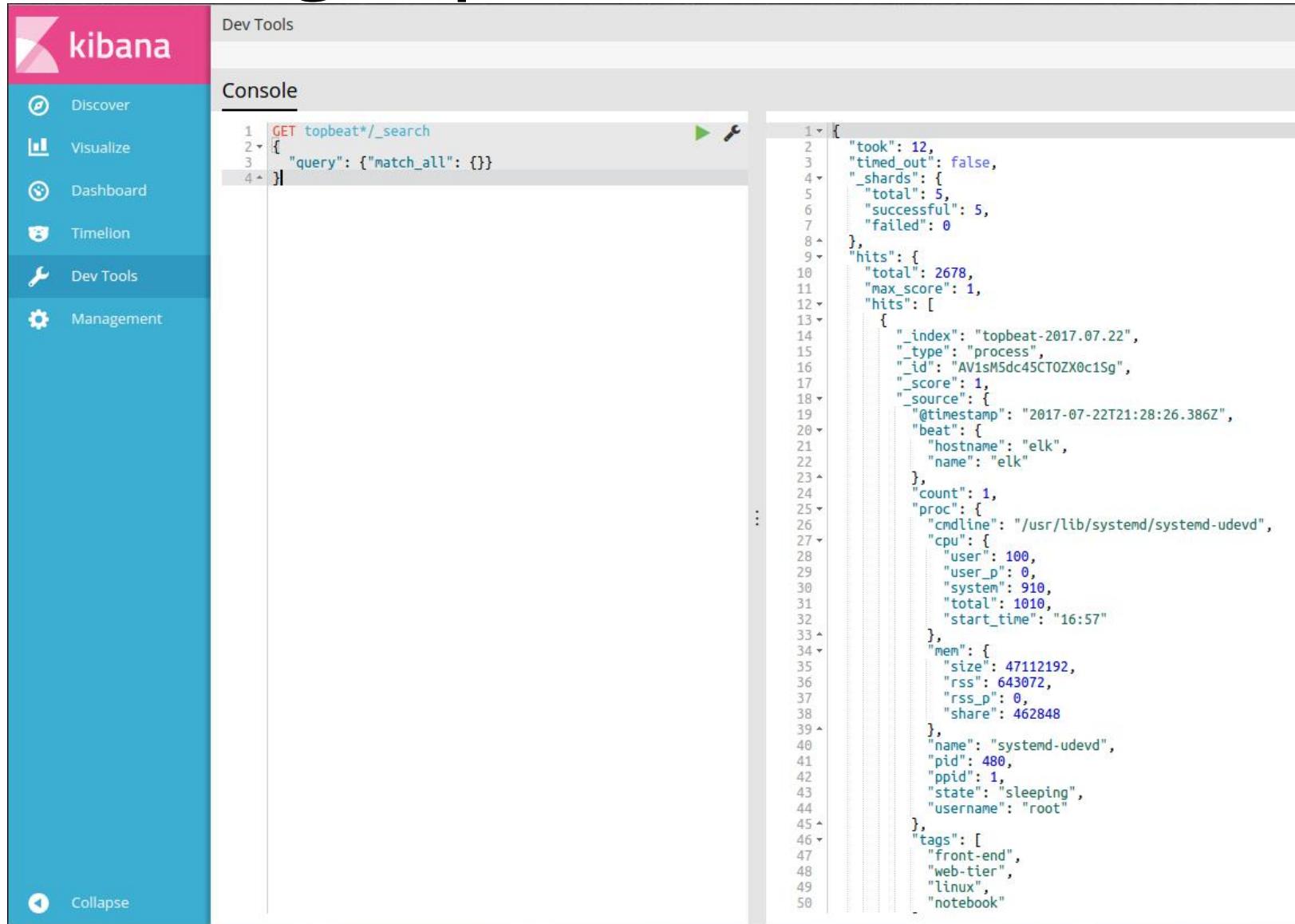


Checking Elasticsearch

```
elk:~> curl -X GET http://localhost:9200/topbeat*/_search -d '{"query": {"match_all": {}}}'  
  
{"took":12,"timed_out":false,"_shards":{"total":5,"successful":5,"failed":0},"hits":{"total":2678,"max_score":1.0,"hits":[{"_index":"topbeat-2017.07.22","_type":"process","_id":"AV1sM5dc45CTOZX0c1Sg","_score":1.0,"_source":{"@timestamp":"2017-07-22T21:28:26.386Z","beat":{"hostname":"elk","name":"elk"}, "count":1, "proc":{"cmdline":"/usr/lib/systemd/systemd-udevd", "cpu":{"user":100, "user_p":0, "system":910, "total":1010, "start_time":"16:57"}, "mem":{"size":47112192, "rss":643072, "rss_p":0, "share":462848}, "name":"systemd-udevd", "pid":480, "ppid":1, "state":"sleeping", "username":"root"}, "tags":["front-end", "web-tier", "linux", "notebook"], "type":"process"}}, .....}
```



Checking topbeat in Kibana



The screenshot shows the Kibana interface with the 'Dev Tools' tab selected in the sidebar. The 'Console' section contains a search query and its results.

```
1 GET topbeat*/_search
2 {
3     "query": {"match_all": {}}
4 }
```

```
1 {
2     "took": 12,
3     "timed_out": false,
4     "_shards": {
5         "total": 5,
6         "successful": 5,
7         "failed": 0
8     },
9     "hits": {
10        "total": 2678,
11        "max_score": 1,
12        "hits": [
13            {
14                "_index": "topbeat-2017.07.22",
15                "_type": "process",
16                "_id": "AV1sM5dc45CT0ZX0c1Sg",
17                "_score": 1,
18                "_source": {
19                    "@timestamp": "2017-07-22T21:28:26.386Z",
20                    "beat": {
21                        "hostname": "elk",
22                        "name": "elk"
23                    },
24                    "count": 1,
25                    "proc": {
26                        "cmdline": "/usr/lib/systemd/systemd-udevd",
27                        "cpu": {
28                            "user": 100,
29                            "user_p": 0,
30                            "system": 910,
31                            "total": 1010,
32                            "start_time": "16:57"
33                        },
34                        "mem": {
35                            "size": 47112192,
36                            "rss": 643072,
37                            "rss_p": 0,
38                            "share": 462848
39                        },
40                        "name": "systemd-udevd",
41                        "pid": 480,
42                        "ppid": 1,
43                        "state": "sleeping",
44                        "username": "root"
45                    },
46                    "tags": [
47                        "front-end",
48                        "web-tier",
49                        "linux",
50                        "notebook"
51                    ]
52                }
53            }
54        ]
55    }
56 }
```



Adding topbeat as an index

The screenshot shows the Kibana Management interface. The left sidebar has a pink header with the Kibana logo and a teal body containing icons for Discover, Visualize, Dashboard, Timelion, Dev Tools, and Management. The main area has a grey header with 'Management / Kibana', 'Index Patterns', 'Saved Objects', and 'Advanced Settings'. A 'Warning' message says 'No default index pattern. You must select or create one to continue.' The central part is titled 'Configure an index pattern' with the sub-instruction 'In order to use Kibana you must configure at least one index pattern. Index patterns are used to identify the Elasticsearch index to run search and analytics against. They are also used to configure fields.' Below this is a form with 'Index name or pattern' set to 'topbeat-*', a note about wildcards, and a dropdown for 'Time Filter field name' set to '@timestamp'. There are two deprecated checkboxes: 'Expand index pattern when searching [DEPRECATED]' and 'Use event times to create index names [DEPRECATED]'. A 'Create' button is at the bottom.

Management / Kibana

Index Patterns Saved Objects Advanced Settings

Warning
No default index pattern. You must select or create one to continue.

Configure an index pattern

In order to use Kibana you must configure at least one index pattern. Index patterns are used to identify the Elasticsearch index to run search and analytics against. They are also used to configure fields.

Index name or pattern

topbeat-*

Patterns allow you to define dynamic index names using * as a wildcard. Example: logstash-*

Time Filter field name i refresh fields

@timestamp

Expand index pattern when searching [DEPRECATED]

With this option selected, searches against any time-based index pattern that contains a wildcard will automatically be expanded to query only the indices that contain data within the currently selected time range.

Searching against the index pattern `logstash-*` will actually query Elasticsearch for the specific matching indices (e.g. `logstash-2015.12.21`) that fall within the current time range.

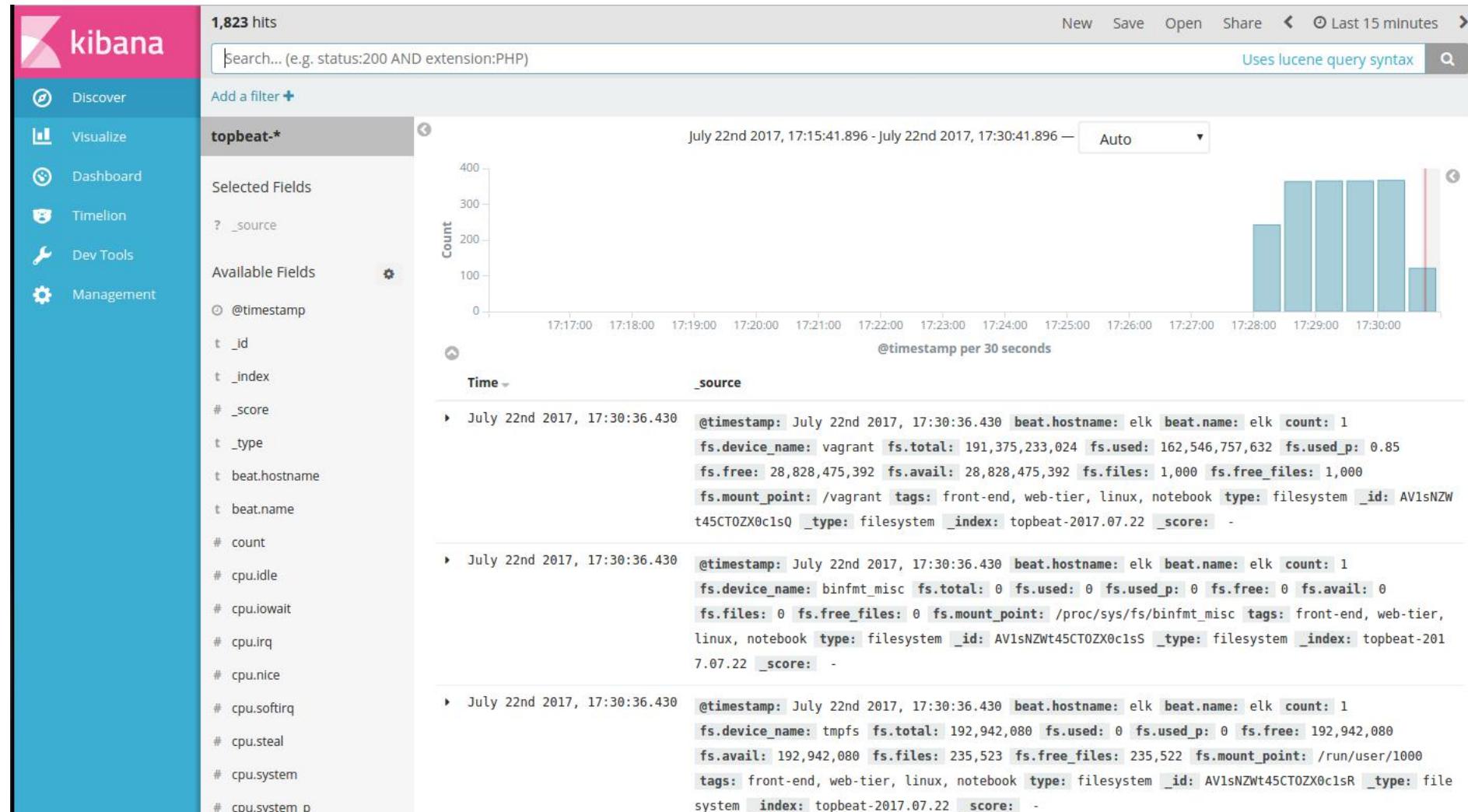
With recent changes to Elasticsearch, this option should no longer be necessary and will likely be removed in future versions of Kibana.

