

PILA ELK

Historial de revisiones

Fecha	Versión	Autor	Revisión
25/01/2018	1	Laurencio Galicia Angeles	Creación de documento.
02/02/2015	2	Laurencio Galicia Angeles	Instalación Logstash
20/02/2015	3	Laurencio Galicia Angeles	Instalación Filebeat y logstash en Docker
23/02/2015	4	Laurencio Galicia Angeles	Instalación Elasticsearch en Docker
2/03/2015	5	Laurencio Galicia Angeles	Instalación de Kibana en docker y creación de dashboards

ÍNDICE

Introducción	4
Objetivos generales del documento	4
Objetivos particulares	4
Pila EKL	5
Logstash	5
ElasticSearch	6
Kibana	7
Instalación de Logstash	8
Pre-requisito	8
Instalación	8
Logstash y Filebeat para la captura de Logs - Ejemplo de uso	9
Instalación de Filebeat	9
Configuración de Filebeat	9
Configuración de Logstash	10
Logstash en Docker.	12
Pre-requisito	12
Obtención de imagen	13
Montando contenedor	13
Filebeat en Docker.	14
Pre-requisito	14
Obtención de imagen	14
Montando contenedor	14
Elasticsearch en Docker	17
Pre-requisito	17
Obtención de imagen	17
Montando contenedor	17
Kibana en Docker	21
Pre-requisito	21
Obtención de imagen	21
Montando contenedor	21
Creación de dashboard	23
Discover	23
Visualize	24
Nota	26
Dashboard	27
Material extra	30
Componentes de la pantalla Discover	30
Metricbeat	30
Referencias	31

Introducción

Un *dashboard* es una representación gráfica de diferentes métricas o indicadores KPI (medidor de desempeño o indicador clave de rendimiento) utilizada generalmente para la obtención de los objetivos del negocio, o bien, el monitoreo de diferentes eventos dentro de un proceso.

En este documento se muestra la información a cerca de Kibana, herramienta de código abierto para la creación de *dashboards*, sin embargo, para hablar de kibana se necesita también conocer otros dos elementos: Elasticsearch y Logstash debido a su interacción para la recolección, almacenamiento y construcción del dashboard. Los tres elementos anteriores son conocidos como la pila ELK.

Objetivos generales del documento

Investigar la plataforma Kibana, Elastisearch y Logstash así como la interacción de estos.

Objetivos particulares

- Conocer la pila ELK.
- investigar de la forma de interactuar kibana con Elasticsearch y Logstash.
- Investigar las fuentes que pueden ser utilizadas para la recolección de información.
- Instalar Logstash

Pila EKL

La pila EKL se refiere a la unión e interacción de tres proyectos de código abierto: Elasticsearch, Logstash y Kibana, cada uno con una tarea diferente dentro del proceso de la recolección, análisis, almacenamiento y representación de la información en cuestión, por ejemplo la obtención de diferentes LOGS, indicadores de temperatura o de consultas de peticiones a servidores.

El proceso comienza con la recolección de los datos, tarea encargada por Logstash.

Logstash

Es un motor de recopilación de datos de código abierto con la capacidad de realizar una canalización en tiempo real (real-time pipelining). Logstash puede unificar dinámicamente datos de diferentes fuentes y formatos de representación para su normalización en el formato que se ha configurado.

Alguna de las siguientes características de este proyecto son las siguientes:

Registros y métricas

- Tubería de procesamiento de datos escalable horizontalmente con la interacción de Elasticsearch y Kibana.
- Permite la mezcla y combinación de diferentes entradas, filtros y salidas de datos.
- Cuenta con más de 200 complementos disponibles para mayor flexibilidad en el manejo de los datos.
- Ingesta fácilmente de una gran cantidad de registros web como Apache , y registros de aplicaciones como log4j para Java.
- Captura formatos de registro como syslog y registros de firewall, entre otros.
- Obtiene métricas de Ganglia, collectd, NetFlow, JMX y muchas otras plataformas de infraestructura y aplicaciones a través de TCP y UDP.

Web

- Transforma solicitudes HTTP en eventos.
- Compatible con Webhook para GitHub, HipChat, JIRA, entre otras.
- Captura el estado, el rendimiento, las métricas y otros tipos de datos de las interfaces de aplicaciones web.

Streams y datos almacenados

- Recolección de cualquier base de datos relacional o NoSQL con una interfaz JDBC
- Unifica mensajes de Apache Kafka , RabbitMQ y Amazon SQS de la cola de mensajes.

Logstash trabaja bajo dos elementos necesarios: entrada y salida, así como uno un elemento opcional: Filtro.

La entrada consume datos de una fuente, los filtros son plugins que modifican los datos obtenidos en las entrada respecto a la configuración hecha, y finalmente se tiene la salida, en ella los datos son escritos en un destino.

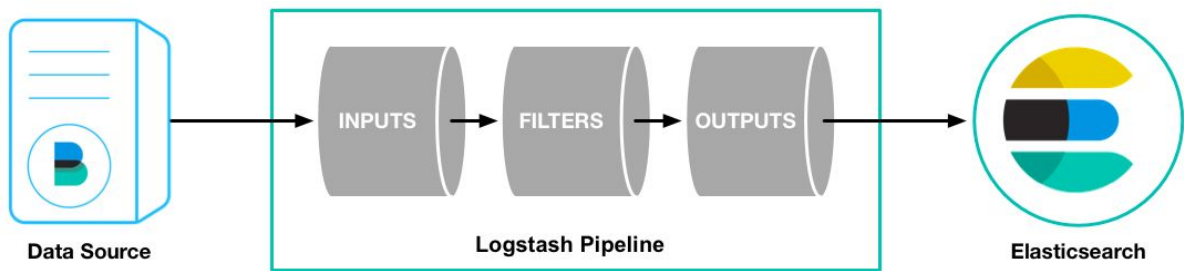


figura 1. Componentes de una tubería Logstash.

Imagen obtenida de: <https://www.elastic.co/guide/en/logstash/6.1/first-event.html>

Al resultado final obtenido del proceso de unificar los diferentes formatos y fuentes de datos se le conoce como **Stash**.

Una vez recolectada y unificada la información de interés, es necesaria almacenarla de una manera óptima para su próximas consultas. El responsable de esto es Elasticsearch.

ElasticSearch

Es un motor de búsqueda y análisis de texto, distribuido y escalable de código abierto. Elasticsearch permite almacenar, buscar y analizar grandes volúmenes de datos de forma rápida; generalmente se utiliza como tecnología subyacente para el impulso de aplicaciones que necesitan realizar búsquedas complejas.

Elasticsearch está desarrollado en Java y utiliza un DSL (lenguaje específico de dominio) basado en JSON para realizar *queries*.

Existen dos tipos de cláusulas para la definición de consultas: *Leaf query clauses* y *Compound query clauses*. Las primeras buscan un valor en particular en un campo particular, ejemplos de esto son las consultas:

- Match o de coincidencia.
- Term o de término.
- Range o de rango.

La diferencia entre *match* y *term* es que la consulta de coincidencia analiza la cadena de entrada y construye consultas más básicas a partir de esa, mientras que la consulta de término analiza la cadena con los términos exactos, por otro lado, la consulta de rango analiza cadenas dentro de un rango de valores establecidos.

Las *Compound query clauses* o cláusulas compuestas de consulta están integradas por otras consultas de hojas o compuestas; se utilizan para combinar múltiples consultas de manera lógica o para alterar su comportamiento. Ejemplo de lo anterior son las consultas *bool*: Consulta que coincide con documentos que coinciden con combinaciones booleanas de otras consultas y las consultas *constant_score*: consulta que ajusta otra consulta y devuelve un número correspondiente a cada documento filtrado.

Por último, Elasticsearch utiliza clusters y nodos para realizar una búsqueda en tiempo casi real (NRT), estos pueden ser configurados mejor convengan a las necesidades o condiciones que se tengan.

Después de recolectar y almacenar la información de interés se puede realizar una representación gráfica de lo obtenido, de esta tarea se encarga Kibana.

Kibana

Es una plataforma de código abierto para el análisis y visualización de datos almacenados en los índices de Elasticsearch, estos pueden ser mostrados en una variedad de *frames*, tablas y mapas.

En conclusión, la pila ELK es de gran ayuda para realizar análisis de información de diferentes fuentes y naturaleza, ofreciendo una herramienta para cada parte del proceso, comenzando con la recolección de información para después almacenarla y finalmente poder representarla de forma gráfica.



figura 2. Pila ELK

Imagen obtenida de: <https://www.elastic.co/guide/en/logstash/6.1/first-event.html>

Instalación de Logstash

Pre-requisito

Logstash requiere Java 8. Java 9 no es compatible.

