



Facultad De Ciencias Exactas, Físicas Y Naturales

INGENIERÍA EN COMPUTACIÓN

PRÁCTICA PROFESIONAL SUPERVISADA “Implementación de sistema de logs centralizados” en la Prosecretaría de Informática - UNC

Informe Final (200 Horas)

Alumno:

Sepulveda, Federico Nicolás (federico.sepulveda@alumnos.unc.edu.ar)

Supervisor:

Menzaque, Fernando (fernando.menzaque@unc.edu.ar)

Tutor:

Ing. Britos, Daniel (dbritos@unc.edu.ar)

2017

Índice

Índice	1
Objetivo	3
Sistema de Logs Centralizados	3
¿Que es un log?	3
¿Por qué un sistema de log centralizados?	3
Desarrollo de Proyecto	4
Investigación de Herramientas Disponibles	5
Envío de mensajes desde los clientes al servidor (logs)	5
Recepción de mensajes	6
Almacenamiento y búsqueda	7
Visualización de datos	8
Elección de las Herramientas	9
Funcionamiento de las Herramientas	10
Programas	10
Puesta en funcionamiento y plataforma de instalación	10
Elección de Docker como plataforma de instalación	11
Arquitecturas de ELK	11
Elección de arquitectura	11
Configuración de ELK simple	11
Esquema de conexión ELK simple	12
Configuración de ELK básico	12
Elasticsearch	12
Filebeat	13
Kibana	18
Logstash	19
Configuración de ELK Avanzado	22
Esquema de conexión ELK avanzado	22
Configuración de ELK avanzado	22
Elasticsearch	22
Filebeat	23
Kibana	27
Logstash Shipper	27
Logstash Index	30
Redis	31
Creacion de Imagenes de Docker	32

Configuraciones de Arranque	37
Instrucciones de uso ELK	40
Consideraciones a Futuro	45
Información adicional	46
Información sobre volúmenes para Elasticsearch	46
Información sobre cluster para Elasticsearch	46
Información sobre volúmenes para Redis	46
Información sobre cluster para Redis	46
Escalabilidad de ELK	46

Objetivo

El objetivo de esta práctica supervisada es implementar un sistema de logs centralizado para la Prosecretaría de Informática de la Universidad Nacional de Córdoba.

Sistema de Logs Centralizados

¿Que es un log?

Los logs son archivos del sistema o de aplicaciones, que va generando un dispositivo, para el caso del data center puede ser un servidor, un router, un switch, etc. Estos archivos contienen información útil sobre el sistema o aplicación, ya que se guarda la hora y fecha de cada evento relevante para los mismos, por ejemplo si es información sobre el sistema puede ser cuando se encendió o se apagó. Si es el registro de una aplicación, por ejemplo una aplicación de servicios web, esta puede ir almacenando las consultas que se le han hecho, quien se las ha hecho y cuando se realizaron. Estos registros no tienen un formato estándar, por lo tanto varían según el dispositivos y/o aplicación.

¿Por qué un sistema de log centralizados?

Se busca implementar una plataforma que reciba los logs de los equipos del data center, de esta manera se centralizan todos los registros haciendo que la búsqueda sea más ágil y sencilla. En base a los datos de los registro se puede tener un mejor monitoreo del sistema, como así también se pueden generar las alertas necesarias. También permite poner a resguardo los registro del sistema del mismo administrador o de intrusos que quieran modificarlos o borrarlos. Otra ventaja que ofrece es la correlacionar eventos a través de los registros por ejemplo cuando se detecta que quisieron acceder por fuerza bruta a varios servidores desde una misma IP. Además permite liberar espacio del disco rígido transfiriendo los registros viejos al servidor de logs centralizado.

Desarrollo de Proyecto

Plan de actividades

- Investigación de las herramientas
- Implementación de los requerimientos necesarios
- Realizar configuraciones
- Pruebas de funcionamiento básicas
- Implementación de los requerimientos deseables
- Puesta en funcionamiento
- Documentación

Objetivos a alcanzar -PPS

Generales

- Instalación por paquetes
- Aplicable a toda la infraestructura de la PSI
- Instalación en servidor físico
- Instalar los programas en un servidor físico
- Manual y procedimiento de instalación (ansible)
- Instructivo para agregar un servidor
- Control de syslog
- Registro de lo que se loguea
- Instructivo para mandar un archivo a syslog
- Plugins para búsquedas
- Utilizar protocolo UDP para enviar los logs

Redes

Separar los distintos tipos de dispositivos (local x)

Guardar en los logs el identificador del equipo e identificador de servicio (etiquetas flexibles)

- Equipos de red
 - Problema de hardware, reinicio del equipo, CRASHED
- Protocolos que soportan los equipos de red
 - Syslog -> Escritura en disco

Servicios

- Logs de syslog
El sistema tiene que ser capaz de recibir y almacenar los logs de syslog
- Logs de archivos
El sistema tiene que ser capaz de recibir y almacenar los logs que las aplicaciones almacenan como archivos.

Distribución de Horas:

Investigación de las herramientas: 56 Hs.

Armado de Informe: 6 Hs.