Use event times to create index names [DEPRECATED]

Create



Viewing Topbeat data



Congratulation

- You have completed Lab 3
- Elasticsearch is now receiving data from a Beat
- Kibana is now showing us data
- Kibana has allowed us to query the data in Dev Tools
- Kibana has allowed us to create an index for the Beat



Kibana View Lab 4

Viewing Beats and using dashboards



Visualise your data

- Load Beats Dashboards
- Setup Index Pattern
- Explore Data In Kibana Discover
- Explore Topbeat Dashboard



Download and install dashboards

- View this link
 - <https://www.elastic.co/guide/en/beats/libbeat/1.3/load-kibana-dashboards.html>
 - A newer version
 - <https://www.elastic.co/guide/en/kibana/current/dashboard.html>
 - <https://www.elastic.co/guide/en/beats/libbeat/5.5/import-dashboards.html#dashboards-archive-structure>

```
# Mac OS, Linux
# grab the dashboards, unzip them and run the load script
#curl -L -O https://artifacts.elastic.co/downloads/beats/beats-dashboards/beats-dashboards-5.5.0.zip
elk:~> curl -L -O http://download.elastic.co/beats/dashboards/beats-dashboards-1.3.1.zip
elk:~> unzip beats-dashboards-1.3.1.zip
elk:~> cd beats-dashboards-1.3.1/
elk:~> ./load.sh
```

- Windows - See Website instructions:

- <https://www.elastic.co/guide/en/beats/libbeat/current/load-kibana-dashboards.html>



Viewing the data

The screenshot shows the Kibana Discover interface. On the left, a sidebar menu includes 'Discover' (selected), 'Visualize', 'Dashboard', 'Timelion', 'Dev Tools', and 'Management'. The main area has a search bar with 'topbeat*' and a dropdown showing 'topbeat-*'. A histogram titled '@timestamp per 30 minutes' shows two bars at 17:00. Below it is a table with two rows of log entries:

Time	_source
July 22nd 2017, 17:31:46.415	proc.name: topbeat @timestamp: July 22nd 2017, 17:31:46.415 beat.hostname: elk beat.name: elk count: 1 proc.cmdline: ./topbeat -e -c topbeat.yml -d elasticsearc proc.cpu.user: 570 proc.cpu.user_p: 0.006 proc.cpu.system: 490 proc.cpu.total: 1,060 proc.cpu.start_time: 21:28 proc.mem.size: 341,553,152 proc.mem.rss: 9,822,208 proc.mem.rss_p: 0.01 proc.mem.share: 4,435,968 proc.pid: 6,816 proc.ppid: 4,945 proc.state: sleeping proc.username: vagrant tags: front-end, we
July 22nd 2017, 17:31:36.397	proc.name: topbeat @timestamp: July 22nd 2017, 17:31:36.397 beat.hostname: elk beat.name: elk count: 1 proc.cmdline: ./topbeat -e -c topbeat.yml -d elasticsearc proc.cpu.user: 540 proc.cpu.user_p: 0.003 proc.cpu.system: 460 proc.cpu.total: 1,000 proc.cpu.start_time: 21:28 proc.mem.size: 341,553,152 proc.mem.rss: 9,814,016 proc.mem.rss_p: 0.01 proc.mem.share: 4,431,872 proc.pid: 6,816 proc.ppid: 4,945 proc.state: sleeping proc.username: vagrant tags: front-end, we

1. Select Discover

3. Set index to search

2. Specify a valid range for data



Restricting data view

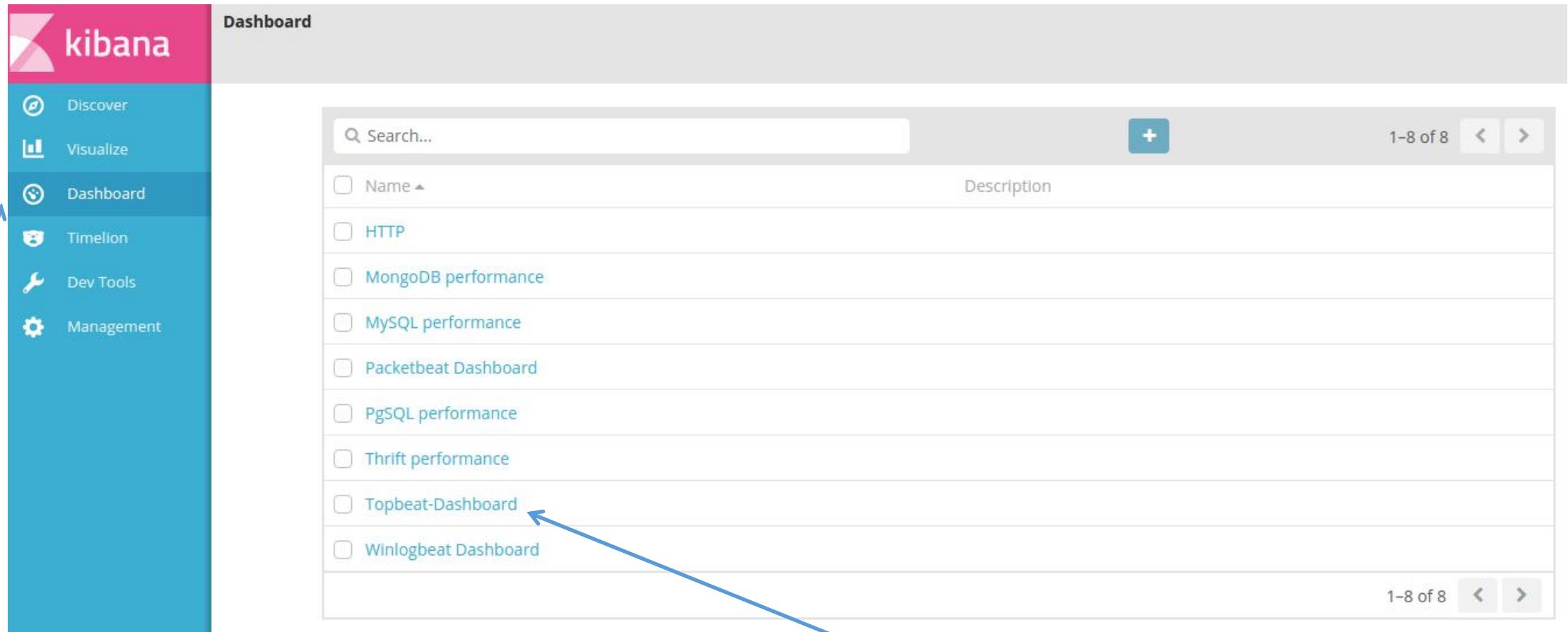
The screenshot shows the Kibana Discover interface with the following elements:

- Left Sidebar:** Includes links for Discover, Visualize, Dashboard, Timelion, Dev Tools, and Management.
- Header:** Shows "22 hits" and a search bar with the query "topbeat*". It also includes buttons for New, Save, Open, Share, and date range selection.
- Selected Fields:** A dropdown menu set to "topbeat-*" which is highlighted with a red arrow. Below it is a chart titled "Count" over time intervals from 02:00 to 23:00, showing two bars at 17:00 and 18:00.
- Available Fields:** A list of fields including @timestamp, t _id, t _index (which is highlighted with a red box and has an "add" button next to it), t _score, t _type, # count, and t proc.cpu.sta... .
- Data Table:** A table showing log entries with columns: Time, beat.hostname, beat.name, proc.cmdline, and proc.cpu.system. The table lists four entries corresponding to the bars in the chart.

1. Click add to specify column



Go to the Dashboards



The screenshot shows the Kibana interface. On the left is a sidebar with a pink header containing the Kibana logo and a teal body with the following items:

- Discover
- Visualize
- Dashboard** (highlighted)
- Timeline
- Dev Tools
- Management

The main area is titled "Dashboard" and shows a list of dashboards:

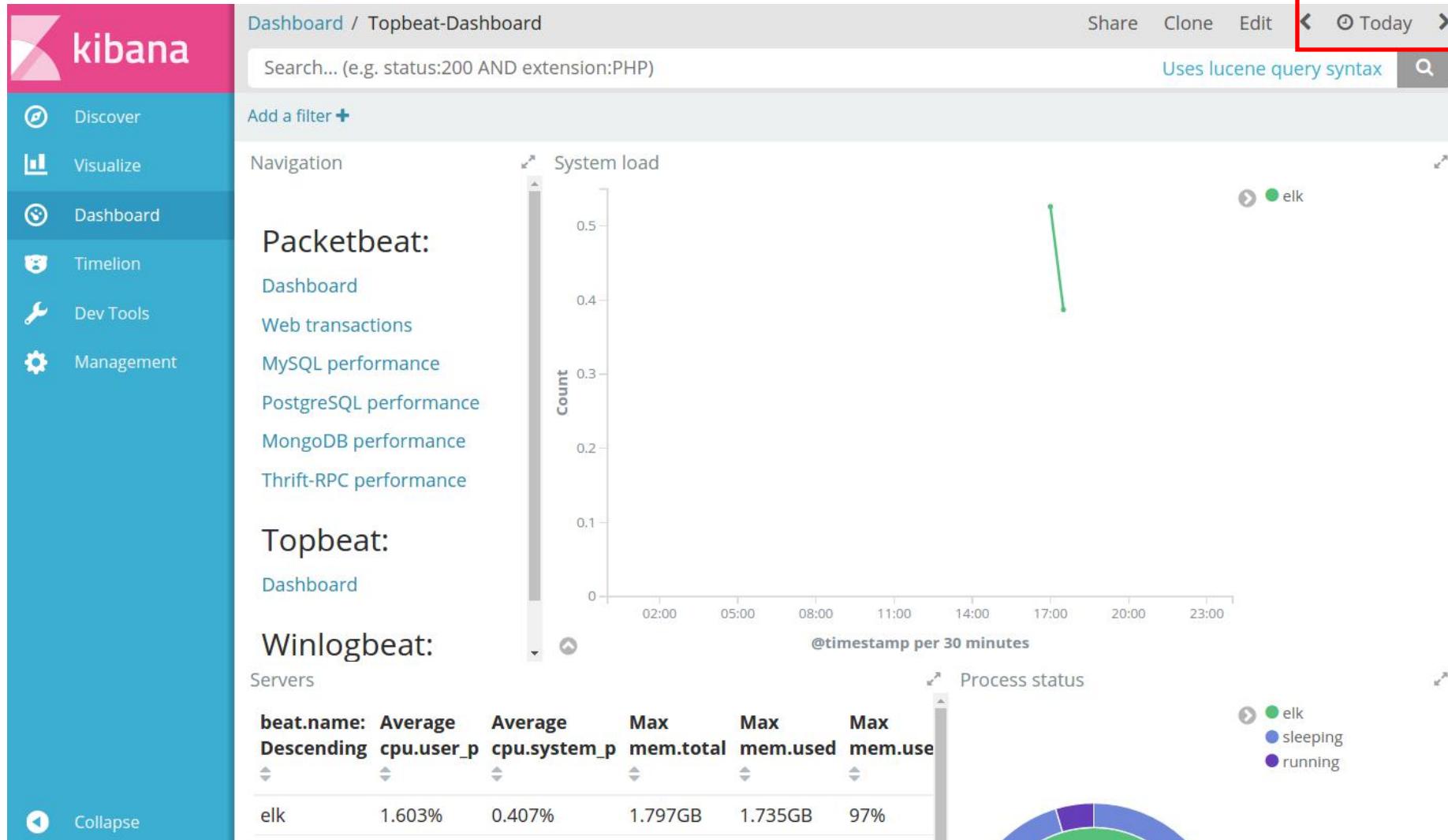
Name	Description
HTTP	
MongoDB performance	
MySQL performance	
Packetbeat Dashboard	
PgSQL performance	
Thrift performance	
Topbeat-Dashboard (highlighted)	
Winlogbeat Dashboard	