Instalación

Se descarga y se instala la clave de firma pública

```
rpm --import https://artifacts.elastic.co/GPG-KEY-elasticsearch
```

Se crea un archivo en la ruta **/etc/yum.repos.d/** con la terminación **.repo**, por ejemplo: **logstash.repo** y se agrega lo siguiente:

```
[logstash-6.x]
name=Elastic repository for 6.x packages
baseurl=https://artifacts.elastic.co/packages/6.x/yum
gpgcheck=1
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
enabled=1
autorefresh=1
type=rpm-md
```

Y se instala

```
sudo yum install logstash
```

Para comprobar que la instalación ha sido correcta se debe ingresar al directorio ubicado en **/usr/share/logstash/bin**. Esta ruta depende de la plataforma instalada, para más información: <https://www.elastic.co/guide/en/logstash/6.1/first-event.html>

Se ejecuta lo siguiente

```
./logstash -e 'input { stdin { } } output { stdout { } }'
```

La tarea de la ejecución anterior es tomar como entrada lo escrito en el teclado y tan cual mostrarlo a la salida en pantalla, como se muestra a continuación.

```
[root@Lauren bin]# ./logstash -e 'input { stdin { } } output { stdout { } }'
WARNING: Could not find logstash.yml which is typically located in $LS_HOME/co
nfig or /etc/logstash. You can specify the path using --path.settings. Continu
ing using the defaults
Could not find log4j2 configuration at path /usr/share/logstash/config/log4j2.
properties. Using default config which logs errors to the console
The stdin plugin is now waiting for input:
Hello world
2018-02-02T01:54:31.597Z Lauren Hello world
```

La ejecución seguirá en la espera de algun texto en la entrada hasta su detención con el comando **CTRL-D**. Con lo anterior se comprueba que logstash se ha instalado correctamente.

Logstash y Filebeat para la captura de Logs - Ejemplo de uso

Una aplicación de logstash a un escenario más realista es la recolección de los Logs del equipo, para ello se utilizará a la par con Filebeat.

Filebeat es un proyecto de código abierto a cargo de Elastic, el cual supervisa los directorios de registro o archivos de registro específicos, los agrupa y reenvía a Elasticsearch o logstash.

Para este caso se enviarán los datos a logstash para realizar algunas modificaciones de formato con la información obtenida.

Logstash no cuenta con Filebat en su instalación, por tal motivo, se necesita realizar su instalación.

Instalación de Filebeat

Pre-requisito

Tener instalado rpm

Instalación

Para instalar Filebeat se deben ejecutar las siguientes dos líneas

```
curl -L -O
https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-6.1.3-x86_64.rpm

sudo rpm -vi filebeat-6.1.3-x86_64.rpm
```

Configuración de Filebeat

Una vez instalado, se debe modificar el archivo **filebeat.yml** ubicado en **etc/filebeat/**

En el se debe especificar la ruta o rutas (**paths**) donde se localizan los archivos o el archivo de registro para su supervisión, además de colocar como salida a Logstash con su puerto por defecto que es el **5044**

```
##### Filebeat: filebeat.yml #####
filebeat.prospectors:
- type: log
  enabled: true
  paths:
- /home/lauren/Descargas/logstash-tutorial.log/data.log # Ruta donde se encuentra
el archivo a supervisar
##### Filebeat modules #####
filebeat.config.modules:
# Glob pattern for configuration loading
path: ${path.config}/modules.d/*.yml
```



```

#===== Elasticsearch template setting =====
setup.template.settings:
  index.number_of_shards: 3
  #index.codec: best_compression
  #_source.enabled: false
#===== Outputs =====
# Configure what output to use when sending the data collected by the beat.
#----- Logstash output -----
output.logstash:
  # The Logstash hosts
  hosts: ["localhost:5044"]

  # Optional SSL. By default is off.
  # List of root certificates for HTTPS server verifications
  #ssl.certificate_authorities: ["/etc/pki/root/ca.pem"]

  # Certificate for SSL client authentication
  #ssl.certificate: "/etc/pki/client/cert.pem"

  # Client Certificate Key
  #ssl.key: "/etc/pki/client/cert.key"

```

Para el ejemplo no se ha especificado un SSL, certificado SSL y clave del certificado del cliente. El archivo **data.log** contiene algunos registros que serán enviados a logstash. Finalmente se ejecuta filebeat bajo la configuración anterior.

```
filebeat -e -c filebeat.yml -d "publish"
```

Se debe ejecutar en el directorio **/etc/filebeat**, lugar donde se encuentra el archivo **filebeat.yml**. Cabe resaltar que cada que se modifique este archivo se debe detener el proceso y volver a ejecutar filebeat con la instrucción anterior.

Configuración de Logstash

Lo siguiente es configurar logstash, para ello se crea un nuevo archivo con la extensión **.conf** para este ejemplo se nombró **first-pipeline.conf** y fue creado en el directorio **/usr/share/logstash/bin**, en él se indicarán la entrada y salida de esta tubería.

```

# ===== Logstash: first-pipeline.conf ===== #
input {
  beats {
    port => "5044"
  }
}
output {
  stdout { codec => rubydebug }
}

```

Como entrada se ha colocado un plug de Elastic Beats framework. (instalado por defecto en logstash), el cual recibe eventos de este framework, en este caso son creados por Filebeat y como salida un stdout para la impresión en pantalla de los eventos procesados por logstash.

Finalmente, se ejecuta la tubería

```
./logstash -f first-pipeline.conf
```

Es importante mencionar que si se modifica la el archivo **.conf** ejecutado se debe detener el proceso y volver a ejecutar, sin embargo, si se agrega la opción **--config.reload.automatic**, no es necesario detener y volver a ejecutar pues éste efectuará los cambios automáticamente.

```
./logstash -f first-pipeline.conf --config.reload.automatic
```

Una vez ejecutado Logstash y Filebeat se obtuvieron los siguientes resultados

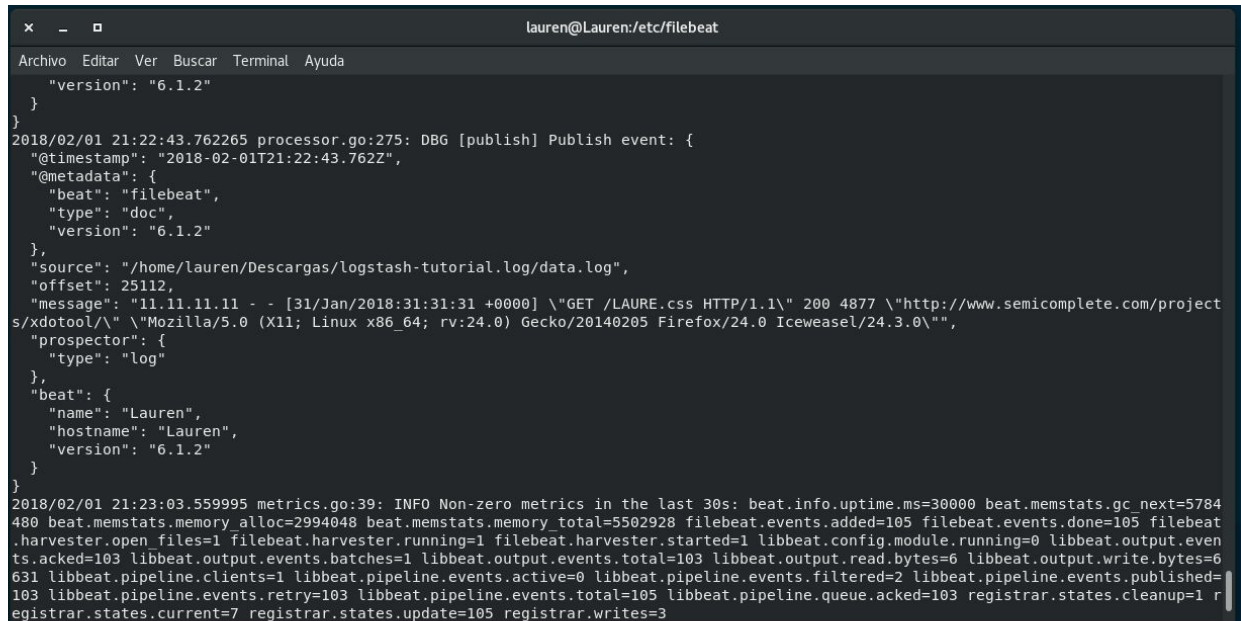
```
{
  "timestamp" => "04/Jan/2015:05:30:37 +0000",
  "beat" => {
    "version" => "6.1.2",
    "hostname" => "Lauren",
    "name" => "Lauren"
  },
  "verb" => "GET",
  "@timestamp" => 2018-02-01T21:36:15.957Z,
  "bytes" => "4877",
  "clientip" => "86.1.76.62",
  "message" => "86.1.76.62 - - [04/Jan/2015:05:30:37 +0000] \"GET /style2.css HTTP/1.1\" 200 4877 \"http://www.semicomplete.com/projects/xdotool/\" \"Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20140205 Firefox/24.0 Icedove/24.3.0\"",
  "prospector" => {
    "type" => "log"
  },
  "auth" => "-",
  "httpversion" => "1.1",
  "request" => "/style2.css",
  "@version" => "1",
  "host" => "Lauren",
  "tags" => [
    [0] "beats_input_codec_plain_applied"
  ],
  "ident" => "-",
  "agent" => "\"Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20140205 Firefox/24.0 Icedove/24.3.0\"",
  "referrer" => "\"http://www.semicomplete.com/projects/xdotool/\"",
  "offset" => 24464,
  "source" => "/home/lauren/Descargas/logstash-tutorial.log/data.log",
  "response" => "200"
}
```

pantalla anterior corresponde al resultado mostrado por Logstash en consola, en ella se

puede ver un registro del archivo data.log de manera estructurada y con los campos que lo componen identificados.

