2. Agile SysAdmin: Networking and Systems Administration

Práctica

Despliegue stack ELK

Durante las sesiones se han explicado numerosos temas como:
- Trabajar con la línea de comandos
- Instalación de software
- Gestión de servicios
- Scripting
- Protocolos más comunes en Internet
- Gestión de Almacenamiento
- Análisis de rendimiento
- Seguridad y monitorización

El objetivo de la práctica de este módulo pasa por intentar profundizar sobre esos conceptos. Para conseguirlo se pedirá que se monten varios servicios en una o varias máquinas virtuales.

Como ejemplo se realizará la implementación de una herramienta de gestión de logs unificada. Para ello nos basaremos en el stack ELK (Elasticsearch, Logstash, Kibana). (El objetivo de la práctica no es conocer el funcionamiento de ELK, se hará una instalación

básica y funcional)

Deberemos instalar estos tres elementos de la pila, se puede instalar todo en la misma máquina si no se disponen de suficientes recursos para ello.

El primer elemento, Logstash, se encarga de analizar registros que se reciben, para parsearlos y separarlos en campos. Una vez parseados los envía a un output que en nuestro caso será Elasticsearch.

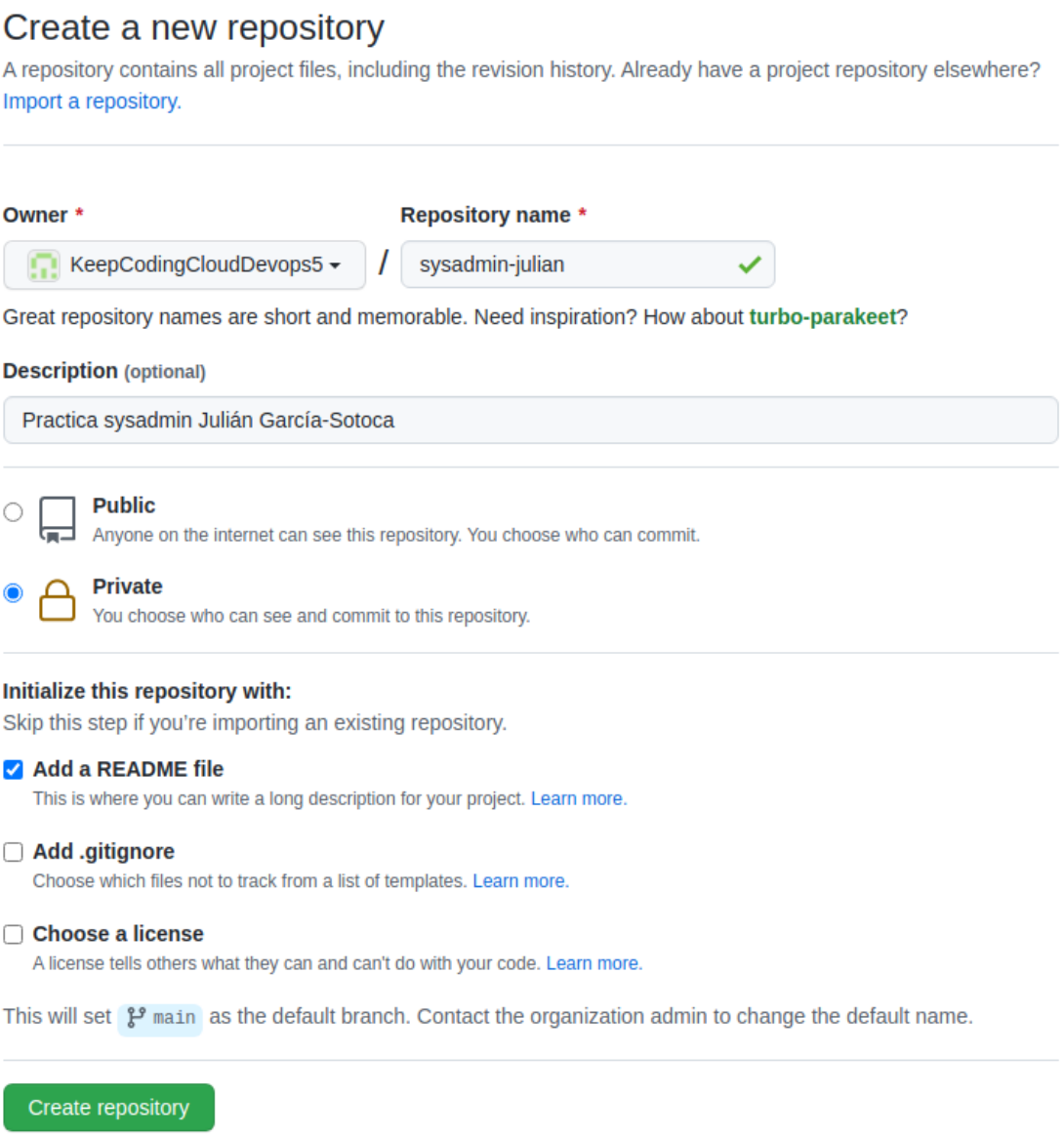
Elasticsearch es un tipo de Base de Datos optimizada para la gestión de documentos y la búsqueda en ellos. Una vez recibe el documento desde Logstash lo procesa e indexa. Finalmente desde Kibana se pueden visualizar los datos y crear dashboards.