1. Select Dashboard

2. Select Topbeat-Dashboard



Topbeat-Dashboard



1. Specify a valid range for data



Explore Topbeat Dashboard

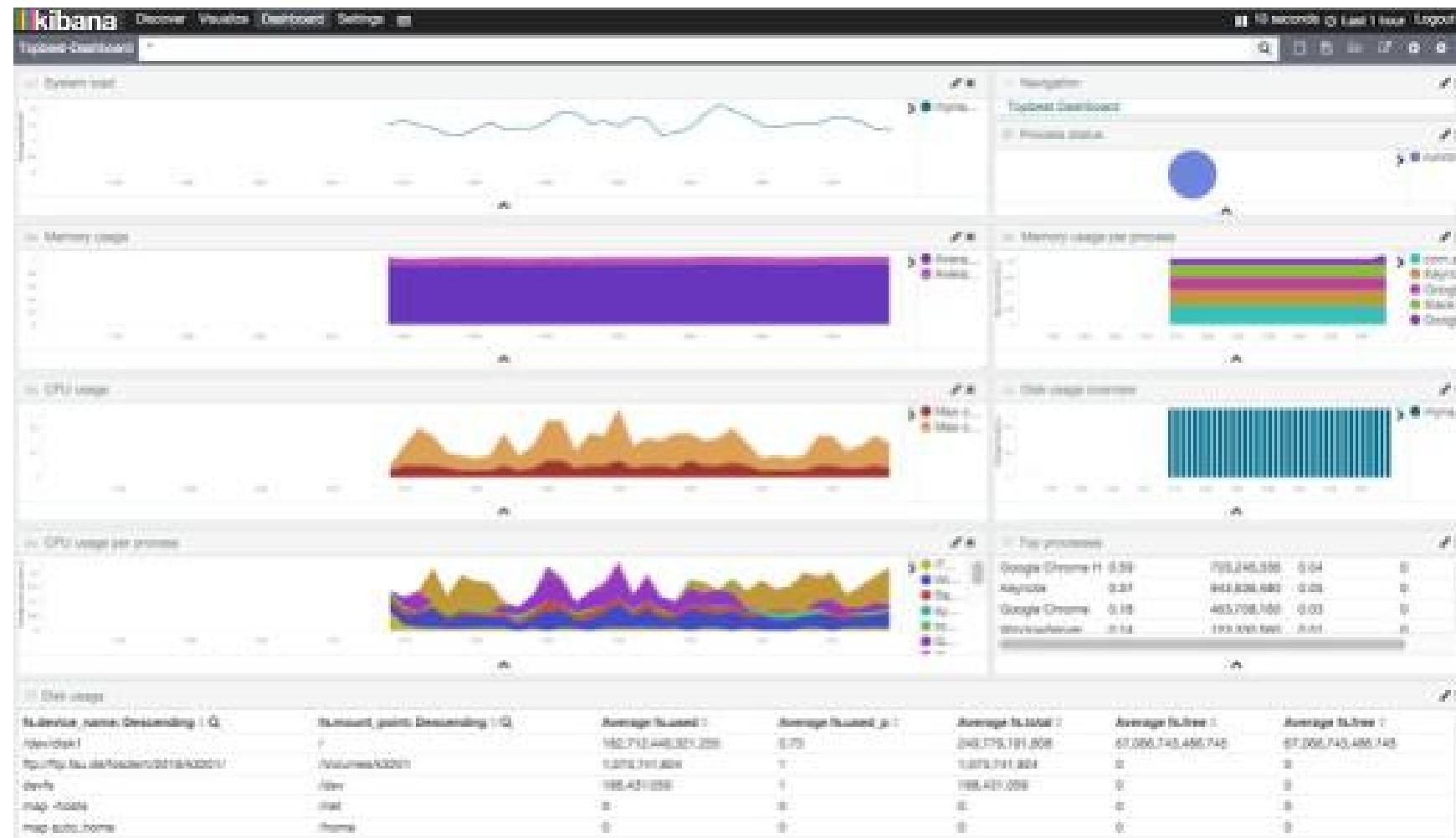
- To move visualizations
 - click in the header, drag and drop.
- To resize visualizations
 - click in the bottom-right corner, drag and drop.
- To remove visualizations
 - click on the X (top-right corner).
- Click, drag and drop inside a chart to drill down the time range.



Explore Topbeat Dashboard (2)

- Use the time-picker to set the time range
 - to 1 hour and the auto-refresh to 10 seconds.
- Click on the Topbeat Dashboard link
 - to reload default dashboard (you will lose unsaved changes).
- Save the modified dashboard with the same name
 - (overwrite) or with a new name (new dashboard).





Congratulation

- You have completed Lab 4
- You can now view and search data
- You can now load dashboards and modify them
- Now lets ingest log data



Logstash Lab 5

Installing, configuring and starting Filebeat and Logstash
Loading Apache log data



Logstash & Filebeat Logging: Steps

- Install Logstash
- Test Logstash
- Prepare Log Data
- Install Filebeat
- Test Filebeat
- Configure Logstash & Filebeat
- Ship Apache Log Data, Transform It, Load It



Download & Install Logstash

- Download Logstash.
 - www.elastic.co/downloads/logstash
- Ask your instructor for the Logdata.zip file.
- Put the following in your course directory
 - logstash_configs
 - log_data folders
- Unzip Logstash into your course directory
 - Open the "logstash_simple.conf" file in a text editor
- Look at the file configuration for logstash_simple.conf



logstash_simple.conf

- Start Logstash against this config

```
# use the standard input
input {
    stdin {}
}

# no filters filter {}

# Just output to the terminal
output {
    stdout {
        codec => rubydebug
    }
}
```



Run logstash

- Logstash will start listening to port
 - 5044 as a listener
 - 9600 as an agent

```
elk:~> cd logstash-5.5.0/config  
# We are here as it needs the log4j configuration too  
elk:~> ../bin/logstash -f logstash_simple.conf
```

```
# Windows  
C:\Users\elk\elkCourse\logstash> .\bin\logstash -f ..\logstash_configs\logstash_simple.conf
```



Logstash output

```
Sending Logstash's logs to /home/vagrant/logstash-5.5.0/logs which is now configured via log4j2.properties
[2017-07-22T23:34:57,887][INFO ][logstash.pipeline      ] Starting pipeline {"id"=>"main", "pipeline.workers"=>1,
"pipeline.batch.size"=>125, "pipeline.batch.delay"=>5, "pipeline.max_inflight"=>125}
[2017-07-22T23:34:57,956][INFO ][logstash.pipeline      ] Pipeline main started
The stdin plugin is now waiting for input:
```



Send Message To Logstash stdin Input

- With Logstash running against the logstash_simple.conf file, type some text in the terminal Logstash is running in:

```
# type "Hello world!" in the terminal  
Hello world!  
  
# you should see the following output  
# notice: the "message" field has the input  
# notice: the other fields are meta information fields  
{  
    "message" => "Hello world!", "@version" => "1",  
    "@timestamp" => "2016-06-15T23:09:11.981Z",  
    "host" => "your-host-name"  
}
```



Using grok

- Change "logstash_simple.conf" to contain a grok filter as follows:

```
input {
    stdin {}
}

filter {
    grok {
        match => {
            "message" => '%{HTTPDATE:timestamp} %{IP:ip} <%{DATA:msg}>'
        }
    }
}

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



Restart logstash

- Any changes to logstash configuration files requires restart
 - Only the logstash service

```
# copy and paste the following in the terminal;
22/Mar/2016:16:38:00 -0700 183.60.215.50 <text at the end>
# you should see the following output
# notice: the <> plays an important part in the grok
# notice: there are 2 timestamp fields with 2 different values
{
    "msg" => "text at the end",
    "@timestamp" => 2017-07-22T23:56:05.596Z,
        "ip" => "183.60.215.50",
    "@version" => "1",
        "host" => "elk",
    "message" => "22/Mar/2016:16:38:00 -0700 183.60.215.50 <text at the end>",
    "timestamp" => "22/Mar/2016:16:38:00 -0700"
}
```



Logstash is running

- Documents are being transformed
- Data coming from stdin and going to stdout
- Let's prepare a real log



Prepare Apache Log Data

- We want the log data to be relevant to the time you are taking this course.
 - This way we can use the Time Picker in Kibana with settings like "Last 30 Days".
 - Otherwise you would be stuck using "old" log data - and that's no fun!
- First, stop Logstash since we will use Logstash to create the log data for us using a prepared Logstash config file

```
# to stop the logstash execution there are two options  
ctrl+C and then press enter  
ctrl+D
```



Edit the log data

- Look inside the "log_data" directory provided to you.
- It should be in your "course" directory if you downloaded and extracted it there:

```
# list the log_data directory
~/course/log_data$ ls
convert_times.conf    original.log
```



Convert the data

```
# Mac/Linux  
cat log_data/original.log | ../../logstash-5.5.0/bin/logstash -f log_data/convert_times.conf  
  
# Windows  
type log_data/original.log | ..\logstash-5.5.0\bin\logstash -f log_data\convert_times.conf  
  
# you will see this on your screen after Logstash starts running:  
Settings: Default pipeline workers: 4  
Pipeline main started  
.....  
.....  
.....  
# the dots show that the process is working
```



Verify conversion

- On completion you should see

```
.....  
2017-07-23T00:07:21,263][WARN ][logstash.agent] stopping pipeline {:id=>"main"}
```

- Directory listing should have a new file called "access.log":

```
~/course/log_data$ ls  
access.log  convert_times.conf  original.log
```



Log file is prepared

- Now we'll configure a beat to talk to logstash



Beats & Logstash

- All beats can be sent through logstash
- Logstash acts as a broker in this case
- Logstash can be on a different host to the Beat



Download & Install Filebeat

- Download Filebeat:
 - www.elastic.co/downloads/beats/filebeat
- Unzip Filebeat into your course directory
- Open the "filebeat.yml" file
- We will now go through some changes



Configure Filebeat input settings

- Read apache web server log file
- Let's change the log file paths

```
filebeat.prospectors:
```

```
- input_type: log
  - <path_to_home>/courseware/access.log
  # - /var/log/*.log
  # - c:\courseware\log_data\access.log
```



Configure the Filebeats shipper

- Setup to tag every document (log line) with the server properties

```
shipper:  
# Tags make it easy properties.  
tags: ["front-end", "web-tier", "apache", "desktop"]
```



Configure Filebeats output

- Setup to send data to logstash running on your localhost

```
#----- Elasticsearch output -----
#output.elasticsearch:
# # Array of hosts to connect to.
#hosts: ["localhost:9200"]

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

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



Examine logstash_stdout.conf

- If Logstash is currently running, stop it
- We will start it with a new config
- Open and examine the "logstash_configs/logstash_stdout.conf"



The input section

```
input {  
  beats {  
    host => "localhost"  
    port => 5044  
    congestion_threshold => 30  
  }  
}
```



Filter section

```
filter {  
    grok {  
        match => {  
            "message" => '%{IPORHOST:clientip} %{USER:ident} %{USER:auth} \[%{HTTPDATE:timestamp}\] "%{WORD:verb}  
            %{DATA:request} HTTP/%{NUMBER:httpversion}" %{NUMBER:response:int} (?:-| %{NUMBER:bytes:int})  
            %{QS:referrer} %{QS:agent}'  
        }  
    }  
  
    date {  
        match => [ "timestamp", "dd/MMM/YYYY:HH:mm:ss Z" ]  
        locale => en  
        remove_field => timestamp  
    }  
  
    geoip {  
        source => "clientip"  
    }  
  
    useragent {  
        source => "agent"  
        target => "useragent"  
    }  
}
```



Output section

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



Start Logstash

- Against the logstash_stdout.conf
 - Working out of the "logstash" directory:

```
# Mac OS, Linux:
```

```
./bin/logstash -f ../courseware/logstash_configs/logstash_stdout.conf
```

```
# Windows:
```

```
.\bin\logstash -f ..\courseware\logstash_configs\logstash_stdout.conf
```

```
# you should see something like the following
```

```
Settings: Default pipeline workers: 4 Logstash startup completed
```



Run Filebeat

- Logstash is waiting for Filebeat to ship it logs - start Filebeat

```
# go back to the filebeat directory in a new terminal and run filebeat  
  
# linux, mac  
.filebeat -e -c filebeat.yml -d "logstash"  
  
# windows  
.filebeat.exe -e -c filebeat.yml -d "logstash"
```



Test the Filebeat/Logstash Integration

- Check if documents are being printed in the terminal

```
# in the logstash-terminal-window you should see documents being printed

{
    "message" => "46.105.14.53 - - [14/Jan/2016:22:36:26 +0000] \"GET /blog/tags/puppet?flav=rss20 HTTP/1.1\" 200 14872 \"-\"
\"UniversalFeedParser/4.2- pre-314-svn +http://feedparser.org/\",
    "@version" => "1",
    "@timestamp" => "2016-02-10T01:05:18.520Z",
        "beat" => { "hostname" => "your_host_name", "name" => "front001" }, "count" => 1,
        "fields" => nil,
    "input_type" => "log",
        "offset" => 3799864
    "source" => "/home/vagrant/courseware/access.log",
        "tags" => [ [0] "front-end", [1] "apache", [2] "web-tier", ... ],
        "type" => "log",
    "host" => "your_host_name"
}
```



Now let's send it to logstash

- Stop Filebeat and Logstash
- Kill Logstash and Filebeat and clean reading state

```
# in the logstash-terminal-window kill the logstash process  
Ctrl-C  
# in the filebeat-terminal-window kill the filebeat process  
Ctrl-C  
  
# Attention: filebeat creates a '.filebeat' file with last reading state.  
# To read the entire log file in the next execution, you need to delete .filebeat before restart.  
# We want all the data in Elasticsearch, so let's delete it.  
# It may or may not exist  
  
rm .filebeat
```



Configure logstash output

- Set the logstash_stdout.conf to elasticsearch
 - Edit courseware/logstash_configs/logstash_stdout.conf

```
# there is a file in the sample folder called logstash_elasticsearch.conf
# with a more complex config. Check it out!

vim ..../courseware/logstash_configs/logstash_elasticsearch.conf

# also, edit the output to add your Elasticsearch host  output {
# for each event prints a dot (.) in the stdout
  stdout { codec => dots }

  elasticsearch {
    hosts => 'localhost:9200'
  }
}
```