Esta información puede ser previamente re-estructurada o tratada con ayuda de un filtro, mientras que en la salida puede ser almacenada en una base de datos o enviada a otro proceso para su uso.

Por otro lado, en la consola donde se ejecutó Filebeat se muestra lo siguiente.



```

x - □ lauren@Lauren:/etc/filebeat
Archivo Editar Ver Buscar Terminal Ayuda
{"version": "6.1.2"}
}
2018/02/01 21:22:43.762265 processor.go:275: DBG [publish] Publish event: {
  "@timestamp": "2018-02-01T21:22:43.762Z",
  "@metadata": {
    "beat": "filebeat",
    "type": "doc",
    "version": "6.1.2"
  },
  "source": "/home/lauren/Descargas/logstash-tutorial.log/data.log",
  "offset": 25112,
  "message": "11.11.11.11 - - [31/Jan/2018:31:31:31 +0000] \"GET /LAURE.css HTTP/1.1\" 200 4877 \"http://www.semicomplete.com/project-s/xdotool/\" \"Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20140205 Firefox/24.0 Iceweasel/24.3.0\"",
  "prospector": {
    "type": "log"
  },
  "beat": {
    "name": "Lauren",
    "hostname": "Lauren",
    "version": "6.1.2"
  }
}
2018/02/01 21:23:03.559995 metrics.go:39: INFO Non-zero metrics in the last 30s: beat.info.uptime.ms=30000 beat.memstats.gc_next=5784
480 beat.memstats.memory_alloc=2994048 beat.memstats.memory_total=5502928 filebeat.events.added=105 filebeat.events.done=105 filebeat
.harvester.open_files=1 filebeat.harvester.running=1 filebeat.harvester.started=1 libbeat.config.module.running=0 libbeat.output.even
ts.acked=103 libbeat.output.events.batches=1 libbeat.output.events.total=103 libbeat.output.read.bytes=6 libbeat.output.write.bytes=6
631 libbeat.pipeline.clients=1 libbeat.pipeline.events.active=0 libbeat.pipeline.events.filtered=2 libbeat.pipeline.events.published=
103 libbeat.pipeline.events.retry=103 libbeat.pipeline.events.total=105 libbeat.pipeline.queue.acked=103 registrar.states.cleanup=1 r
egistrar.states.current=7 registrar.states.update=105 registrar.writes=3

```

En la imagen anterior se muestran los eventos publicados por filebeat así como el mensaje de inactividad en la creación de eventos, es decir, no se modificó el archivo de registro que supervisa.

Logstash en Docker.

A continuación se mostrará la instalación de Logstash como un contenedor para Docker, antes de comenzar es sano recordar que Docker es una herramienta de código abierto para facilitar la creación, implementación y ejecución de aplicaciones mediante el uso de contenedores.

Dentro de los contenedores se encuentran todos los requisitos o pre-requisitos, como bibliotecas, dependencias u otras aplicaciones, necesarias para la correcta ejecución de la aplicación, es decir, dentro del contenedor de una aplicación, no solo se encuentra la herramienta que se desea utilizar, sino también todos los elementos necesarios para su funcionamiento. Lo anterior permite ejecutar la aplicación en diferentes plataformas, independientemente de la configuración que pueda tener la máquina.

Pre-requisito

Contar con Docker instalado

Obtención de imagen.

Para poder utilizar un contenedor, es necesario contar con su imagen, es por ello que el primer paso es obtenerla.

Cabe resaltar que para poder ejecutar comandos de Docker es necesario tener permisos de superusuario.

La imagen de logstash se obtiene utilizando la siguiente sentencia

```
docker pull docker.elastic.co/logstash/logstash:6.2.1
```

Para comprobar que la imagen se ha descargado, se ejecuta **docker images** en consola para obtener el listado de imágenes almacenadas en el quipo, como se muestra en la figura siguiente.

```
[laurencio.galicia@amxctingrl01 ~]$ sudo su
[root@amxctingrl01 laurencio.galicia]# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
<none>	<none>	69888f6341c5	4 days ago	378 MB
<none>	<none>	5d813797cea1	4 days ago	378 MB
docker.elastic.co/logstash/logstash-oss	6.2.1	7f2488816fbd	11 days ago	702.1 MB
docker.elastic.co/beats/filebeat	6.2.1	26b5e1828fad	12 days ago	378 MB
docker.io/graylog/graylog	2.4.3-1	7d2f93c37e8f	3 weeks ago	696.4 MB
docker.io/mongo	3	a0c8a8d1f8ec	8 weeks ago	365.5 MB
docker.io/graylog/graylog	2.3.0-1	4df6dfdf78c1	6 months ago	412 MB
docker.elastic.co/elasticsearch/elasticsearch	5.5.1	74ef44f69db6	6 months ago	544.2 MB

```
[root@amxctingrl01 laurencio.galicia]# Connection to 201.161.87.8 closed by remote host
```

Montando contenedor

Lo siguiente es montar el contenedor en el equipo, esto se hace con la siguiente línea.

```
docker run --rm -it -v
/var/containers/logstash/pipeline/:/usr/share/logstash/pipeline/bin
docker.elastic.co/logstash/logstash-oss:6.2.1
```

La línea anterior está compuesta por diferentes banderas, las cuales realizan lo siguiente:

- run: Ejecuta un comando en un contenedor nuevo, este tiene el siguiente formato
docker run [OPTIONS] IMAGE[:TAG|@DIGEST] [COMMAND] [ARG...]

Las opciones que se utilizan son las siguientes

- --rm: Elimina el contenedor si existe o elimina el demonio si es que existe.
- -it: Indica a Docker que asigne un pseudo-TTY conectado al stdin del contenedor (pseudoterminal)
- -v: Monta el directorio de trabajo actual en el contenedor, en este caso, en el directorio **/var/containers/logstash/pipeline/**¹ se encuentran los archivos de configuración para la aplicación y estos se montan en el directorio de configuración de logstash dentro del contenedor que es **/usr/share/logstash/pipeline/bin**

La imagen que se utiliza es **docker.elastic.co/logstash/logstash-oss:6.2.1**

Las herramientas que hasta ahora se han utilizado necesitan ser configuradas conforme a las necesidades del proyecto, por tal razón, es necesario montar sobre el contenedor de la aplicación los archivos correspondientes, para ello, se debe consultar la documentación de la herramienta con el fin de conocer los archivos o archivo que se necesitan modificar así como la dirección de estos.

¹ Véase [nota](#)

Nota:

Los archivos de configuración que se crearán para después montarlos en el contenedor se deben almacenar en **/var/containers/[Aplicación]**, este conjunto de directorios debe crearse en caso de no existir; para este caso la aplicación es logstash, con base a lo anterior, la dirección para alojar los archivos de configuración de la aplicación es **/var/containers/logstash**, sin embargo, dentro de logstash se creará un directorio llamado pipeline, el cual tiene el archivo de configuración llamado **pipeline.conf**

```
[laurencio.galicia@amxctingrl01 pipeline]$ pwd
/var/containers/logstash/pipeline
[laurencio.galicia@amxctingrl01 pipeline]$ ls
pipeline.conf
```

El archivo **pipeline.conf** contiene lo siguiente

```
# ===== Logstash: pipeline.conf ===== #
input{
  beats{
    port => "5044"
  }
}
output{
  stdout{ codec => rubydebug}
}
```

Con la configuración anterior logstash a través del puerto 5044 recibirá beats y estos serán impresos en pantalla como salida. Los eventos de beats serán creados por filebeat.

Filebeat en Docker.

Pre-requisito

Contar con Docker instalado

Obtención de imagen

La imagen de Filebeat se obtiene a través de la siguiente línea

```
docker pull docker.elastic.co/beats/filebeat:6.2.2
```

Montando contenedor

Lo siguiente es montar el contenedor así como los archivos de configuración.