Para enviar los datos montaremos otro stack, donde instalaremos un servicio web tipo Wordpress. **Se da libertad al alumno a elegir la aplicación que considere siempre que se instale sobre un servidor Linux y los datos se almacenen en una Base de Datos, ya sea SQL (MySQL, MariaDB o Postgresql) o NoSQL (MongoDB)**. En el servidor donde se instale este stack se instalará el servicio encargado de enviar los logs de la BBDD y del servidor web al logstash, mediante filebeat. (https://www.elastic.co/es/products/beats).

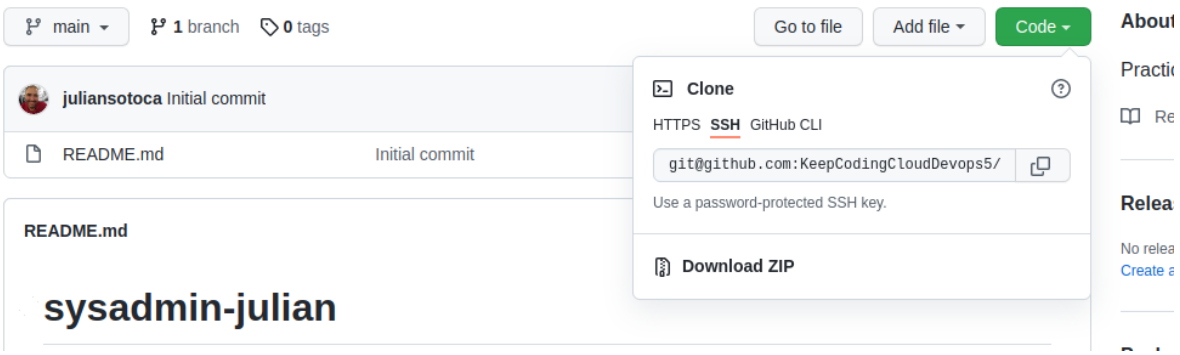
Detalle de la práctica

Creación del repositorio en Github

Desde la pantalla principal del Github de keepcoding crearemos un nuevo repositorio con el nombre sysadmin-<nombre del alumno> y añadiendo el fichero README:



Una vez creado podremos obtener los enlaces para clonarlo y empezar a trabajar con él:



Es importante que antes de entregar la práctica os aseguréis de que tengo permisos para revisarlo. Para ello en Settings → Manage Access debe aparecer mi nombre con permisos de lectura.

Ya que vamos a trabajar con ficheros de log y máquinas virtuales, es importante no subir al repositorio esta información que no es necesaria para la replicación del entorno. Por ello, en el repositorio deberá existir un fichero **.gitignore** con el siguiente contenido:

```
*/.vagrant
.vagrant
*.vmdk
*.log
.DS_Store
```

Podéis añadir todas las excepciones que consideréis necesarias para no subir ficheros innecesarios.

Nota: Si en el ordenador donde vas a realizar la práctica no funciona Vagrant con Virtualbox puedes utilizar algunos de los proveedores cloud que ofrecen una capa gratuita para levantar 2 máquinas virtuales y realizar la instalación a través de los scripts.