Run logstash

- Logstash will start listening to port 5044, but no data yet...

```
# execute logstash with elasticsearch config from the logstash directory:  
# linux, mac  
.bin/logstash -f ..\courseware\logstash_configs\logstash_elasticsearch.conf  
  
# windows  
.bin\logstash -f ..\courseware\logstash_configs\logstash_elasticsearch.conf  
  
# you should see something like the following  
  
Settings: Default pipeline workers: 4 Logstash startup completed
```



Run filebeat

- Start sending apache log events to Logstash
 - one line is one event

```
# in the filebeat-terminal-window start filebeat

# linux, mac
./filebeat -e -c filebeat.yml -d "logstash"

# windows
.\filebeat.exe -e -c filebeat.yml -d "logstash"
```

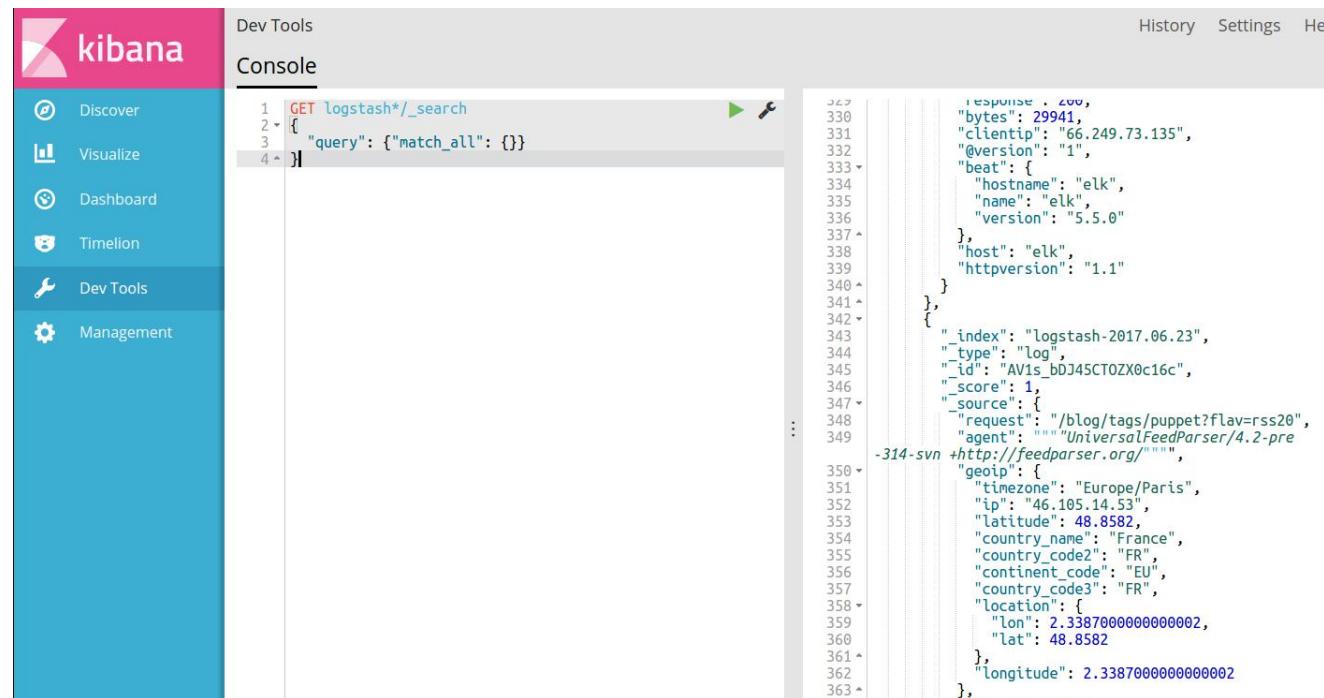
You should see lots of dots in the logstash window!!!!



Checkout Kibana

- In the Dev Tools section
- Type the following into the Console

```
GET logstash*/_search
{
  "query": {"match_all": {}}
}
```



The screenshot shows the Kibana interface with the Dev Tools section selected. In the Dev Tools Console, the following Elasticsearch search query is typed:

```
1 GET logstash*/_search
2 {
3   "query": {"match_all": {}}
4 }
```

The results pane displays the search results, which include a single document from the 'logstash-2017.06.23' index. The document contains the following fields:

```
347 response : 200,
348 "bytes": 29941,
349 "clientip": "66.249.73.135",
350 "@version": "1",
351 "beat": {
352   "hostname": "elk",
353   "name": "elk",
354   "version": "5.5.0"
355 },
356 "host": "elk",
357 "httpversion": "1.1"
358 },
359 {
360   "_index": "logstash-2017.06.23",
361   "_type": "log",
362   "_id": "AV1s_DDJ45CT0ZX0c16c",
363   "_score": 1,
364   "_source": {
365     "request": "/blog/tags/puppet?flav=rss20",
366     "agent": "UniversalFeedParser/4.2-pre-314-svn +http://feedparser.org/",
367     "geoip": {
368       "timezone": "Europe/Paris",
369       "ip": "46.105.14.53",
370       "latitude": 48.8582,
371       "country_name": "France",
372       "country_code2": "FR",
373       "continent_code": "EU",
374       "country_code3": "FR",
375       "location": {
376         "lon": 2.3387000000000002,
377         "lat": 48.8582
378       },
379       "longitude": 2.3387000000000002
380     }
381   }
382 }
```



Congratulation

- You have completed Lab 5
- You have now configured logstash to forward Beats
- You can now configure different outputs of logstash
- You can perform some filtering



Enhanced Logstash Lab 6

Doing more work with Logstash



Customizing Logstash

- Using different inputs
- GROKing
- Section references



Different inputs

- Logstash can take data from many sources
- Inputs get their data from plugins
 - <https://www.elastic.co/guide/en/logstash/current/input-plugins.html>
- If there isn't one then you can use
 - exec
 - Run a custom command
 - beats
 - Build your own Beat



Example Yahoo data feed

- This returns a csv formated output
 - "Microsoft Corporation",73.60,73.55

```
input {  
    exec {  
        command => "curl -s 'http://download.finance.yahoo.com/d/quotes.csv?s=MSFT&f=nab'"  
        type => stock  
        interval => 30  
    }  
}
```



Filter the data (groking)

- Define the format of the input stream
- Apply names to the data
 - stock_name
 - askprice
 - bidprice

```
filter {
  grok {
    match => {"message" => "%{QUOTEDSTRING:stock_name},%{NUMBER:askprice:float},%{NUMBER:bidprice:float}"}
    # This checks the message field for a Number and creates a stockprice field of type float
  }
}
```



Remove dangerous characters

- Mutate the data
- Comes after you assign the field name from grok
 - This example removes the dot

```
mutate {  
  gsub => [  
    "stock_name", '\.', ''  
  ]  
}  
}
```



Filter reference

<https://www.elastic.co/guide/en/logstash/current/filter-plugins.html>



Output plugins

- Logstash can send to other systems
 - Not specific to Elasticsearch

<https://www.elastic.co/guide/en/logstash/current/output-plugins.html>



Lab

- Implement the Yahoo feed into your logstash
- Get information for the following stocks
 - IBM
 - GOOGLE
 - MICROSOFT
 - APPLE



Kibana Customising Lab 7

Creating custom Kibana Visualizations and Dashboards for the
Apache log data



Visualize Overview

- Configure 'logstash-*' Index Pattern
- Create a Metrics Visualization
- Create a Pie Chart
- Create a Bar Chart
- Create a Line Chart
- Create a Tile Map
- Create a Dashboard
- Create Extra Visualizations



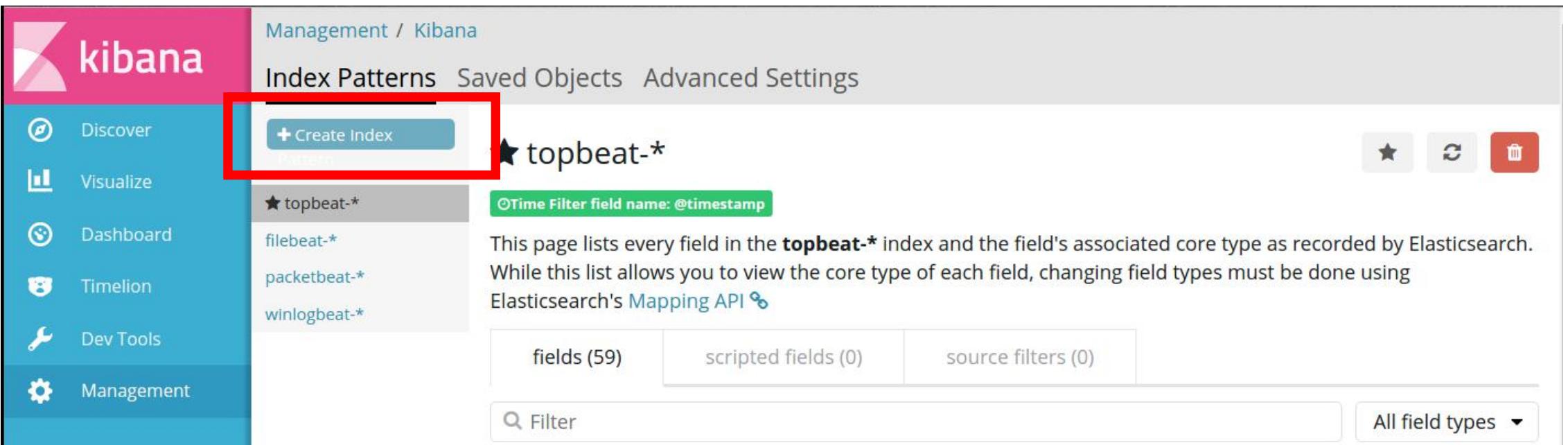
Set The Time Picker To "Last 30 Days"

- Kibana needs to see a time range with data



Configure logstash-* index

- Go to the Management screen
- Click Create Index



The screenshot shows the Kibana Management interface. On the left is a sidebar with icons for Discover, Visualize, Dashboard, Timelion, Dev Tools, and Management. The Management icon is selected. The main area has a header with tabs: Index Patterns, Saved Objects, and Advanced Settings. A red box highlights the blue '+ Create Index' button. Below it, a list of index patterns is shown: topbeat-* (selected), filebeat-* (disabled), packetbeat-* (disabled), and winlogbeat-* (disabled). A green note says 'Time Filter field name: @timestamp'. The main content area describes the fields in the topbeat-* index and how to change types using the Mapping API. At the bottom are buttons for fields (59), scripted fields (0), source filters (0), a filter input, and a dropdown for All field types.



Create the index pattern

The screenshot shows the Kibana Management interface. On the left, there's a sidebar with icons for Discover, Visualize, Dashboard, Timelion, Dev Tools, and Management. The Management option is selected. The main area has a header 'Management / Kibana' with tabs for Index Patterns, Saved Objects, and Advanced Settings. Below the header, there's a list of existing index patterns: topbeat-* (with a star icon), filebeat-* (without a star), packetbeat-* (without a star), and winlogbeat-* (without a star). To the right, there's a section titled 'Configure an index pattern'. It contains a text input field labeled 'Index name or pattern' which currently contains 'logstash-*'. A red box highlights this input field. A blue arrow points from the text 'Type in the pattern' at the top right towards this input field. Below the input field, there's a note: 'Patterns allow you to define dynamic index names using * as a wildcard. Example: logstash-*'. There are also two checkboxes: 'Time Filter field name' set to '@timestamp' and 'Expand index pattern when searching [DEPRECATED]'. Another note explains that this checkbox allows searches against any time-based index pattern containing a wildcard to query only matching indices within the current time range. A third checkbox, 'Use event times to create index names [DEPRECATED]', is also present. At the bottom, there's a 'Create' button.

Existing patterns

Type in the pattern



Creating visuals

2. Create new chart

The screenshot shows the Kibana interface. On the left, there is a sidebar with the following options:

- Discover
- Visualize** (highlighted with a red arrow)
- Dashboard
- Timelion
- Dev Tools
- Management

The main area is titled "Visualize". It contains a search bar labeled "Search...", a button with a plus sign "+", and a pagination indicator "1-20 of 67" with arrows. Below these are several visualization cards:

Name	Type
Average system load across all systems	42 Metric
CPU usage	Area
CPU usage per process	Area
Cache transactions	Vertical Bar

1. Select Visualize

TPS Services Ltd



Copyright 2017 – Course Title

Choose type

Start typing the type name to filter

The screenshot shows the Kibana interface with a pink header containing the 'kibana' logo. The main title is 'Visualize / New'. Below it, the sub-section title is 'Select visualization type'. A search bar with the placeholder 'Search visualization types...' is present. A red arrow points from the text 'Start typing the type name to filter' to this search bar. The interface is divided into sections: 'Basic Charts' and 'Advanced Charts'. Under 'Basic Charts', there are five cards: 'Area', 'Heat Map', 'Horizontal Bar' (which is highlighted with a blue border), 'Line', and 'Pie'. Each card has a small icon above its name. A tooltip 'Assign a continuous variable to each axis' is visible near the bottom of the 'Horizontal Bar' card. On the far left, a sidebar menu lists 'Discover', 'Visualize' (selected), 'Dashboard', 'Timelion', 'Dev Tools', and 'Management'.