```
docker run -v
/var/containers/logstash/filebeat/filebeat.yml:/usr/share/filebeat/filebeat.yml -v
/var/containers/logstash/filebeat/prospectors.d:/usr/share/filebeat/prospectors.d
-v /var/containers/logstash/filebeat/data.log:/var/log/data.log
docker.elastic.co/beats/filebeat:6.2.1
```

Las líneas anteriores montan el contenedor de filebeat a partir de la su imagen **docker.elastic.co/beats/filebeat:6.2.1**. Las opciones que utiliza es -v para montar los archivos del equipo en el contenedor de la imagen, en este caso se realizaràn 3 copias:

- /var/containers/logstash/filebeat/filebeat.yml a /usr/share/filebeat/filebeat.yml

```
# ===== Filebeat: filebeat.yml ===== #
filebeat.config:
  prospectors:
    path: ${path.config}/prospectors.d/*.yaml
    reload.enabled: false
  modules:
    path: ${path.config}/modules.d/*.yaml
    reload.enabled: false

processors:
- add_cloud_metadata:

output.logstash:
  hosts: ['172.17.0.3:5044'] #ip de logstash
```

filebeat.yml contiene la ruta de los archivos que se encargará de supervisar, los módulos así como la salida, que en este caso es enviar eventos beats hacia logstash. Cabe mencionar que la comunicación entre contenedores en docker es a través de direcciones IP y puertos, debido a que docker crea una red privada , siendo sus contenedores sus hosts. Para conocer los detalles de cada contenedor se utiliza **docker inspect [ID o Nombre del contenedor]**

```
[root@amxctingrl01 laurencio.galicia]# docker inspect logstash
```

El comando anterior despliega la información detallada del contenedor, entre la cual se encuentra la IP que le corresponde a logstash.

- /var/containers/logstash/filebeat/prospectors.d a /usr/share/filebeat/prospectors.d

Dentro de este directorio se encuentra el archivo **prueba.yml**

```
# ===== Filebeat: prueba.yml ===== #
- type: log
  paths:
    - /var/log/data.log
```

En el archivo se especifica que se supervisaràn logs que se encuentran en la ruta **/var/log/data.log**

- /var/containers/logstash/filebeat/data.log a /var/log/data.log

data.log contiene el registro de algunos eventos, este se utiliza con el fin de procesar su contenido.

Finalmente, con el contenedor de filebeat y logstash en ejecución se obtiene lo mostrado en las siguientes figuras

```
[root@amxctingrl01 laurencio.galicia]# docker run --name=filebeat -v
/var/containers/logstash/filebeat/filebeat.yml:/usr/share/filebeat/filebeat.yml -v
/var/containers/logstash/filebeat/prospectors.d:/usr/share/filebeat/prospectors.d
-v /var/containers/logstash/filebeat/data.log:/var/log/data.log
docker.elastic.co/beats/filebeat:6.2.1

2018-02-20T22:29:56.921Z      INFO      instance/beat.go:468      Home path:
[/usr/share/filebeat] Config path: [/usr/share/filebeat] Data path:
[/usr/share/filebeat/data] Logs path: [/usr/share/filebeat/logs]
2018-02-20T22:29:56.922Z      INFO      instance/beat.go:475      Beat UUID:
4f5f83ce-3ac1-4571-bdad-4188b840d29b
2018-02-20T22:29:56.922Z      INFO      instance/beat.go:213      Setup Beat: filebeat;
Version: 6.2.1
2018-02-20T22:29:59.923Z      INFO      add_cloud_metadata/add_cloud_metadata.go:297
add_cloud_metadata: hosting provider type not detected.
2018-02-20T22:29:59.923Z      INFO      pipeline/module.go:76      Beat name:
31d81bb86d8e
```

** Extracto de lo mostrado en terminal al montar el contenedor de filebeat*

```
[root@amxctingrl01 laurencio.galicia]# docker run --name=logstash --rm -it -v
/var/containers/logstash/pipeline:/usr/share/logstash/pipeline/bin
docker.elastic.co/logstash/logstash-oss:6.2.1

Sending Logstash's logs to /usr/share/logstash/logs which is now configured via
log4j2.properties
[2018-02-20T22:28:53,102][INFO ][logstash.modules.scaffold] Initializing module
{:module_name=>"fb_apache",
:directory=>"/usr/share/logstash/modules/fb_apache/configuration"}
[2018-02-20T22:28:53,134][INFO ][logstash.modules.scaffold] Initializing module
{:module_name=>"netflow",
:directory=>"/usr/share/logstash/modules/netflow/configuration"}
[2018-02-20T22:28:53,269][INFO ][logstash.setting.writabledirectory] Creating
directory {:setting=>"path.queue", :path=>"/usr/share/logstash/data/queue"}
[2018-02-20T22:28:53,283][INFO ][logstash.setting.writabledirectory] Creating
directory {:setting=>"path.dead_letter_queue",
:path=>"/usr/share/logstash/data/dead_letter_queue"}
[2018-02-20T22:28:54,129][INFO ][logstash.agent] No persistent UUID
file found. Generating new UUID {:uuid=>"b99b5fffd-6212-47b5-9a24-03f3acffba38",
:path=>"/usr/share/logstash/data/uuid"}
[2018-02-20T22:28:55,087][INFO ][logstash.runner] Starting Logstash
{"logstash.version"=>"6.2.1"}
```

** Extracto de lo mostrado en terminal al montar el contenedor de logstash*

```
{
  "host" => "31d81bb86d8e",
  "@version" => "1",
  "prospector" => {
    "type" => "log"
  },
  "tags" => [
    [0] "beats_input_codec_plain_applied"
  ],
  "offset" => 25114,
  "message" => "11.11.11.12; - - [31/Jan/2018:31:31:31 +0000] \"GET /LAURE.css
HTTP/1.1\" 200 4877 \"http://www.semicomplete.com/projects/xdotool/\" \"Mozilla/5.0
(X11; Linux x86_64; rv:24.0) Gecko/20140205 Firefox/24.0 Iceweasel/24.3.0\" ",
  "source" => "/var/log/data.log",
  "beat" => {
    "version" => "6.2.1",
    "hostname" => "31d81bb86d8e",
    "name" => "31d81bb86d8e"
  },
  "@timestamp" => 2018-02-20T22:29:59.968Z
}
```

** Extracto de lo mostrado en terminal por parte de logstash al recibir los beats creados por filebeat*

Una vez recolectada la información, el siguiente paso es almacenarla, para ello, se utilizará Elasticsearch, otro elemento de la pila ELK.

Elasticsearch en Docker

Elasticsearch permite almacenar, buscar y analizar grandes volúmenes de datos de forma rápida debido a su capacidad de escalabilidad y distribución, haciendo uso de nodos y clusters.

Pre-requisito

Contar con Docker instalado

Obtención de imagen

```
docker pull docker.elastic.co/elasticsearch/elasticsearch-oss:6.2.2
```

Montando contenedor

Una vez obtenida la imagen y después de haber comprobado que se encuentra almacenada en el equipo se realiza el montaje del contenedor.


```
docker run --name=elasticsearch -p 9200:9200 -p 9300:9300 -d -e
"discovery.type=single-node" -v
/var/containers/elk/elasticsearch/elasticsearch.yml:/usr/share/elasticsearch/config
/elasticsearch.yml:z docker.elastic.co/elasticsearch/elasticsearch-oss:6.2.2
```

Las opciones que componen al comando **run** son las siguientes:

- -p: Vincula el puerto del contenedor con el puerto del host, de esta manera, el contenedor será alcanzable por cualquiera que llegue a este puerto del host. El formato utilizado para esta bandera es **Puerto_del_Host:Puerto_del_Contenedor**, para este caso, el puerto del host y contenedor es el mismo: 9200
- -d: Ejecutara el contenedor en segundo plano, devuelve el ID del contenedor
- -e: Establece un valor a la variable de entorno, en este caso la variable de entorno es **discovery.type** y el valor es **single-node**. La asignación anterior significa que un nodo se elegirá a sí mismo maestro y no se unirá a un clúster con ningún otro nodo.
- -v: Monta el archivo de configuración **elasticsearch.yml** almacenado de manera local en **/var/containers/elk/elasticsearch/** en el contenedor, en la ubicación correspondiente para el archivo de configuración con el mismo nombre pero ubicado en **/usr/share/elasticsearch/config**

```
# ===== ELASTICSEARCH: elasticsearch.yml ===== #
cluster.name: "docker-cluster"
network.host: 0.0.0.0

# minimum_master_nodes need to be explicitly set when bound on a public IP
# set to 1 to allow single node clusters
# Details: https://github.com/elastic/elasticsearch/pull/17288
discovery.zen.minimum_master_nodes: 1
```

La configuración que se ha realizado, por el momento es la de defecto.

Hecho lo anterior, se realiza la siguiente petición

```
curl http://127.0.0.1:9200/_cat/health
```

Dependiendo del color que regrese la petición anterior es el estado del cluster.

- **Green:** Todo está bien, el cluster es completamente funcional
- **Yellow:** Todos los datos están disponibles, pero algunas réplicas aún no están asignadas. El cluster es completamente funcional
- **Red:** Algunos datos no están disponibles por razones diversas. El cluster es parcialmente funcional.

Es importante aclarar que Elasticsearch permite dividir la información almacenada en múltiples piezas, esto recibe el nombre de réplicas (*shards*), por otro lado, Elasticsearch tiene la capacidad de realizar copias de estos fragmentos a los cuales llama replicas.

Para este caso, el resultado obtenido después de realizar esta petición fue la siguiente

```
[laurencio.galicia@amxctinelk ~]$ curl http://127.0.0.1:9200/_cat/health
1519414171 19:29:31 docker-cluster yellow 1 1 5 5 0 0 5 0 - 50.0%
```

Hasta el momento, Filebeat se comunica con Logstash por medio de Beats con el fin de procesar un archivo de registros, finalmente, el resultado es impreso en pantalla por Logstash, sin embargo, lo deseado es almacenar esta información en Elasticsearch, por tal motivo se deben realizar modificaciones al archivo de configuración de logstash, específicamente en la sección de salida (*output*) del archivo ***pipeline.yml***

```
# ===== Logstash: pipeline.yml ===== #
input{
  beats{
    port => "5044"
  }
}
output{
  elasticsearch {
    hosts => ['172.17.0.2:9200'] #IP y puerto del contenedor de Elasticsearch
    index => "logstash-%{+YYYY.MM.dd}" #Nombre del indice
  }
}
```

En esta configuración la salida se enviará a Elasticsearch, se ha colocado la IP correspondiente a su contenedor y puerto, además se estableció el nombre del índice.

Un índice es una colección de documentos que tienen alguna característica similar, estos son identificados por un nombre que **debe** ser escrito en minúsculas, en ellos se pueden agregar, buscar, modificar y eliminar documentos. Por otro lado, un documento es la unidad básica de información que puede ser agregada a un índice.

Modificado lo anterior, se detiene el contenedor activo de logstash y se vuelve a iniciar con la configuración actual.