Despliegue de las máquinas virtuales

Para el despliegue de las máquinas virtuales usaremos Vagrant como hemos ido haciendo a lo largo del curso. Desplegaremos dos máquinas virtuales, una donde instalaremos el servidor de Wordpress, y otra donde instalaremos el Stack ELK.
El sistema operativo base que usaremos será Ubuntu 20.04, la última versión LTS.
Los requisitos para las máquinas virtuales serán los siguientes:

- Imagen base: `ubuntu/focal64`
- VM1 (Wordpress): 1 CPU, 1GB RAM
- VM2 (Elasticsearch): 1 CPU, 3GB RAM
- Ambas máquinas deberán tener visibilidad entre ellas, para ello habrá que configurarles una “private_network” y asignarle IPs en el mismo rango
- Se deberá configurar el port forwarding del puerto 80, para la máquina de Wordpress, y del puerto 9200 y 80 para la máquina de ELK.
- Ambas máquinas deberán tener un disco adicional que será usado para la base de datos MariaDB de Wordpress y para el propio Elasticsearch respectivamente.
- Ambas máquinas se autoprovisionarán con un script de “provision”.

Todo esto lo encapsulamos en un Vagrantfile que añadiremos al repositorio.

Script de provision de la VM1

En esta primera máquina virtual vamos a instalar un servidor Wordpress. Para configurarlo es necesario instalar y configurar los siguientes componentes:

- Nginx
- MariaDB

- Wordpress

A parte habrá que instalar el Filebeat para el envío de logs a la máquina con el Logstash. Como paso previo a todo eso deberemos configurar el sistema operativo.

Configuración de los puntos de montaje

En este punto vamos a configurar el punto de montaje para la Base de Datos de Wordpress. Para ello deberemos usar la herramienta parted para crear la tabla de particiones y crear una con todo el disco. A continuación usaremos LVM para crear un volumen lógico a partir de ese disco. Una vez tengamos el volumen lógico lo formateamos como EXT4 y lo montaremos en /var/lib/mysql. Debemos asegurarnos de que se monta al reiniciar la máquina virtual añadiendo la correspondiente entrada en el FSTAB.

Instalación de Nginx y MariaDB

Seguiremos los siguientes pasos:

1. Actualización de repositorios
2. Instalación de los siguientes paquetes: `nginx mariadb-server mariadb-common php-fpm php-mysql expect php-curl php-gd php-intl php-mbstring php-soap php-xml php-xmlrpc php-zip`
3. Una vez tengamos nginx, MariaDB y las dependencias de PHP instalados deberemos configurarlos. Para configurar el Nginx usaremos la siguiente configuración en el fichero `/etc/nginx/sites-available/wordpress` (se puede usar heredoc para crearlo):

```
# Managed by installation script - Do not change
server {
        listen 80;
        root /var/www/wordpress;
        index index.php index.html index.htm index.nginx-debian.html;
        server_name localhost;

        location / {
                try_files \$uri \$uri/ =404;
        }

        location ~ \.php\$ {
                include snippets/fastcgi-php.conf;
                fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
        }

        location ~ /\.ht {
                deny all;
        }
}
```

4. Esta configuración la habilitaremos creando un enlace simbólico en /etc/nginx/sites-enabled/ y borrando el fichero que viene por defecto.
5. Con el nginx ya configurado podremos habilitar los servicios de nginx y php7.4-fmp.
6. Antes de poder usar la BBDD MariaDB deberemos securizarla como hemos visto durante las sesiones.

Instalación de Wordpress

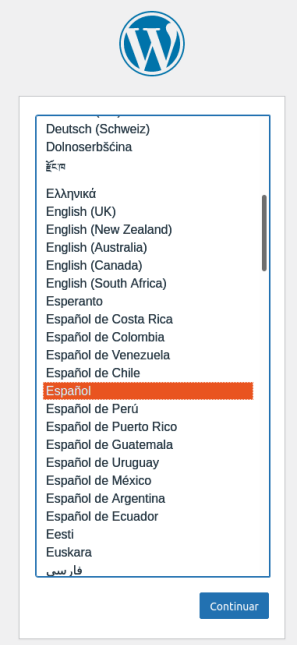
Para la instalación de Wordpress deberemos ejecutar los siguientes pasos:

1. Creación de la BBDD para wordpress. Podemos usar heredoc o crear un fichero .sql que se lo pasaremos al comando mysql:

```
CREATE DATABASE wordpress DEFAULT CHARACTER SET utf8 COLLATE
utf8_unicode_ci;
GRANT ALL ON wordpress.* TO 'wordpressuser'@'localhost' IDENTIFIED BY
'keepcoding';
FLUSH PRIVILEGES;
```