Data source

Select a new Index to create the search

Visualize / New / Choose search source

From a New Search, Select Index

Filter... 5 of 5

Name ▲
filebeat-*
logstash-*
packetbeat-*
topbeat-*
winlogbeat-*

Choose an existing search

Or, From a Saved Search

Saved Searches Filter... 1-20 of 23 Manage saved searches

Name ▲
Cache transactions



Create the metric

Select logstash-*

Visualize / New / Choose search source

From a New Search, Select Index

Name ▲

- filebeat-*
- logstash-*** ↓
- packetbeat-*
- topbeat-*
- winlogbeat-*

Or, From a Saved Search

Saved Searches Filter...

Name ▲

- Cache transactions



Metric visualization

You can change the font size in the options menu.

The screenshot shows the Kibana Visualize interface for creating a new visualization. The left sidebar has links for Discover, Visualize, Dashboard, Timelion, Dev Tools, and Management. The main area is titled "Visualize / New Visualization (unsaved)". It includes a search bar, filter buttons, and a "logstash-*" index pattern. The "Metrics" section is active, showing a dropdown set to "Count". A red arrow points from the "Options" button at the top right of the metrics panel to a callout text: "Use this button to apply changes when not dimmed". Another red arrow points from the "Count" button in the metrics panel to a callout text: "By default you will see the count of documents. You can change the metric to average, sum, min. Or even add multiple metrics in the same visualizations." A large "165,985" is displayed below the metrics panel. The bottom of the interface has "Save", "Share", "Refresh", and date range buttons.

Count
165,985

By default you will see the count of documents. You can change the metric to average, sum, min. Or even add multiple metrics in the same visualizations.



Saving

2. Enter the name to save the visual

1. Save it

3. Save the visual

The screenshot shows a visualization interface with the following elements:

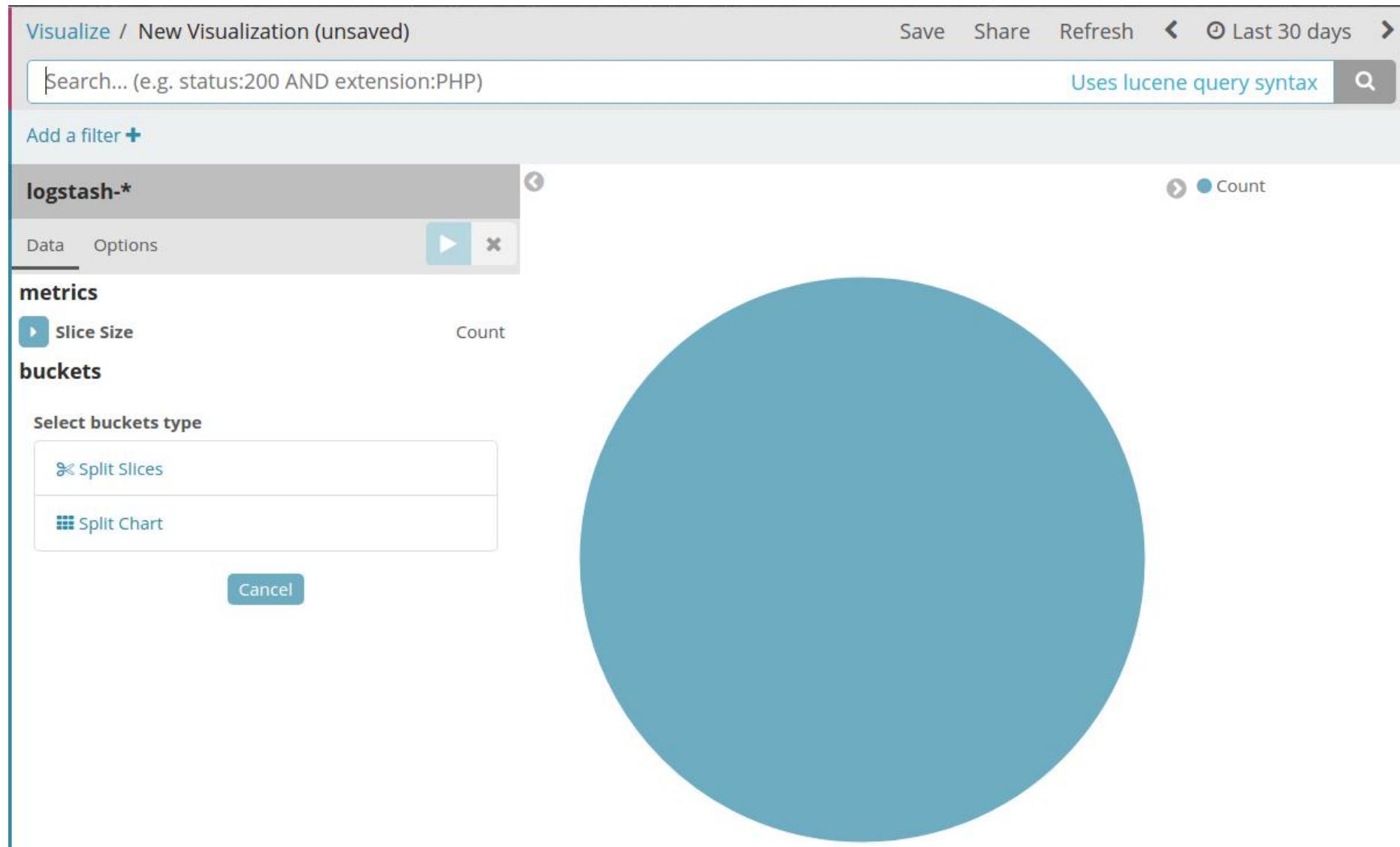
- Header: Visualize / Total Events (unsaved)
- Toolbar: Save, Share, Refresh, Last 30 days
- Section: Save Visualization
- Input field: Total Events (highlighted with a blue border)
- Save button: A teal button labeled "Save" (highlighted with a red arrow)
- Search bar: Search... (e.g. status:200 AND extension:PHP)
- Help text: Uses lucene query syntax
- Icon: A magnifying glass icon next to the search bar

Red arrows indicate the following steps:

- A vertical red arrow points from the text "1. Save it" down to the "Save" button.
- A diagonal red arrow points from the text "2. Enter the name to save the visual" down to the input field containing "Total Events".
- A diagonal red arrow points from the text "3. Save the visual" down to the "Save" button.



Pie chart



Split by top 10 responses

Add a filter +

logstash-*

Data Options

metrics

Slice Size Count

buckets

Split Slices x

Aggregation

Date Histogram
Histogram
Range
Date Range
IPv4 Range
Terms Selected
Filters
Significant Terms

Create a terms aggregation

logstash-*

Data Options

Slice Size

buckets

Split Slices x

Aggregation

Terms

Field

Select a field

- number
- bytes
- geoip.dma_code
- geoip.latitude
- geoip.longitude
- offset
- response

Select the response field

logstash-*

Data Options

Slice Size

buckets

Split Slices x

Aggregation

Terms

Field

response

Order By

metric: Count

Order Size

10

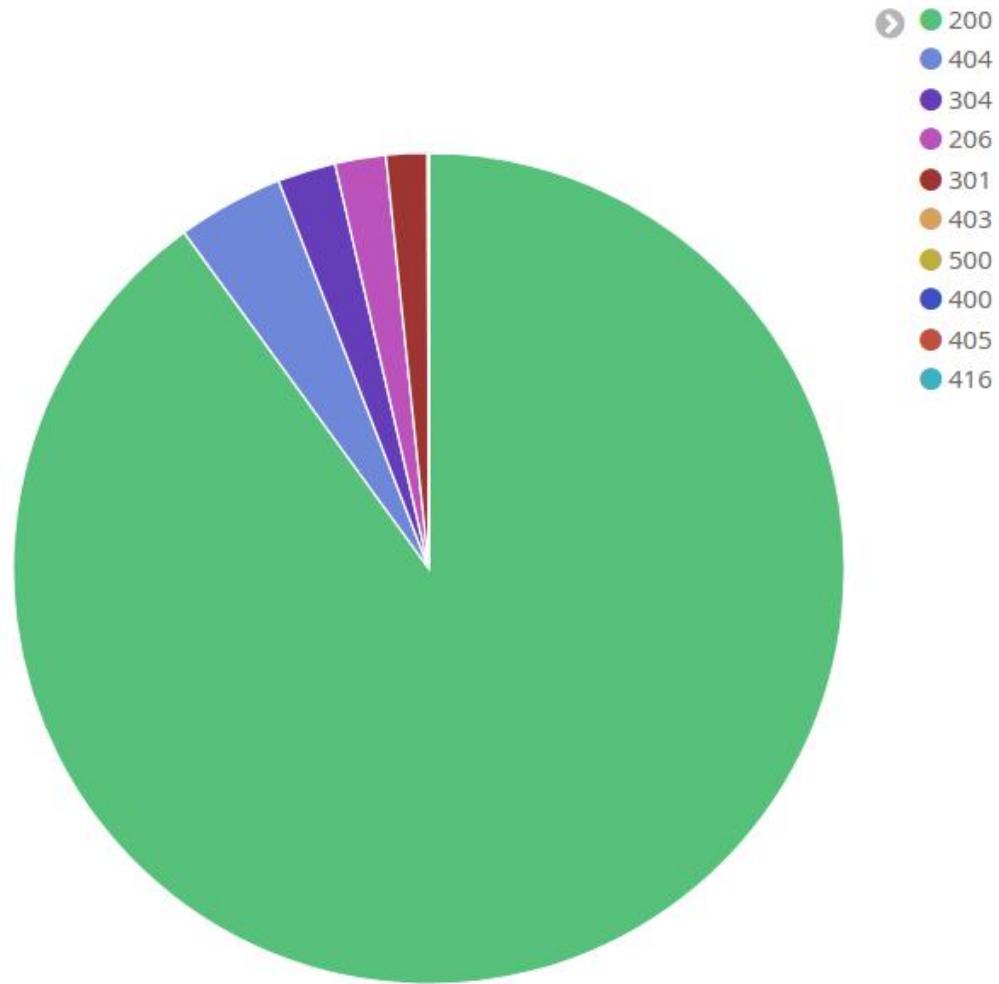
Custom Label

Set the size to 10 and click
Apply Changes



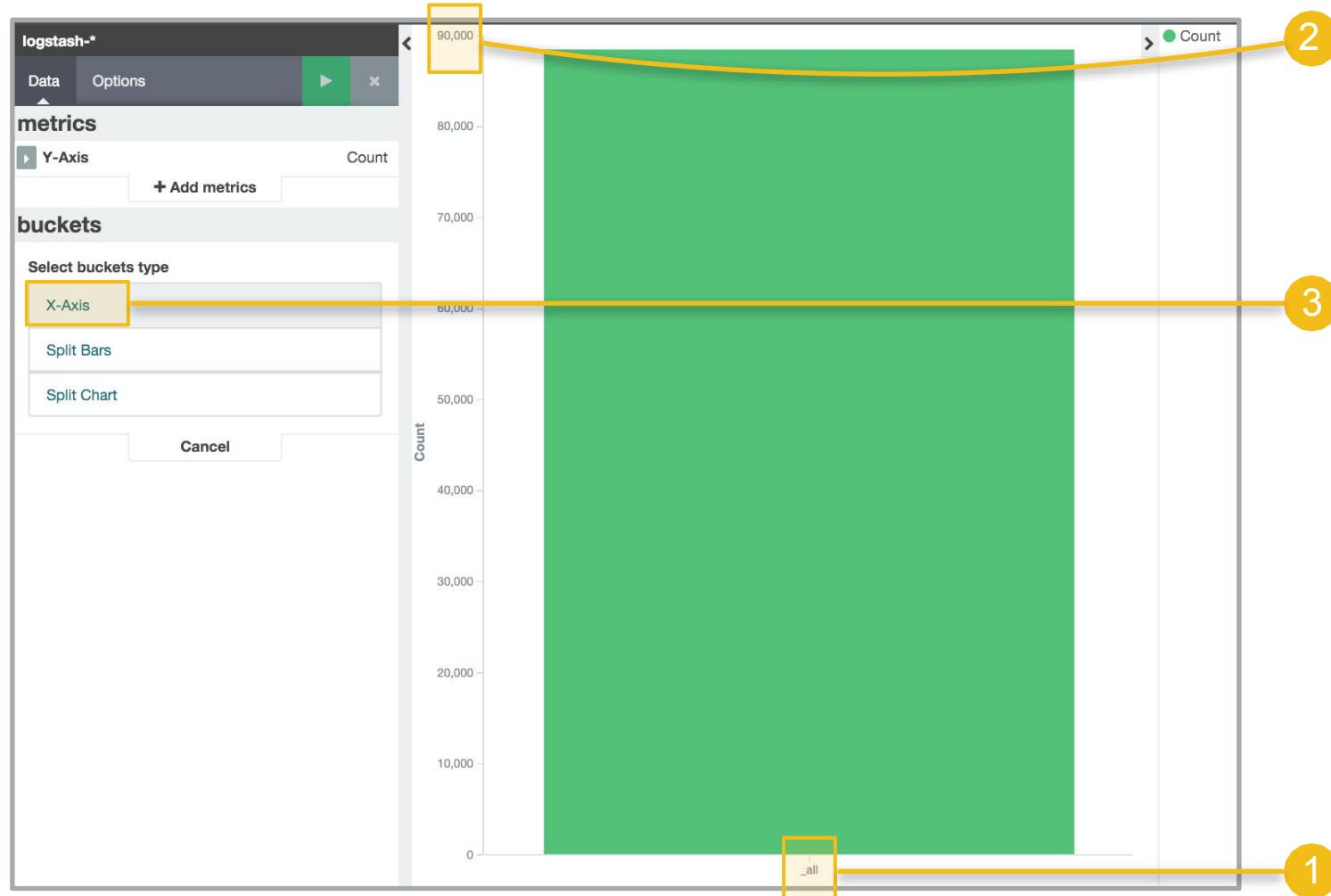
Pie chart

Do not forget to save and name your visualization (e.g. Response Codes)



Bar chart

Create a new visualization of type **Vertical bar chart**



Y-Axis represents the count.