```
docker run --name=logstash -it -d -v
/var/containers/elk/logstash/pipeline/:/usr/share/logstash/pipeline:z
docker.elastic.co/logstash/logstash-oss:6.2.2
```

Con lo anterior, se espera un índice con la información proporcionada por Logstash, para verificar lo anterior se ejecuta la siguiente petición con el fin de listar todos los índices del cluster.

```
curl -XGET 'localhost:9200/_cat/indices?v&pretty'
```

El resultado fue el siguiente

```
[laurencio.galicia@amxctinelk ~]$ curl -XGET 'localhost:9200/_cat/indices?v&pretty'
```

health	status	index	uuid	pri	rep	docs.count	docs.deleted	store.size	pri.store.size
yellow	open	logstash-2018.02.22	LQNcKNnoQpe2N2T5NRcw A	5	1	103	0	79.9kb	79.9kb

Como se muestra en la imagen anterior, se creó un índice llamado **logstash-2018.02.22** que cuenta con 103 documentos y tiene un tamaño de 79.9kb

Para revisar el contenido se realizó la siguiente consulta para listar todos los documentos del índice

```
curl -XGET 'localhost:9200/logstash-2018.02.22/_search?pretty' -H 'Content-Type: application/json' -d'
{
  "query": { "match_all": {} }
}
```

El formato para realizar las consultas está hecha en QueryDSL basado en JSON.

La consulta realiza una búsqueda (**_search**) en el índice **logstash-2018.02.22** de todos los documentos (**"query": { "match_all": {} }**) almacenados el cluster (**localhost**) por el puerto **9200**. '**?pretty**' se coloca con el fin de presentar de mejor manera la información en consola.

```
{
  "took" : 4,
  "timed_out" : false,
  "_shards" : {
    "total" : 5,
    "successful" : 5,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : 103,
    "max_score" : 1.0,
    "hits" : [
      {
        "_index" : "logstash-2018.02.22",
        "_type" : "doc",
        "_id" : "99Qxv2EBCSqtSbtQzPd0",
        "_score" : 1.0,
        "_source" : {
          "source" : "/var/log/data.log",
          "offset" : 5518,
          "tags" : [
            "beats_input_codec_plain_applied"
          ]
        }
      }
    ]
  },
  "@version" : "1",
}
```

```

    "message" : "83.149.9.216 - - [04/Jan/2015:05:13:47 +0000] \"GET
/presentations/logstash-monitorama-2013/images/logstashbook.png HTTP/1.1\" 200
54662 \"http://semicomplete.com/presentations/logstash-monitorama-2013/\"
\"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_1) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/32.0.1700.77 Safari/537.36\"",
    "@timestamp" : "2018-02-22T20:26:02.891Z",
    "beat" : {
        "hostname" : "11f0c9f93813",
        "name" : "11f0c9f93813",
        "version" : "6.2.2"
    },
    "host" : "11f0c9f93813",
    "prospector" : {
        "type" : "log"
    }
}
},

```

** Extracto de lo mostrado en terminal al ejecutar la consulta*

Finalmente con los datos almacenados, se puede implementar el ultimo elemento de la pila ELK: Kibana.

Kibana en Docker

Pre-requisito

Contar con Docker instalado

Obtención de imagen

```
docker pull docker.elastic.co/kibana/kibana-oss:6.2.2
```

Montando contenedor

Una vez obtenida la imagen y después de verificar que se encuentra almacenada en el equipo se realiza el montaje del contenedor.

```

docker run --name=kibana -p 5601:5601 -d -v
/var/containers/elk/kibana/kibana.yml:/usr/share/kibana/config/kibana.yml:z
docker.elastic.co/kibana/kibana-oss:6.2.2

```

Las opciones utilizadas por **run** son las siguientes:

- **--name**: Asigna nombre al contenedor
- **-p**: Expone y Vincula el puerto del contenedor con el puerto del host. Formato **-p Puerto_Host:Puerto_Contenedor**. En este caso es el puerto **5601**
- **-v**: Toma el archivo de configuración **kibana.yml** almacenado en **/var/containers/elk/kibana/** en el equipo y lo monta en **/usr/share/kibana/config/** dentro del contenedor.

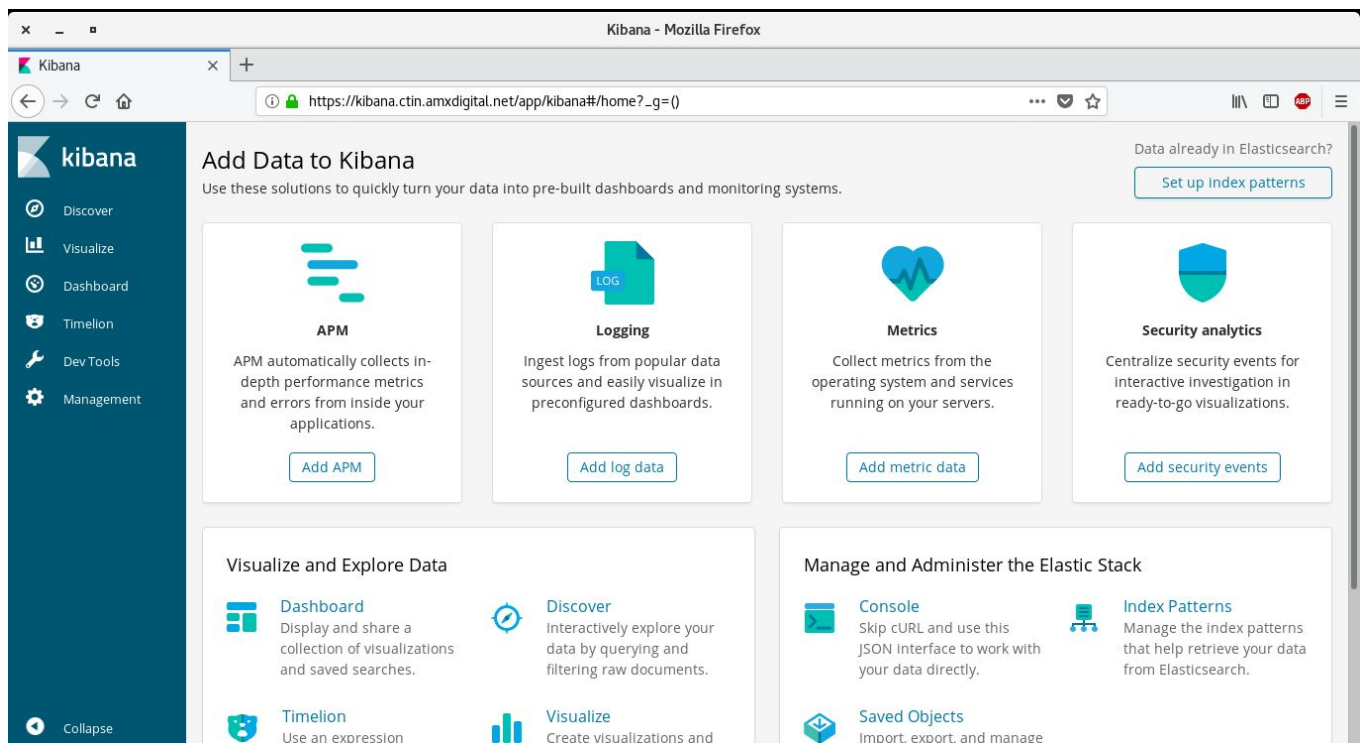
Todo esto lo ejecuta tomando la imagen **docker.elastic.co/kibana/kibana-oss:6.2.2** correspondiente a kiabana.

El archivo de configuración kibana.yml contiene lo siguiente