2. Descargar la última release: https://wordpress.org/latest.tar.gz
3. Descomprimimos en /var/www/wordpress
4. Configuraremos la conexión a BBDD en el fichero wp-config.php
5. Nos aseguraremos de que el directorio /var/www/wordpress es del usuario y grupo www-data

Siguiendo estos pasos ya deberíamos poder acceder a la siguiente URL para empezar a configurar el Wordpress: http://localhost:8081



Instalación de Filebeat

La instalación de Filebeat requiere la configuración del repositorio de Elastic.co. Para ello primero deberemos importar la Key de su repositorio y a continuación se añade el repositorio:

```
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | apt-key add -
echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | tee -a
/etc/apt/sources.list.d/elastic-7.x.list
```

Con el repositorio ya instalado se deben actualizar los índices de APT y ya podemos instalar el paquete filebeat.
Una vez instalado habilitaremos los módulos de system y de nginx:

```
filebeat modules enable system
filebeat modules enable nginx
```

Antes de arrancar el servicio debermos configurar el filebeat indicándole los ficheros de logs que queremos que monitorize y el destino donde debe enviar esos logs, en nuestro caso el logstash.

```
filebeat.inputs:

- type: log
 enabled: true
 paths:
   - /var/log/*.log
   - /var/log/nginx/*.log
   - /var/log/mysql/*.log

...

output.logstash:
 hosts: ["192.168.100.6:5044"]

...
```

Existe un fichero por defecto que se puede usar como referencia para crear uno específico a nuestra instalación modificando únicamente los parámetros anteriormente indicados: /etc/filebeat/filebeat.yml
Finalmente, una vez configurado, ya se puede arrancar el servicio de filebeat: `systemctl enable filebeat --now`

Script de provisión de la VM2

Configuración de los puntos de montaje

Seguiremos la misma metodología que para la VM1, en este caso el punto de montaje será `/var/lib/elasticsearch`

Instalación de dependencias

Las herramientas del stack ELK son aplicaciones Java, con lo que es importante tener instalado el JDK antes de instalar estos elementos. Para ello bastará con instalar el paquete default-jre.
También, vamos a usar nginx para publicar el Kibana (la interfaz web del Stack) a través de un puerto estándar. Por tanto también instalaremos el paquete nginx.

El resto de herramientas se instalan desde el repositorio de Elastic.co, por tanto, también es necesario, como paso previo, configurar este repositorio igualmente que lo hemos hecho en la VM1.

Instalación de Logstash

La instalación de Logstash consiste en simplemente instalar el paquete logstash. Una vez instalado tendremos el directorio /etc/logstash donde estarán ubicados los ficheros de configuración.

El funcionamiento de logstash consiste en tres fases:

1. Input: definimos de donde esperamos recibir logs
2. Filtros: se configura cómo se van a parsear estos logs
3. Output: se establece el destino de estos logs una vez parseados

En nuestro caso, el input será el filebeat y el output la propia máquina local en elasticsearch que instalaremos a continuación.
La configuración deberá ser similar a la siguiente:

/etc/logstash/conf.d/02-beats-input.conf

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

/etc/logstash/conf.d/10-syslog-filter.conf

```
filter {
 if [fileset][module] == "system" {
   if [fileset][name] == "auth" {
     grok {
       match => { "message" => ["%{SYSLOGTIMESTAMP:[system][auth][timestamp]}
%{SYSLOGHOST:[system][auth][hostname]}
sshd(?:\[%{POSINT:[system][auth][pid]}\])?: %{DATA:[system][auth][ssh][event]}
%{DATA:[system][auth][ssh][method]} for (invalid user
)?%{DATA:[system][auth][user]} from %{IPORHOST:[system][auth][ssh][ip]} port
%{NUMBER:[system][auth][ssh][port]} ssh2(:
%{GREEDYDATA:[system][auth][ssh][signature]})?",
                 "%{SYSLOGTIMESTAMP:[system][auth][timestamp]}
%{SYSLOGHOST:[system][auth][hostname]}
sshd(?:\[%{POSINT:[system][auth][pid]}\])?: %{DATA:[system][auth][ssh][event]}
user %{DATA:[system][auth][user]} from %{IPORHOST:[system][auth][ssh][ip]}",
                 "%{SYSLOGTIMESTAMP:[system][auth][timestamp]}
%{SYSLOGHOST:[system][auth][hostname]}
sshd(?:\[%{POSINT:[system][auth][pid]}\])?: Did not receive identification string
from %{IPORHOST:[system][auth][ssh][dropped_ip]}",
                 "%{SYSLOGTIMESTAMP:[system][auth][timestamp]}
%{SYSLOGHOST:[system][auth][hostname]}
sudo(?:\[%{POSINT:[system][auth][pid]}\])?: \s*%{DATA:[system][auth][user]} :(
%{DATA:[system][auth][sudo][error]} ;)? TTY=%{DATA:[system][auth][sudo][tty]} ;
PWD=%{DATA:[system][auth][sudo][pwd]} ; USER=%{DATA:[system][auth][sudo][user]} ;
COMMAND=%{GREEDYDATA:[system][auth][sudo][command]}",
                 "%{SYSLOGTIMESTAMP:[system][auth][timestamp]}
%{SYSLOGHOST:[system][auth][hostname]}
groupadd(?:\[%{POSINT:[system][auth][pid]}\])?: new group:
name=%{DATA:system.auth.groupadd.name}, GID=%{NUMBER:system.auth.groupadd.gid}",
                 "%{SYSLOGTIMESTAMP:[system][auth][timestamp]}
%{SYSLOGHOST:[system][auth][hostname]}
useradd(?:\[%{POSINT:[system][auth][pid]}\])?: new user:
name=%{DATA:[system][auth][user][add][name]},
UID=%{NUMBER:[system][auth][user][add][uid]},
GID=%{NUMBER:[system][auth][user][add][gid]},
home=%{DATA:[system][auth][user][add][home]},
shell=%{DATA:[system][auth][user][add][shell]}$",
                 "%{SYSLOGTIMESTAMP:[system][auth][timestamp]}
%{SYSLOGHOST:[system][auth][hostname]}
%{DATA:[system][auth][program]}(?:\[%{POSINT:[system][auth][pid]}\])?:
%{GREEDYMULTILINE:[system][auth][message]}"] }
       pattern_definitions => {
```

```
        "GREEDYMULTILINE"=> "(.|\n)*"
       }
       remove_field => "message"
     }
     date {
       match => [ "[system][auth][timestamp]", "MMM  d HH:mm:ss", "MMM dd
HH:mm:ss" ]
     }
     geoip {
       source => "[system][auth][ssh][ip]"
       target => "[system][auth][ssh][geoip]"
     }
   }
   else if [fileset][name] == "syslog" {
     grok {
       match => { "message" => ["%{SYSLOGTIMESTAMP:[system][syslog][timestamp]}
%{SYSLOGHOST:[system][syslog][hostname]}
%{DATA:[system][syslog][program]}(?:\[%{POSINT:[system][syslog][pid]}\])?:
%{GREEDYMULTILINE:[system][syslog][message]}"] }
       pattern_definitions => { "GREEDYMULTILINE" => "(.|\n)*" }
       remove_field => "message"
     }
     date {
       match => [ "[system][syslog][timestamp]", "MMM  d HH:mm:ss", "MMM dd
HH:mm:ss" ]
     }
   }
 }
}
```

/etc/logstash/conf.d/30-elasticsearch-output.conf

```
output {
 elasticsearch {
   hosts => ["localhost:9200"]
   manage_template => false
   index => "%{[@metadata][beat]}-%{[@metadata][version]}-%{+YYYY.MM.dd}"
 }
}
```

Una vez configurado podremos arrancar el servicio: `systemctl enable logstash --now`

Instalación de Elasticsearch

La instalación de elasticsearch únicamente consiste en instalar el paquete elasticsearch. Por defecto ya usará la ruta /var/lib/elasticsearch para almacenar la base de datos, pero deberemos asegurarnos antes de arrancar el servicio que el usuario elasticsearch tiene permisos para escribir ahí.