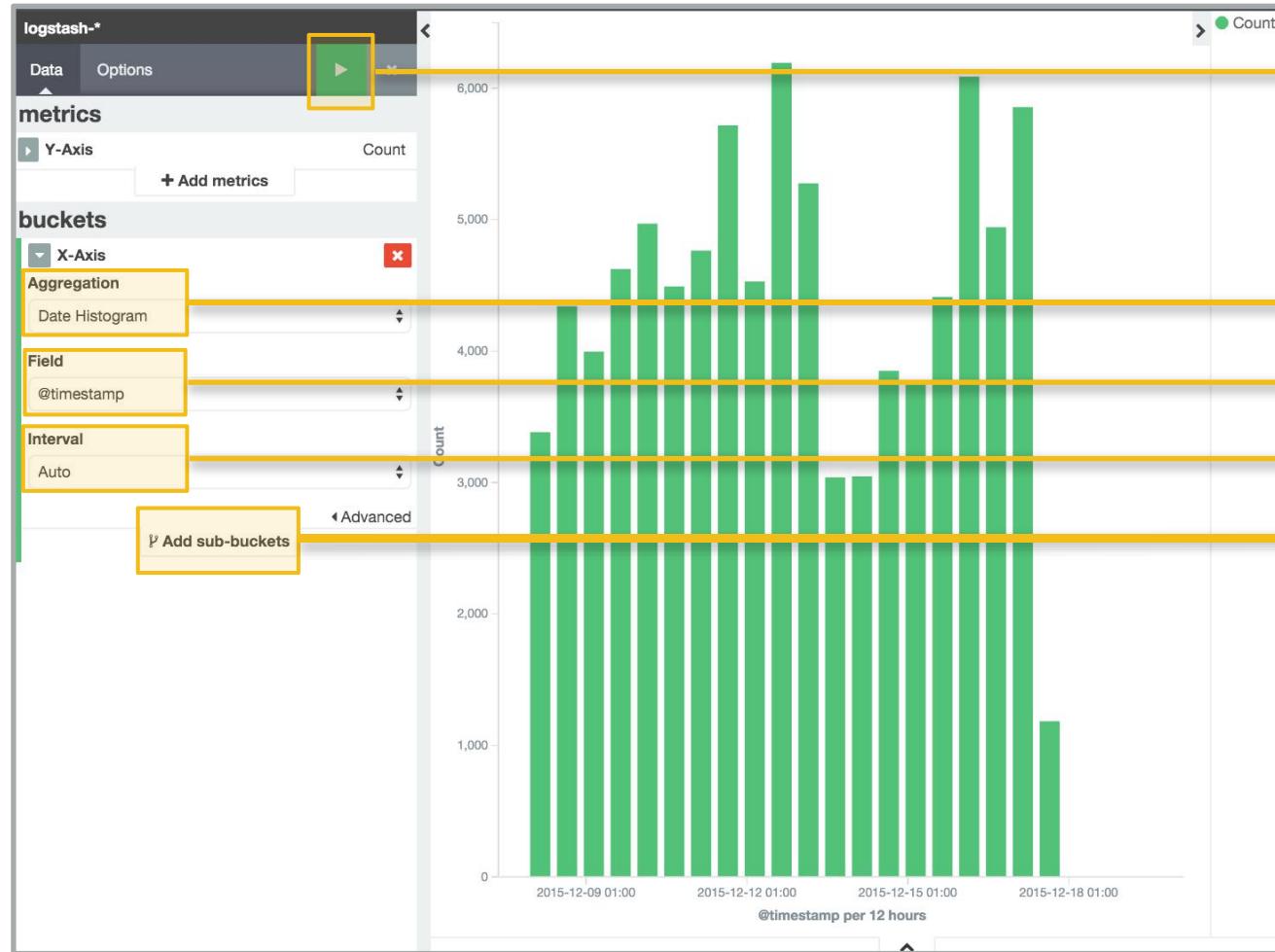
Split the X-Axis into many buckets.

All documents in a single bar (bucket).



Bar chart

Split the X-Axis into multiple bars based on the event time



Click apply.

Select **Date Histogram** aggregation.

On the field **@timestamp**.

Set the bucket **Interval** to **Auto**.

Let's sub-divide each bar.



Bar chart

Sub-divide each bar into 5 bars based on the response code



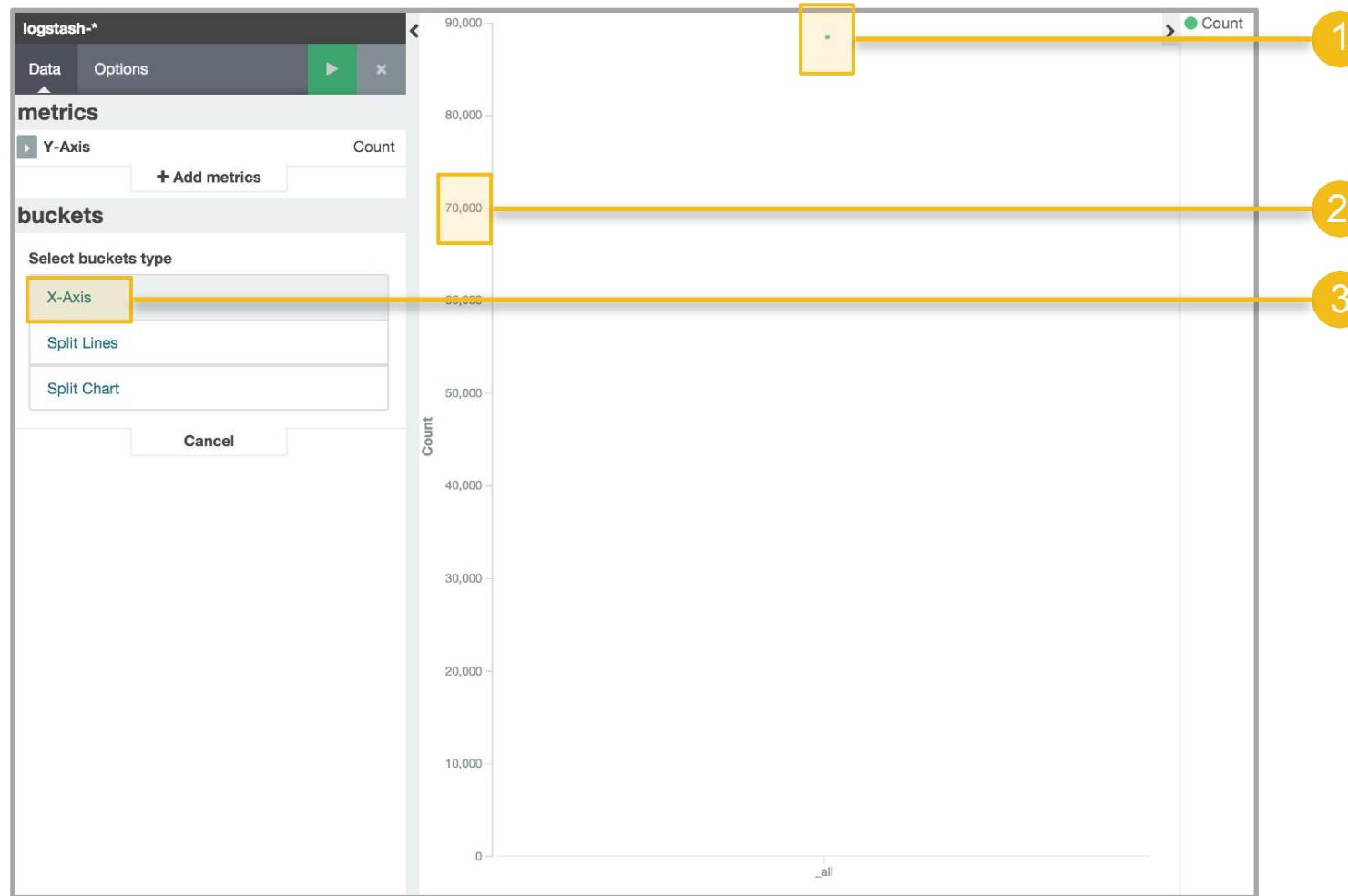
Click apply. And do not forget to save the visualization (e.g. **Events over Time by Response Code**).

- 1 Select **Terms** aggregation.
- 2 On the field **response**.
- 3 Order by the number of events.
- 4 Sub-divide each bar into a maximum of 5 bars.
- 5 Order in descending order.
- 6 Click apply. And do not forget to save the visualization (e.g. **Events over Time by Response Code**).



Line chart

Create a new visualization of type Line chart



All documents in a single point (bucket).

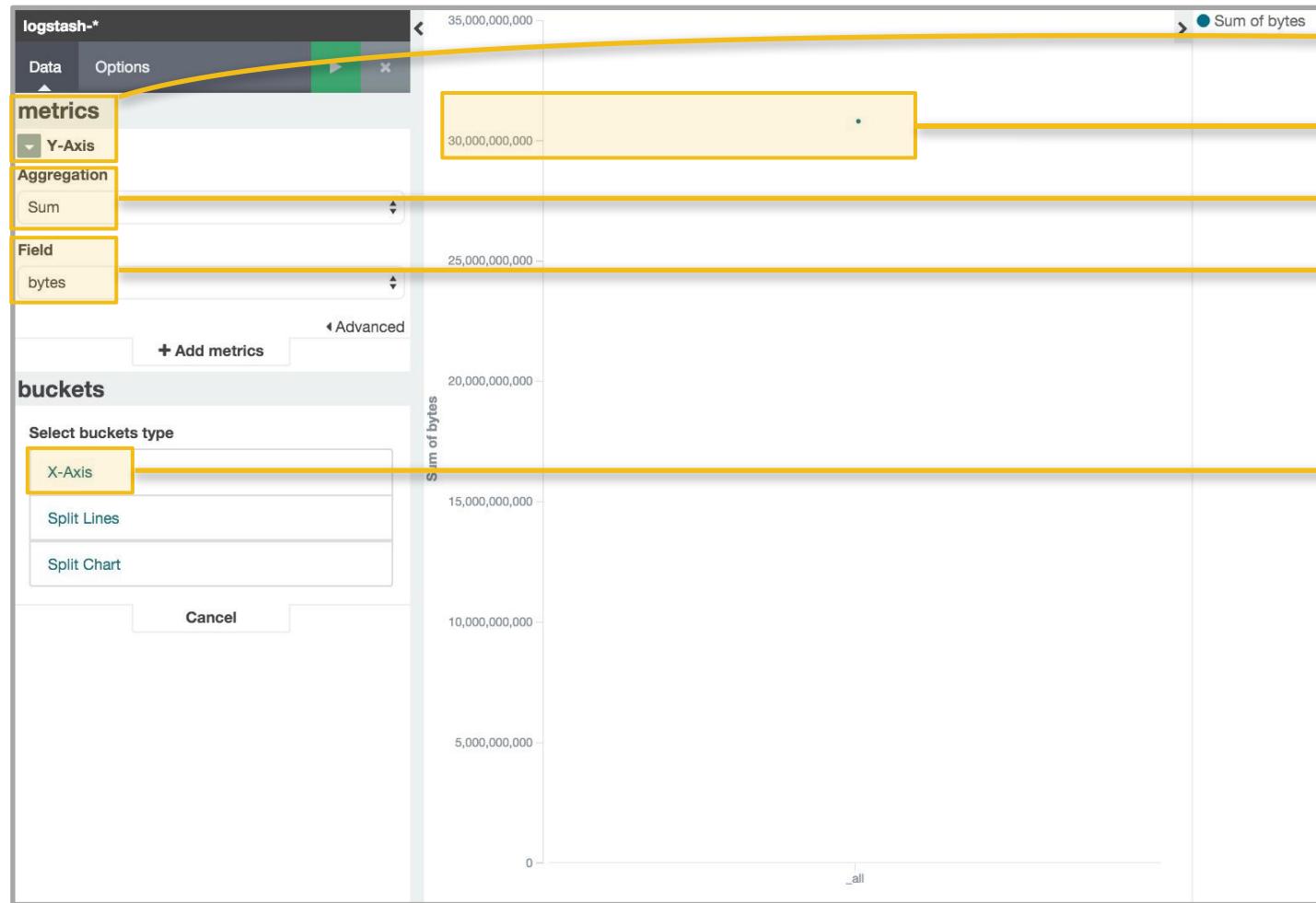
Y-Axis represents the count.