```
# ===== kibana: kibana.yml ===== #
#kibana configuration from kibana-docker.
server.name: kibana
server.host: "0"
elasticsearch.url: http://172.17.0.2:9200 #Dirección IP del contenedor de
elasticsearch
```

Para visualizar el sitio web de **kibana** de manera local basta con ingresar a **http://localhost:5601**, sin embargo, para poder ingresar y visualizar a la página web desde cualquier lugar, se asignó un dominio y permisos para poder ingresar al puerto 5601 en el servidor, de tal manera, la URL de kibana quedó de la siguiente manera: **https://kibana.ctin.amxdigital.net/**

La pantalla de inicio de kibana es la siguiente.



De lado izquierdo se encuentra un barra lateral con las opciones:

- **Discover:** En esta sección se pueden visualizar los índices y documentos de elasticsearch, además de crear consultas de búsqueda, filtrar datos y listar a detalle cada documento del índice seleccionado.

AMX/CTIN INFRAESTRUCTURA 2017

- **Visualize:** Aquí se crean visualizaciones de los datos del índice seleccionado, la extracción de datos a representar se realiza mediante consultas DSL. Estas visualizaciones también pueden ser creadas desde *Discover*
- **Dashboard:** En esta sección se crean tableros con que muestran una colección de visualizaciones guardadas.
- **Timelion:** Es una herramienta que permite visualizar datos de series temporales, permite agregar datos de fuentes diferentes dentro de una única gráfica.
- **Dev-Tools:** Contiene una consola con el fin de poder interactuar con los datos almacenados en los índices de elasticsearch.
- **Management:** Aquí se realizan configuraciones a Kibana, visualizaciones, dashboards e índices.

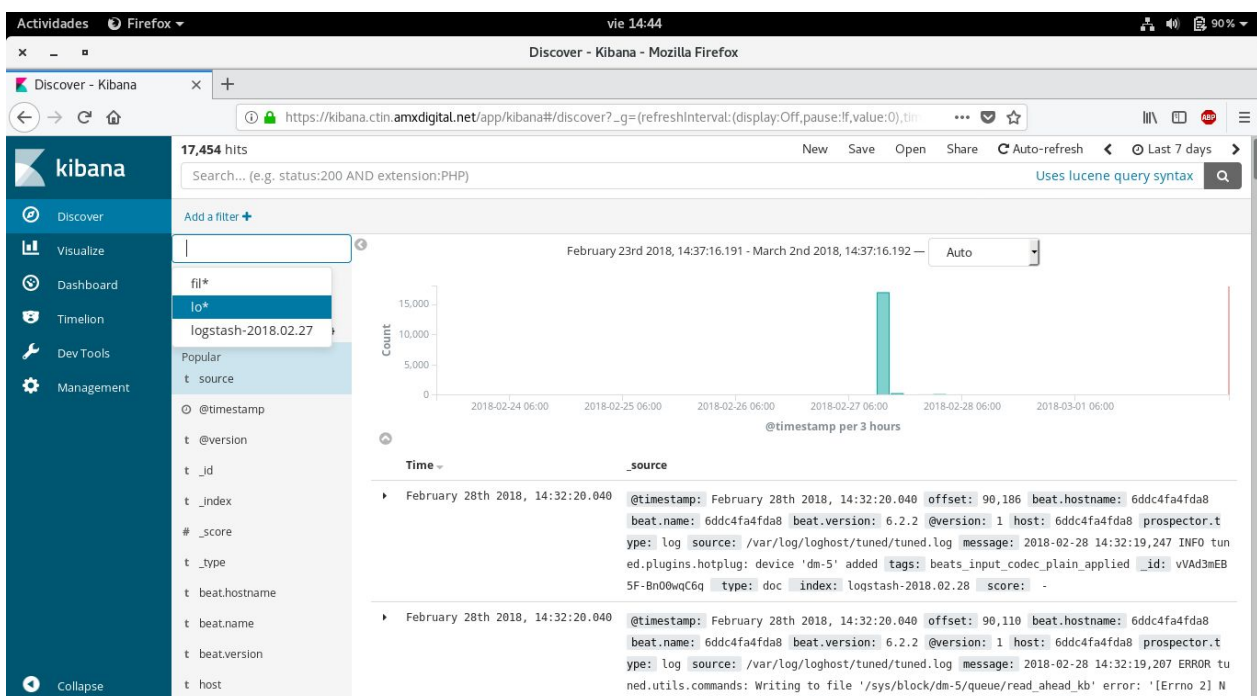
Creación de dashboard

Discover

En la sección *Discover* se pueden visualizar los patrones de índices o índices almacenados en elasticsearch, cuando se selecciona uno se muestran a detalle los documentos que lo componen así como los campos disponibles en la información para su manipulación, como se muestra en la siguiente imagen, se tienen 3 índices: **fil***, **lo*** y **logstash-2018.02.27**, al elegir alguno realiza una gráfica automática, es importante mencionar que esta gráfica depende del tiempo que se haya seleccionado en el filtro de tiempo, este se ubica en la barra superior, para esta visualización se seleccionó la opción **last 7 days**.

Aquí se realiza una manipulación de datos de manera interactiva y con la opción de realizar una visualización a partir de los campos seleccionados, sin embargo, el filtrado de datos es un tanto limitado debido a las opciones definidas.

[Para más información de la pantalla *discover* ver la sección de material extra.](#)



Visualize

Por tal motivo, la mejor es crear una visualización desde 0 en la sección de *visualize*, al abrir esta sección se muestran las visualizaciones guardadas en caso de que existan, como se muestra en la siguiente imagen, para crear una, se da clic sobre el botón +

Visualize

Q Search...	+	1-6 of 6
<input type="checkbox"/> Title ↑	Type	
<input type="checkbox"/> FirewallId	Timelion	
<input type="checkbox"/> Log: /var/log	Vertical Bar	
<input type="checkbox"/> Visualize: CountAllFirewallId	42 Metric	
<input type="checkbox"/> Visualize: CountTry	42 Metric	
<input type="checkbox"/> Visualize: CountWarnigs	42 Metric	
<input type="checkbox"/> Warnig FirewallId	Vertical Bar	
0 Items selected		1-6 of 6

La pantalla que se muestra al dar clic sobre + es un menú con los diferentes tipos de visualizaciones, como se muestra en la siguiente imagen. Para esta visualización se seleccionó la gráfica de barras vertical o *vertical Bar*.

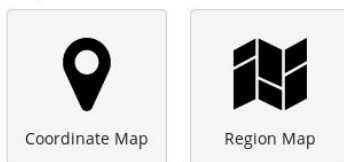
Basic Charts



Data



Maps



Una vez hecho lo anterior, se debe seleccionar el índice o alguna búsqueda guardada anteriormente para la representación de datos de forma gráfica. Se seleccionó el índice **lo***, como se muestra en la siguiente imagen.

AMX/CTIN INFRAESTRUCTURA 2017

From a New Search, Select Index

Filter...

3 of 3

Name ▲

fil*

logstash-2018.02.27

lo*

Or, From a Saved Search

Saved Searches Filter...

1-1 of 1

Manage saved searches

Name ▲

Logs /var/log

Una vez elegido el índice se deben asignar las métricas para el eje **x** y **y**. En la siguiente imagen se muestra la métrica en eje **y**, para este caso registrará el conteo del número de logs.

lo*

Data Metrics & Axes Panel Settings

Metrics

Y-Axis

Aggregation

Count

Custom Label

veces

Add metrics

Advanced

En la siguiente se muestra que los datos reflejados en el eje **x** serán los timestamp de los logs recolectados.

Buckets

X-Axis

Aggregation

Date Histogram

Field

@timestamp

Interval

Auto

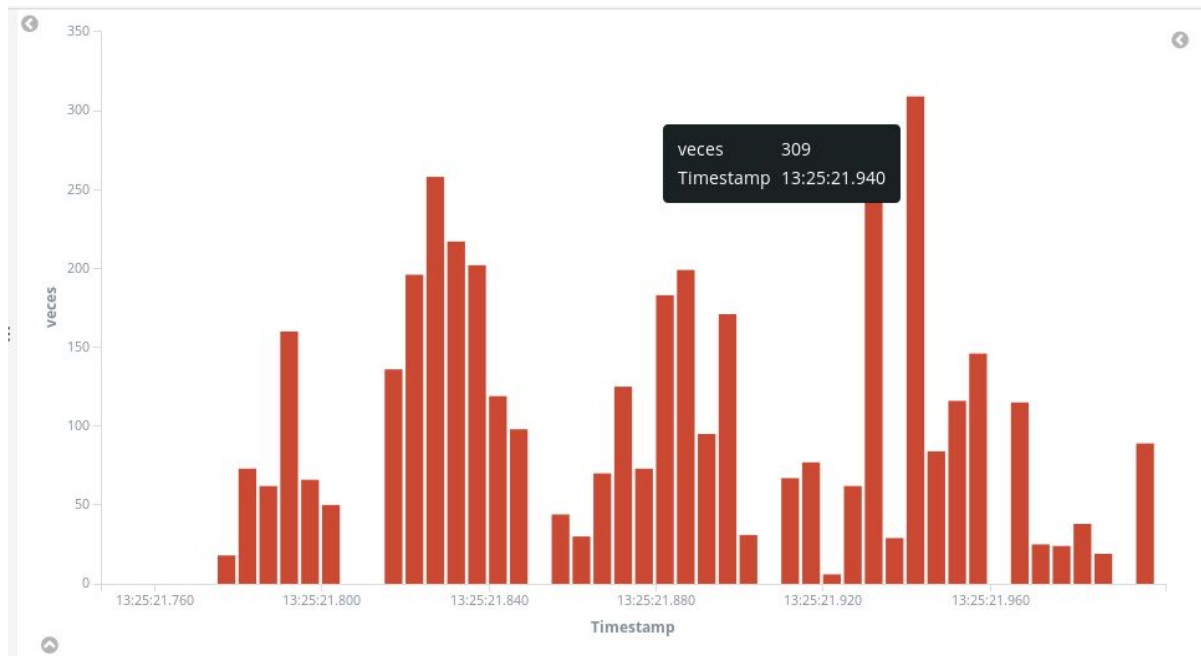
Custom Label

Timestamp

Advanced

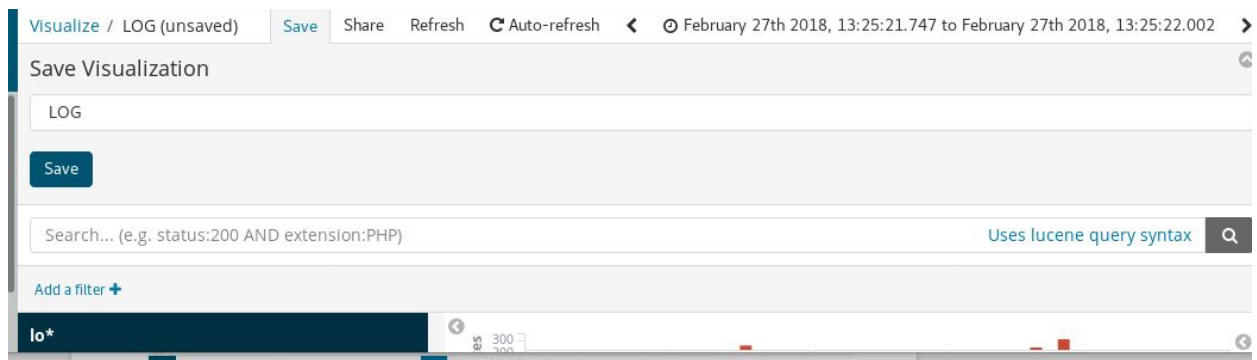
Finalmente la visualización obtenida se muestra en la siguiente imagen.

AMX/CTIN INFRAESTRUCTURA 2017



En la visualización anterior se muestran el número de logs registrados por timestamp, agrupándolos cada 5 milisegundos.

Finalmente, en las opciones de la parte superior se selecciona **save**, se asigna un nombre y nuevamente en **save**, como se muestra en la siguiente imagen, hecho lo anterior, se ha creado y almacenado una visualización la cual puede ser incluida en un dashboard.



Nota

Los logs reflejados en la visualización anterior corresponden a todos los registros almacenados en el directorio **/var/log** así como sus directorios internos con la extensión ***.log**, esto se hizo con el fin de tener una visualización con más registros, de forma que la visualización fuera más representativa.

Por tal motivo, el archivo correspondiente a **prueba.yml** ubicado dentro del directorio de **prospectors.d** de los archivos de configuración de **filebeat** (**/usr/share/filebeat/prospectors.d**) se modificó, quedando de la siguiente manera.