Una vez instalado arrancamos el servicio con `systemctl enable elasticsearch --now`

Instalación de Kibana

La instalación de kibana igualmente que en los anteriores casos consiste en instalar el paquete kibana, y con la configuración por defecto será suficiente. Como queremos usar un puerto estandard modificaremos la configuración del nginx

(`/etc/nginx/sites-available/default`) para que redirija las peticiones al puerto 80 al puerto del kibana:

```
# Managed by installation script - Do not change
server {
   listen 80;

   server_name kibana.demo.com localhost;

   auth_basic "Restricted Access";
   auth_basic_user_file /etc/nginx/htpasswd.users;

   location / {
       proxy_pass http://localhost:5601;
       proxy_http_version 1.1;
       proxy_set_header Upgrade \$http_upgrade;
       proxy_set_header Connection 'upgrade';
       proxy_set_header Host \$host;
       proxy_cache_bypass \$http_upgrade;
   }
}
```

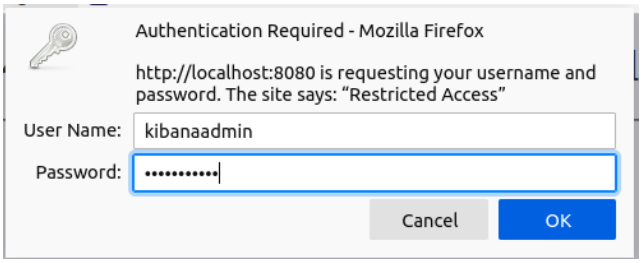
Ademas, hemos configurado una autenticación básica, deberemos generar para ello un fichero htpasswd.users donde especificaremos el usuario y contraseña encriptado. Para la creación de ese fichero será necesario tener anteriormente un fichero .kibana con la contraseña que vamos a usar. Ese fichero al estar en el directorio de Vagrant desde la VM se va a poder acceder a el en la ruta /vagrant. Para la generación del fichero de contraseñas ejecutaremos:

```
echo "kibanaadmin:$(openssl passwd -apr1 -in /vagrant/.kibana)" | sudo tee -a
/etc/nginx/htpasswd.users
```

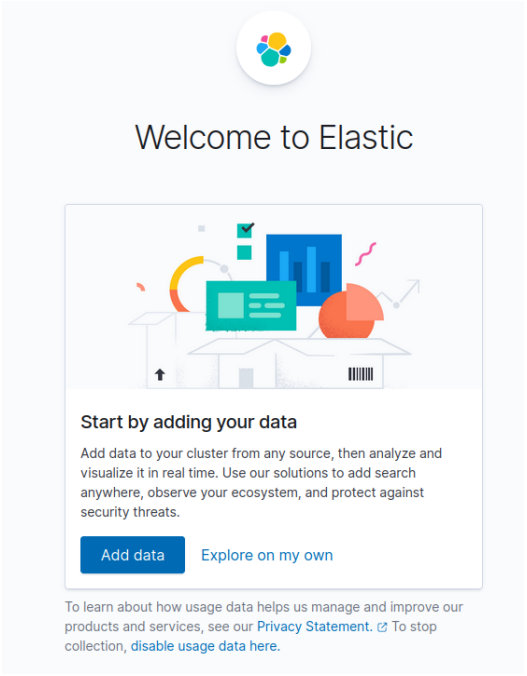
Una vez configurado el nginx reiniciaremos los servicios de nginx y kibana.

Configuración de Kibana

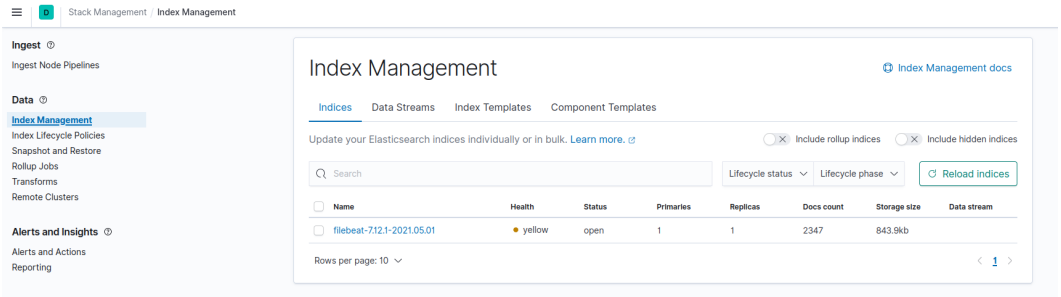
Como hemos indicado en el apartado de instalación de kibana, hemos habilitado una autenticación básica a nivel de nginx. Al intentar acceder a la interfaz http://localhost:8080/ nos pedirá el usuario y password configurado:



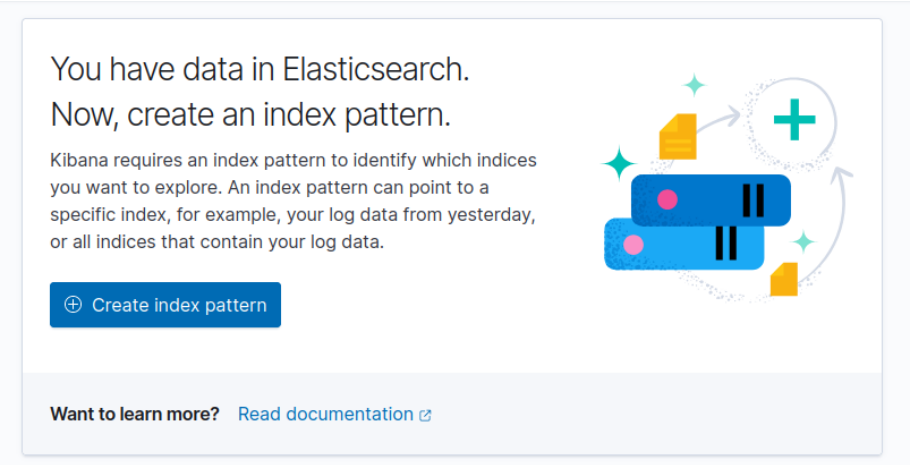
La primera vez que entremos en kibana nos dirá que no tiene datos, de momento pulsaremos en “Explore on my own” para configurarlo como lo necesitamos:



En el menú de la izquierda, en Management, seleccionaremos Stack Management. Una vez dentro podremos ver si se nos han creado los índices. si ha ido todo bien deberemos tener un índice que empiece por filebeat-.:

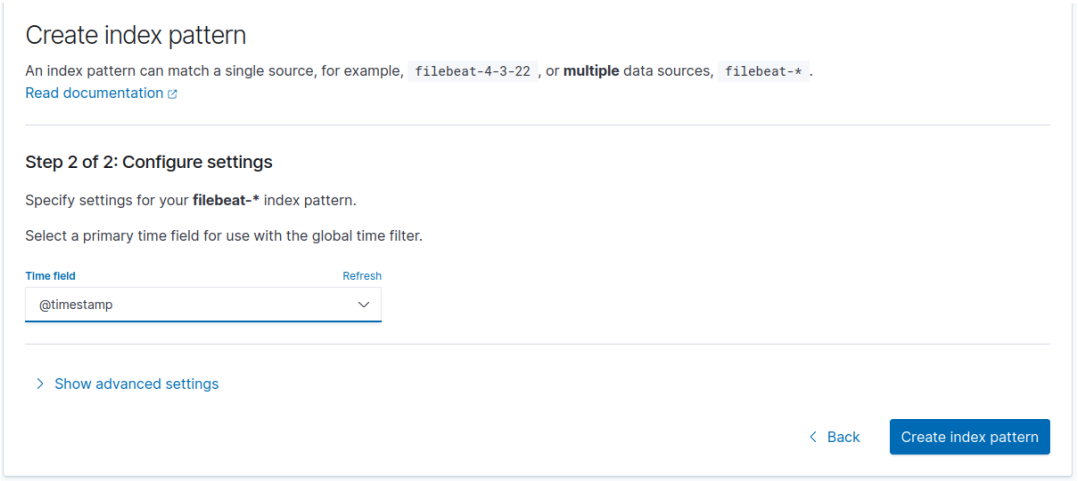


Una vez sabemos que tenemos el índice, debemos crear un “Index pattern”. Para ello lo seleccionaremos en el menú de la izquierda y crearemos uno nuevo:

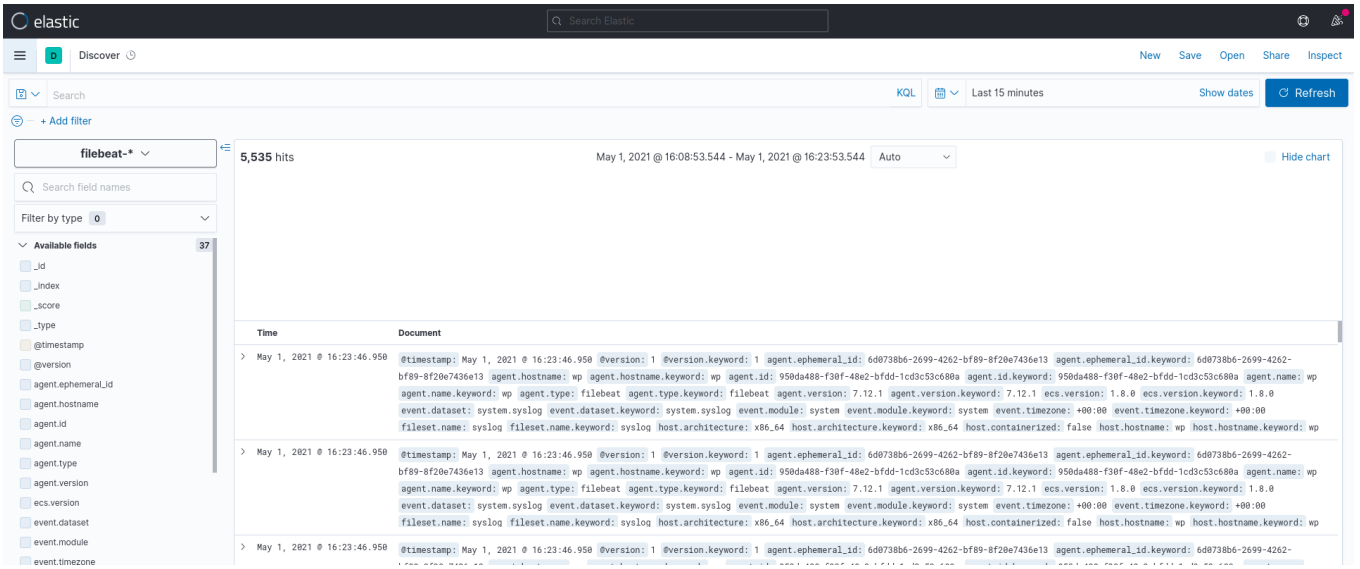


Indicaremos el nombre que va a tener, debe coincidir con el nombre de nuestro índice de elasticsearch. Si es correcto nos dirá que machea correctamente:

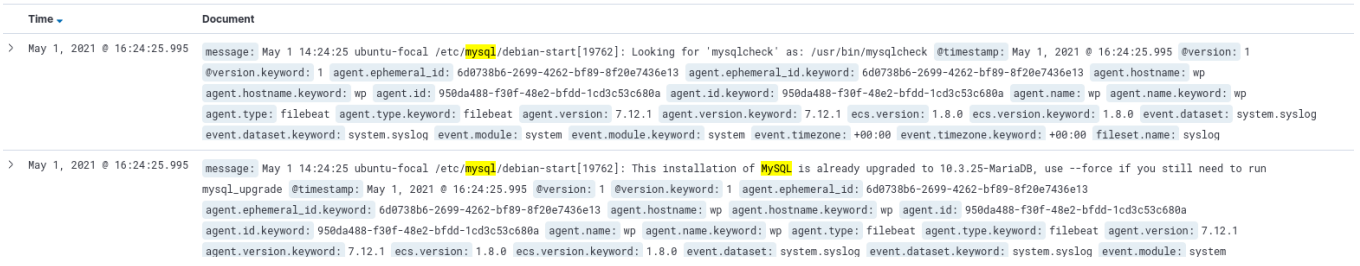
En el siguiente paso nos ide qué campo es el que se usará como referencia temporal, en nuestro caso seleccinamos @timestamp:



Volviendo al menú principal, en Discover, ya podremos navegar por los logs de nuestro servidor de Wordpress. Podemos generar algo de tráfico navegando o reiniciando servicios en la VM o deberíamos ver cómo los logs van apareciendo en la interfaz web:



Podemos hacer búsquedas por palabras clave:



O podemos hacer algunas búsquedas más complejas:

Evaluación

Para la evaluación de la práctica se tendrá en cuenta:

- Debes entregar los scripts de instalación de todo el stack en un repositorio del **GITHUB** de Keepcoding.
- Debes darme acceso a tu repositorio para que yo sea capaz de clonarlo y mediante esos scripts desplegar las aplicaciones.
- Debes incluir un README.md en el repositorio con instrucciones de cómo lanzar los despliegues.
- Debes incluir evidencias de Kibana demostrando que te llegan los logs de la aplicación