Split the X-Axis into many buckets.



Line chart

Instead of documents, let's "count" bytes served

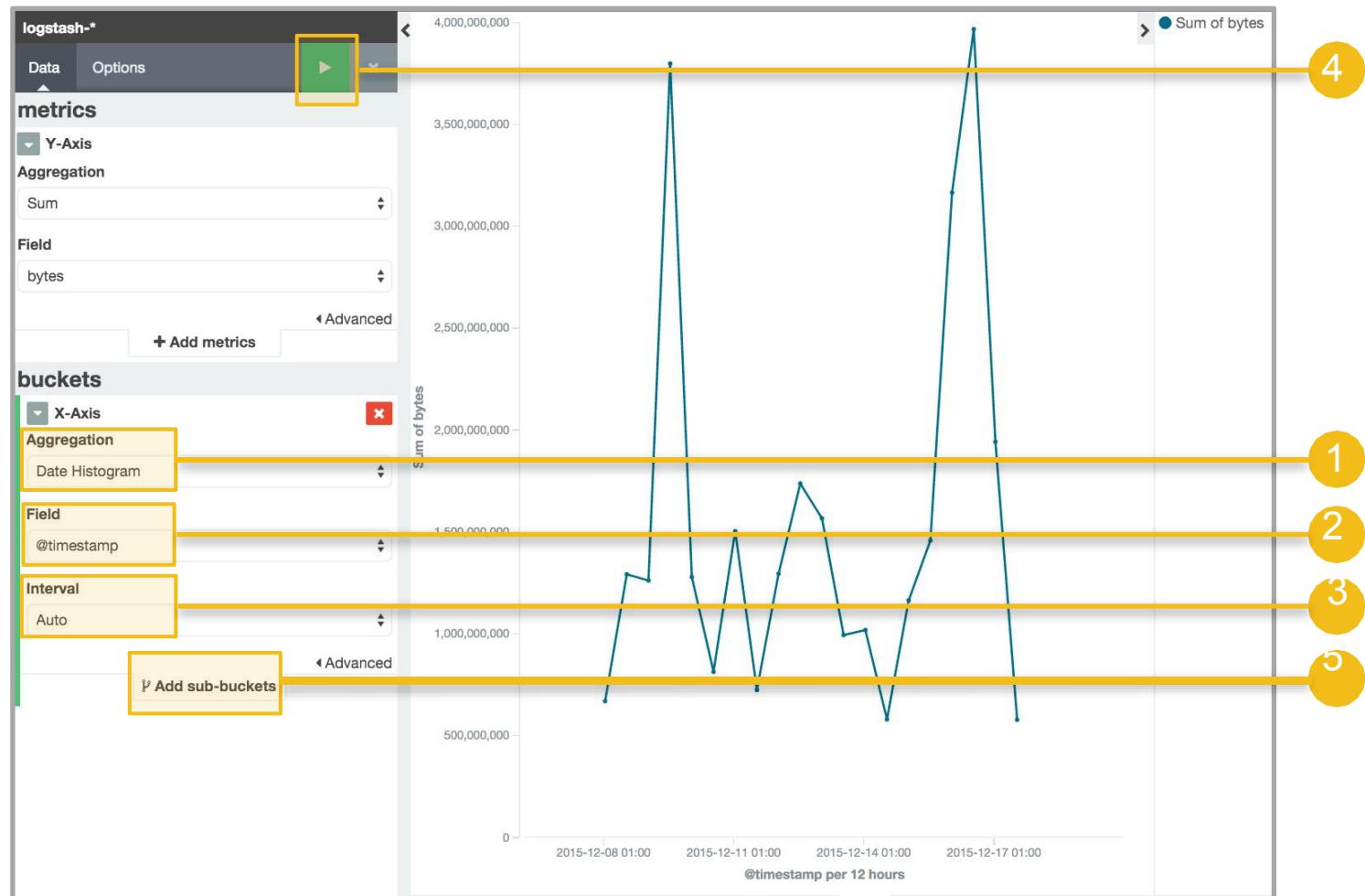


- 1 Open the Y-Axis metric, so we can change it.
- 2 After the apply button, you will still see one bucket, but with a very different value.
- 3 Sum values.
- 4 On the field **bytes**.
- 5 Split the X-Axis into many buckets (points).



Line chart

Split the X-Axis into multiple bars based on the event time



Click apply.

1 Select Date Histogram aggregation.

2 On the field @timestamp.

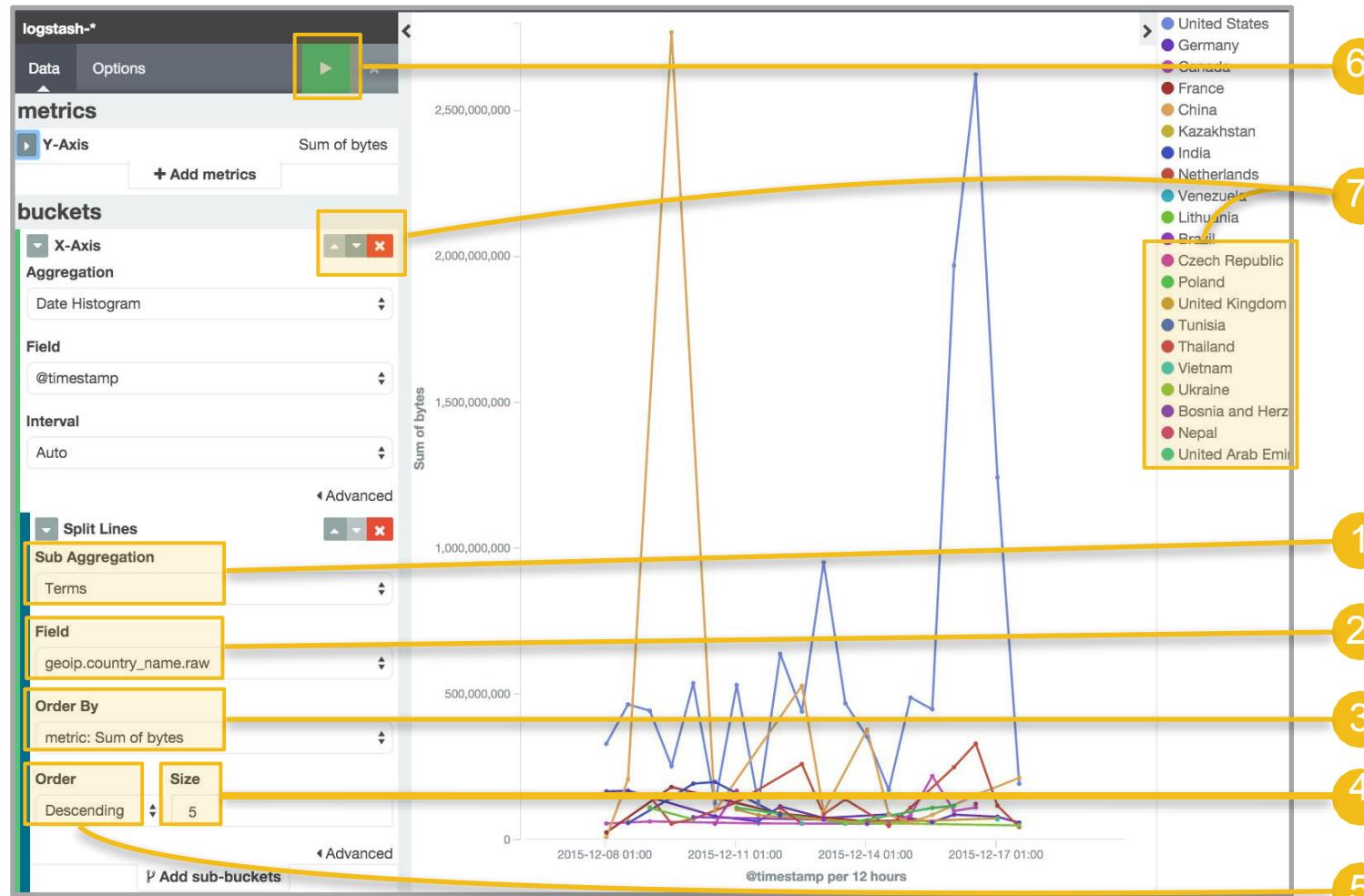
3 Set the bucket Interval to Auto.

4 Let's sub-divide each point.



Line chart

Bandwidth over Time by Country



6

Click apply.

7

Why do we have more than 5 countries?
Try clicking the down arrow (changes the aggregation execution order) and click apply again. And do not forget to save the visualization (e.g. **Top 5 Countries Bandwidth over Time**).

1

Select **Terms** aggregation.

2

On the field **geoip.country_name.raw**.
Ask an instructor why the **.raw** inner field!!

3

Order by the **Sum of bytes**.

4

Sub-divide each point into a maximum of 5 points.

5

Order in descending order.