```
# ===== Filebeat: prueba.yml ===== #
```

paths:

- /var/log/loghost/**/*.log
- /var/log/loghost/**/*.log

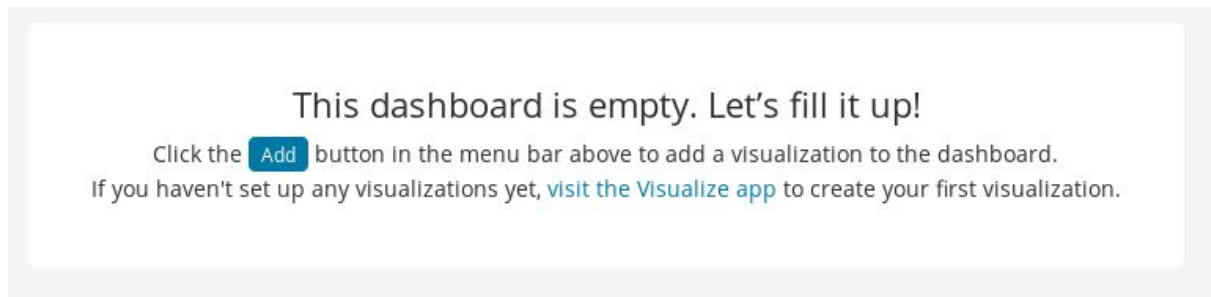
Una vez creado la visualización, lo siguiente es incluirlo en un dashboard, por tal motivo, se selecciona la pantalla **dashboard**.

Dashboard

Dentro de esta sección se encuentran listados los dashboard creados, como se muestra en la siguiente imagen, en caso de que existan. Para crear uno nuevo, se da clic sobre el botón **+**.

<input type="text" value="Search..."/> + 1-1 of 1	
Name ↑	Description
<input type="checkbox"/> Dashboard:VizualicaciOn	
1-1 of 1	

La siguiente pantalla que se muestra al seleccionar **+** será como se muestra a continuación

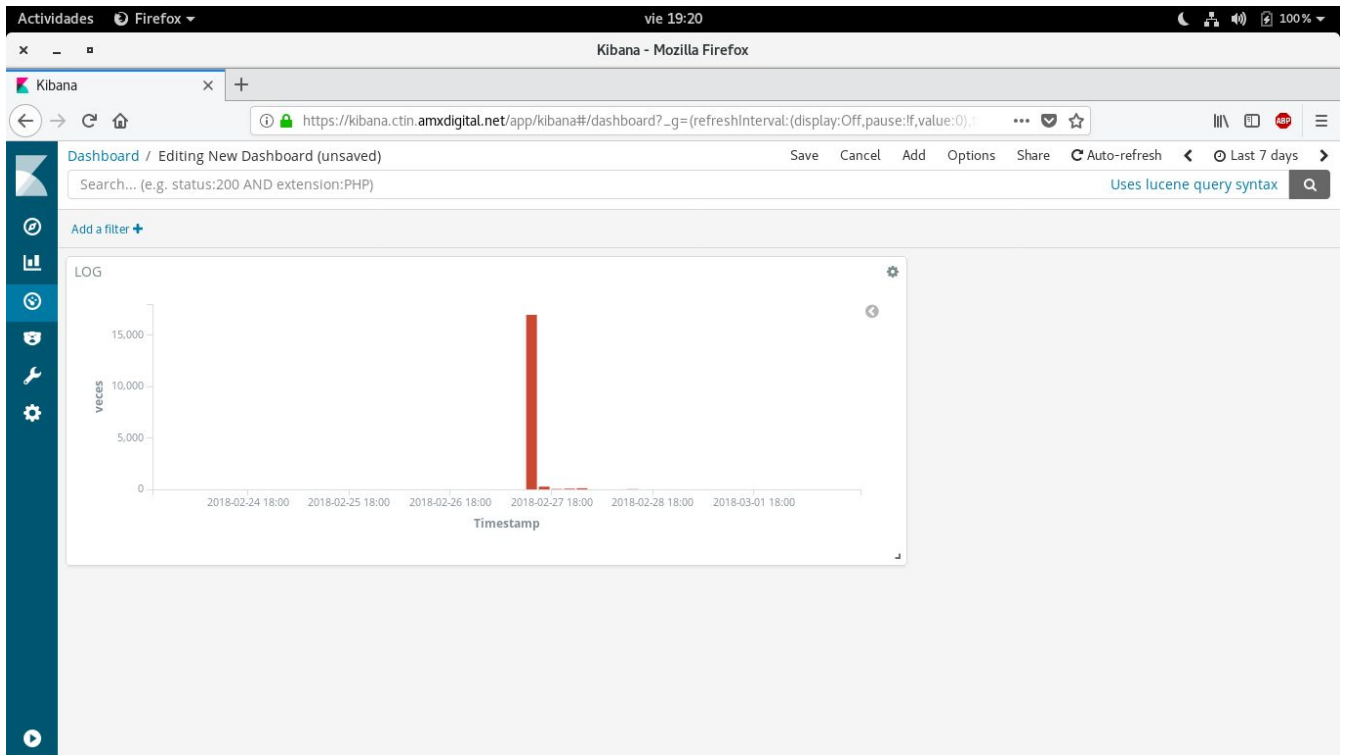


Al dar click en **add**, se listaràn las visualizaciones guardadas, como se muestra en la siguiente imagen.

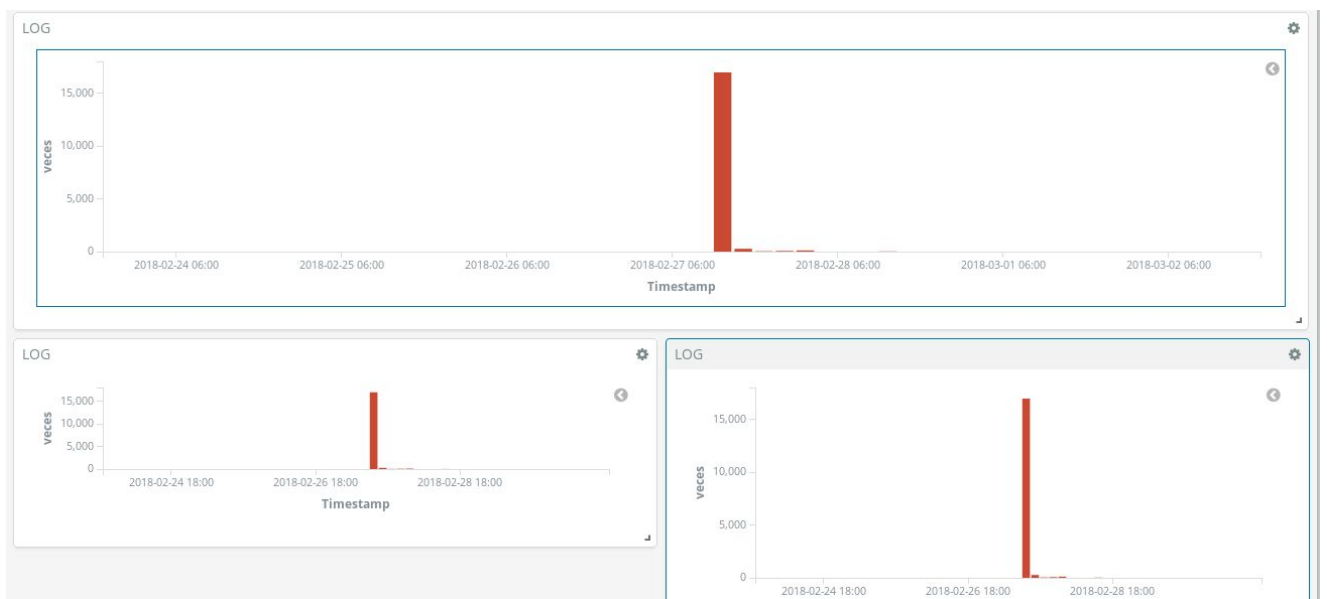
Add Panels	
Visualization	Saved Search
<input type="text" value="Visualizations Filter..."/> 1-7 of 7 Add new Visualization	
Name ▲	
Firewalld	
LOG	
Log: /var/log	
Visualize: CountAllFirewalld	
Visualize: CountTry	
Visualize: CountWarnigs	
Warnig Firewalld	

AMX/CTIN INFRAESTRUCTURA 2017

Seleccionamos la que queremos agregar a este nuevo dashboard, esta se incrustará como se muestra en la siguiente figura



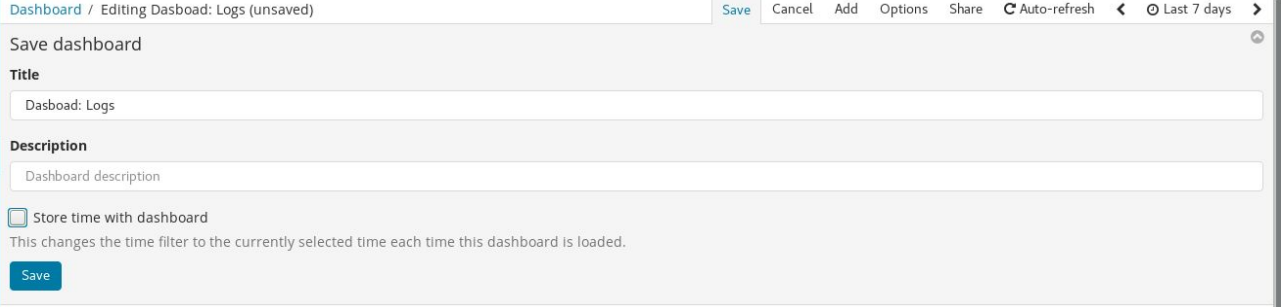
Una vez teniendo la visualización en el dashboard, esta se puede cambiar de lugar y modificar su tamaño, para mostrar lo anterior, se incrustaron las mismas visualizaciones y se modificaron en tamaño, como se muestra en la siguiente imagen.



Finalmente, al terminar de agregar las visualizaciones deseadas para el dashboard, se debe almacenar, para ello se selecciona la opción **save** ubicada en la barra de opciones del

AMX/CTIN INFRAESTRUCTURA 2017

menú superior, se asigna un nombre y se almacena, seleccionando un clic sobre **save**, como se muestra en la siguiente imagen.



The screenshot shows the 'Save dashboard' form in the AMX/CTIN interface. The breadcrumb navigation at the top reads 'Dashboard / Editing Dashboard: Logs (unsaved)'. The form has a title bar with 'Save', 'Cancel', 'Add', 'Options', 'Share', 'Auto-refresh', and a time filter set to 'Last 7 days'. The form itself has two main sections: 'Title' and 'Description'. The 'Title' section contains a text input field with the value 'Dashboard: Logs'. The 'Description' section contains a text input field with the value 'Dashboard description'. Below these fields is a checkbox labeled 'Store time with dashboard' which is currently unchecked. A note below the checkbox states: 'This changes the time filter to the currently selected time each time this dashboard is loaded.' At the bottom left of the form is a blue 'Save' button.

Hecho lo anterior, el primer dashboard se ha **creado**.

:')

Material extra

Componentes de la pantalla *Discover*

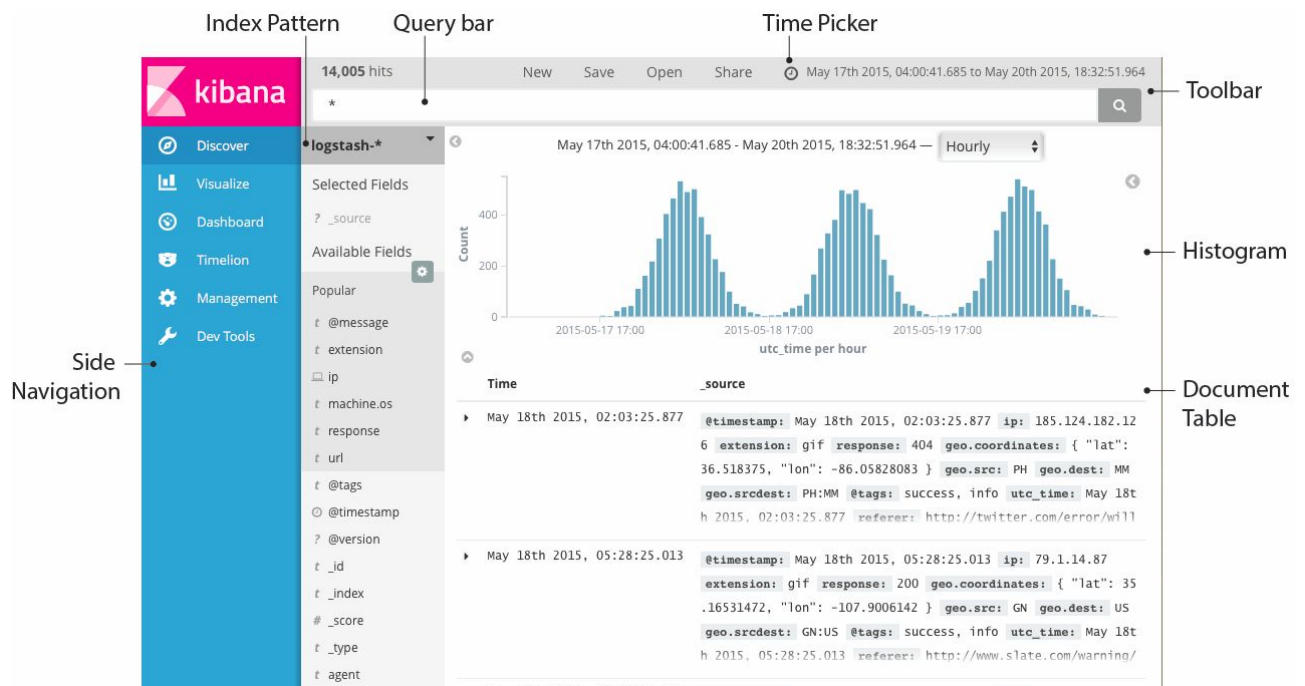


Imagen extraída de <https://www.elastic.co/guide/en/kibana/current/discover.html> 02 Marzo 2018

Metricbeat

Herramienta que recopila periódicamente métricas y estadísticas del sistema operativo y de los servicios que se ejecutan en él, enviando los datos recopilados hacia logstash o elasticsearch, estas métricas pueden ser de los servicios como:

- Apache
- HAProxy
- MongoDB
- MySQL
- Nginx
- PostgreSQL
- Redis
- System
- Zookeeper

Referencia 02 Marzo 2018:

<https://www.elastic.co/guide/en/beats/metricbeat/current/index.html>

Referencias

Elastic.co. (2018). Kibana User Guide [6.2] | Elastic. [online] Available at: <https://www.elastic.co/guide/en/kibana/current/index.html> [Accessed 20 Feb. 2018].

Elastic.co. (2018). Logstash: Collect, Parse, Transform Logs | Elastic. [online] Available at: <https://www.elastic.co/products/logstash> [Accessed 20 Feb. 2018].

Elastic.co. (2018). Constant Score Query | Elasticsearch Reference [6.2] | Elastic. [online] Available at: <https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-constant-score-query.html#query-dsl-constant-score-query> [Accessed 20 Feb. 2018].

Elastic.co. (2018). Constant Score Query | Elasticsearch Reference [6.2] | Elastic. [online] Available at: <https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-constant-score-query.html> [Accessed 20 Feb. 2018].

Discuss the Elastic Stack. (2018). Term query vs match query. [online] Available at: <https://discuss.elastic.co/t/term-query-vs-match-query/14455> [Accessed 20 Feb. 2018].

Discuss the Elastic Stack. (2018). Term query vs match query. [online] Available at: <https://discuss.elastic.co/t/term-query-vs-match-query/14455> [Accessed 20 Feb. 2018].