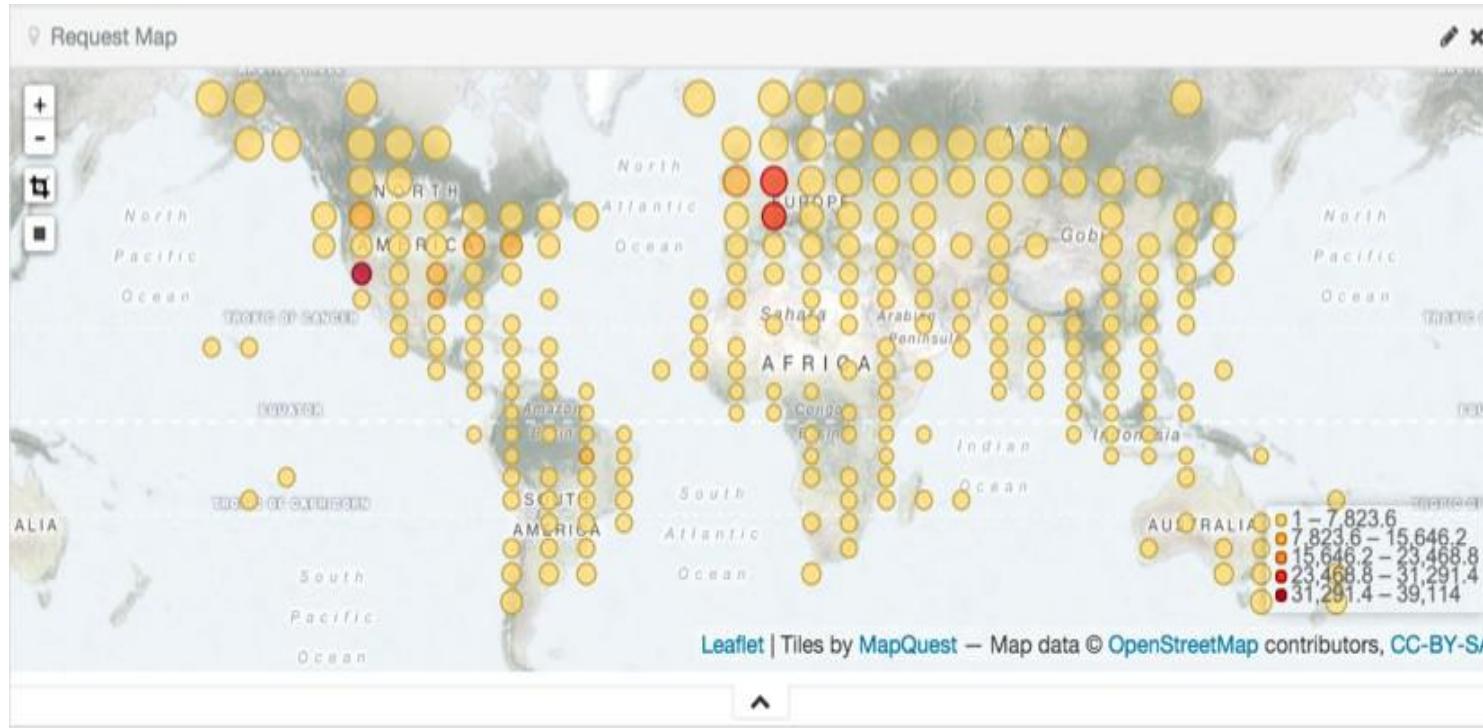
Tile Map

Create a visualization that uses geo-coordinates and a map

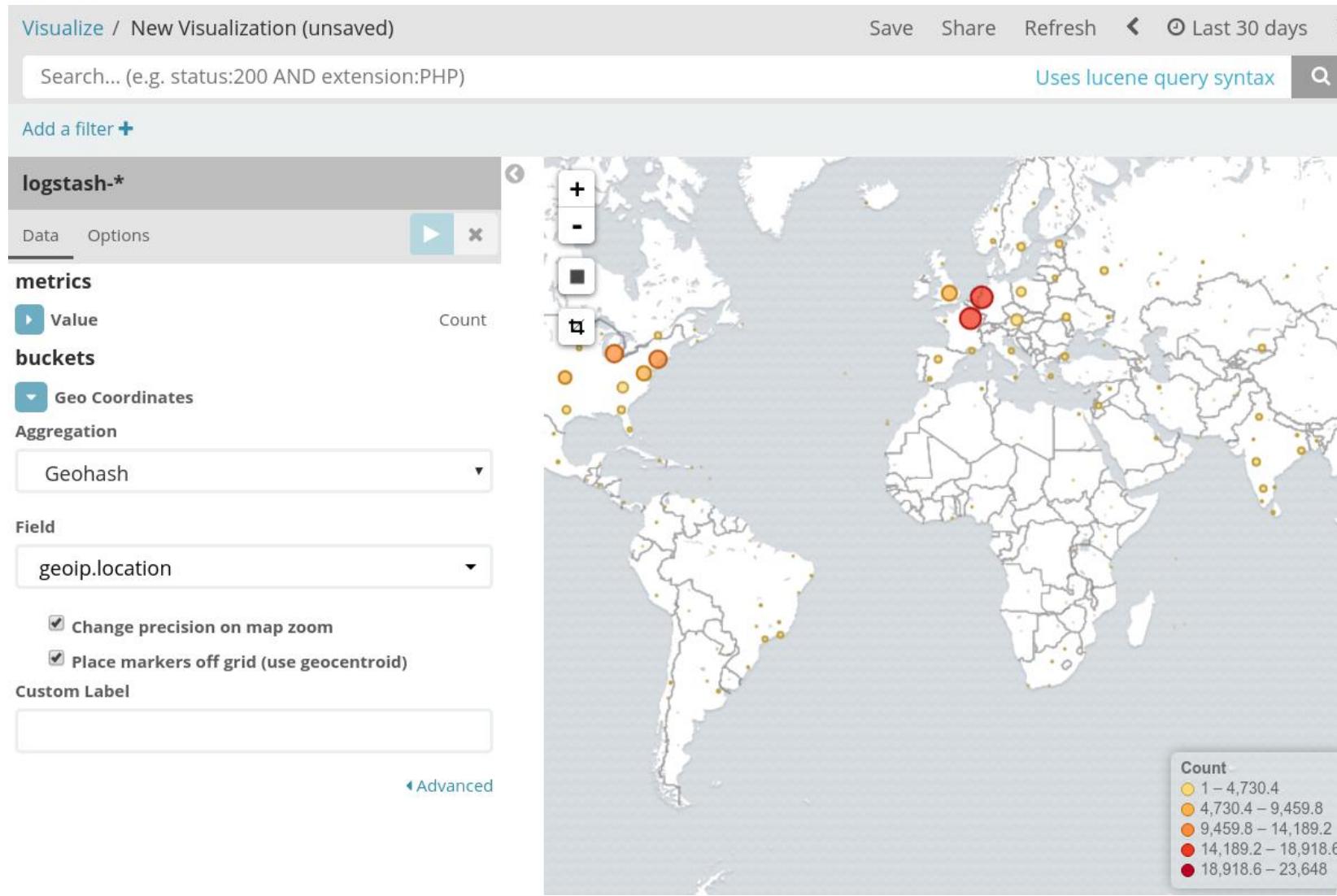
1. Create a "Co-ordinate Map" Visualization
2. Add "Geo Coordinates" based on the "geoip.location" field

In the "Options" select "Shaded Circle Markers"

Apply and save the visualization



The co-ordinate map visual



Putting it together in a dashboard

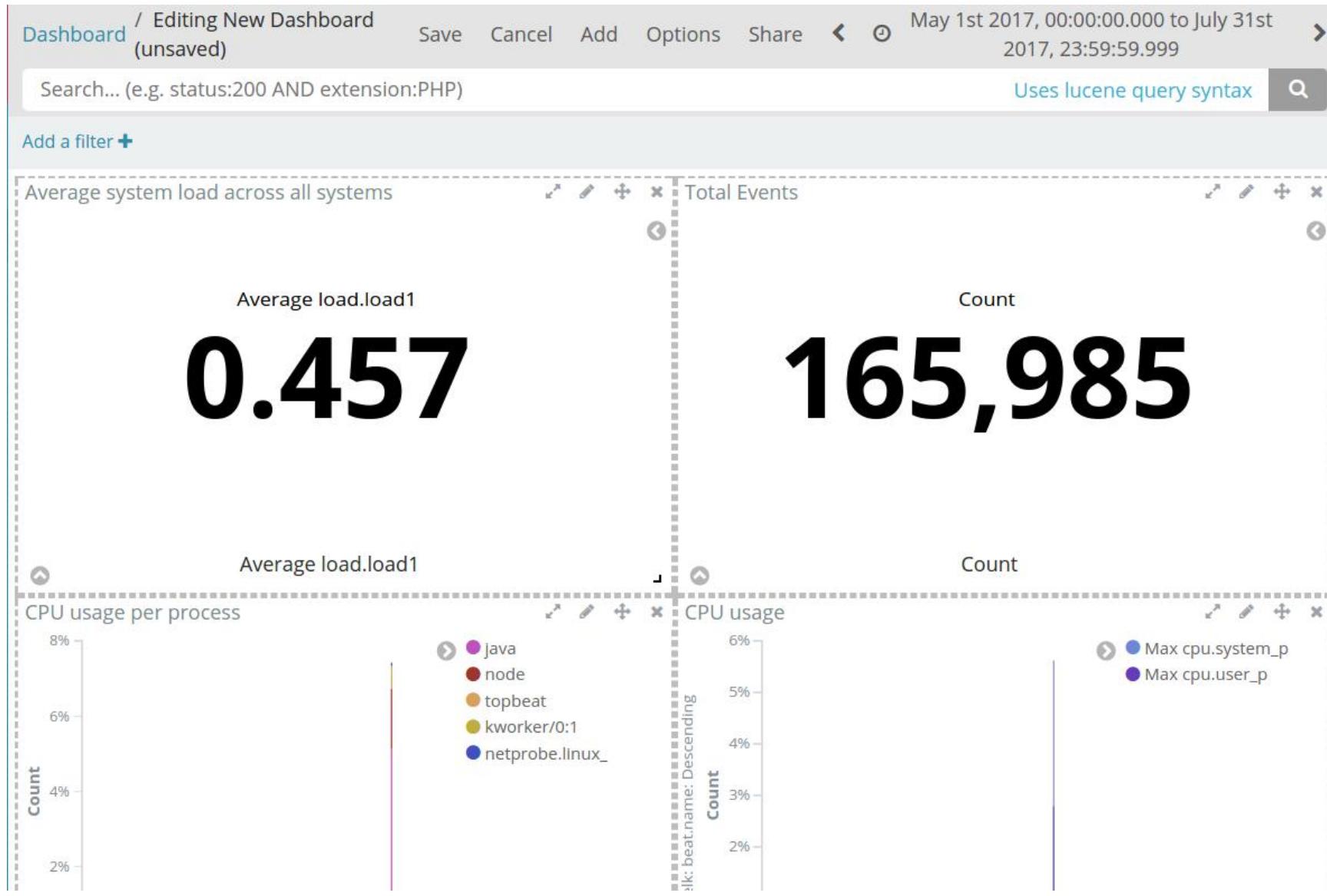
The screenshot shows the Kibana interface with the title "Dashboard". On the left is a sidebar with the Kibana logo and links: Discover, Visualize, **Dashboard**, Timelion, Dev Tools, and Management. The main area displays a list of existing dashboards: Name (HTTP, MongoDB performance, MySQL performance), Description, and a search bar. A red arrow points from the "Select existing dashboards" text to the list of dashboards. Another red arrow points from the "Create a new dashboard" text to the blue "+" button in the top right corner.

Select existing dashboards

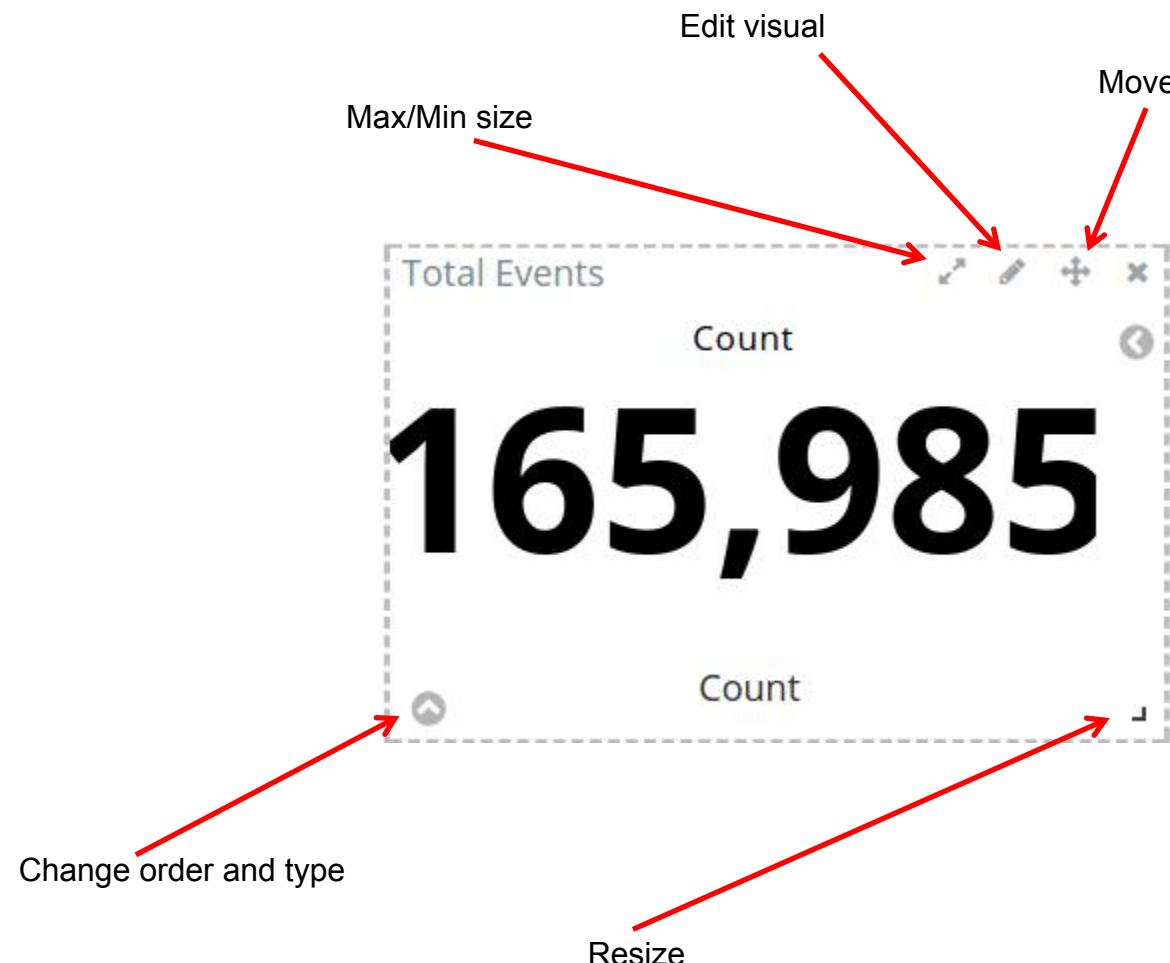
Create a new dashboard



Dashboard not looking good?



Drag and resize



Visualizations – Technical Challenges

Here's some good ideas on visualizations to make - try and solve them!

(Bar chart) Histogram on response codes with interval 100	(Line chart) Bytes Date Histogram	(Pie chart) Doughnut chart for User agents by device by country
(Data table) Top 10 IPs	(Data table) Top 10 Requests	(Line chart) Split Lines of country fields by response codes (exclude USA)



Congratulation

- You have completed Lab 7
- You now have a fully functional ELK stack
- You now have the ability to create visualizations