Reunion de presentacion: 1 Hs

Comparación de Herramientas: 2 Hs.

Instalación y pruebas desarrollo simple: 86 Hs

Instalación y pruebas desarrollo avanzado: 52 Hs

Testing y documentación: 24 Hs.

Total: 227Hs.

Investigación de Herramientas Disponibles

A continuación se ven las tablas comparativas de las herramientas disponibles para cada etapa del sistema

Envío de mensajes desde los clientes al servidor (logs)

- syslog-ng
- rsyslog
- Filebeat

Sistemas operativos compatibles:

SO/Software	syslog-ng	rsyslog	Filebeat
SO Ubuntu	si	si	si
SO Debian	si	si	si
SO Centos	si	si	si
SO Oracle Linux	no	si*	no
SO Opensuse	si	si*	si*

*: la información no es de una página oficial del producto

Características:

Características/ Software	syslog-ng	rsyslog	Filebeat
Envía log Apache	si	si	si
Envía log syslog	si	si	si
Instalación por paquetes	si	si	si
Repositorio	No oficial	Por defecto en linux	Oficial
Envío de logs automatico	si	si	si

Recepción de mensajes

- Graylog Server
- Logstash
- Apache Flume
- rsyslog
- Fluentd
- nxlog

Sistemas operativos compatibles:

SO/Software	Graylog Server	Logstash	Apache Flume	rsyslog	Fluentd	nxlog
SO Ubuntu	si	si	si	si	si	si
SO Debian	si	si	si*	si	no	si
SO Centos	si	si	si	si	si	si
SO Oracle Linux	no	no	no	si*	no	no
SO Opensuse	no	no	no	si*	no	no

*: la información no es de una página oficial del producto

Tipo de recepción:

Tipo de recepción/ Software	Graylog Server	Logstash	Apache Flume	rsyslog	Fluentd	nxlog
Recibe log SNMP trap	Si, con plugin	Si, con plugin	Si, con plugin	si*	si*	si(enterprise edition)
Recibe log SNMP	no	no	no	si*	Si, con plugin	no
Recibe log syslog	si	si	si	si	si	si
Recibe por protocolo UDP	si	Si, con plugin	si	si	Si, con plugin	Si, con modulo

*: la información no es de una página oficial del producto

Características:

Características/ Software	Graylog Server	Logstash	Apache Flume	rsyslog	Fluentd	nxlog
Permite etiquetar mensajes	si	si	no	si	si	si
Instalación por paquetes	si	si	si	si	si	no
Propósito	Recibe consultas http	Colector de mensajes. Filtro (pipelining)	Colector de mensajes	Captura, procesamiento transporte de mensajes (Envío y recepcion)	Colector de mensajes	Colector de mensajes. Filtro.
Servicio pago o gratuito	gratuito	gratuito	gratuito	gratuito	gratuito	Gratuito (version community)
Repositorio plugin	oficial	oficial	No oficial	--	No oficial	oficial

Almacenamiento y búsqueda

- Elasticsearch
- Apache solr
- MongoDB

Sistemas operativos compatibles:

SO/Software	Elasticsearch	Apache solr	MongoDB
SO Ubuntu	si	si	si
SO Debian	si*	si	si
SO Centos	si	si	si
SO Oracle Linux	no	no	no
SO Opensuse	si	si	si

Características:

Características/ Software	Elasticsearch	Apache solr	MongoDB
Permite insertar etiquetas	si	si	si
Búsqueda por etiquetas	si	si	si
Análisis en tiempo real	si	si	si
Escalabilidad de procesamiento	si	si	si
Escalabilidad de capacidad	si	si	si
Destaca	Escalabilidad horizontal	Velocidad de búsqueda	--
Instalación por paquetes	si	no	si

Visualización de datos

- Graylog Web (graylog)
- Kibana (elastic)
- Pentaho

Sistemas operativos compatibles:

SO/Software	Graylog Web	Kibana	Pentaho
SO Ubuntu	si	si	si
SO Debian	no	si	no
SO Centos	si	si	si
SO Oracle Linux	no	no	no
SO Opensuse	no	no	no

Características:

Características/ Software	Graylog Web	Kibana	Pentaho
Instalación por paquetes	si	si	si
Browser soportados	<ul style="list-style-type: none">• Chrome• Firefox• Internet Explorer• Microsoft Edge• Safari	<ul style="list-style-type: none">• IE11+• Firefox• Chrome• Safari (Mac)• Safari (iOS)• Chrome (Android)	<ul style="list-style-type: none">• Apple Safari 6.x & 7.x• Google Chrome 36 – 37• Microsoft Internet Explorer 8 – 11*****• Mozilla Firefox 31 - 32

Elección de las Herramientas

Se optó por usar el stack Elastic porque tiene una gran flexibilidad para la recepción de logs y otros tipos de mensajes ya que tiene una amplia gama de plugin (Logstash), también tiene una configuración bastante simple ya que se puede separar en distintos archivos para entradas, filtros y salidas. Elasticsearch también es una base de datos que tiene la posibilidad de escalamiento horizontal, lo cual es muy útil para ambientes que se maneja mucha información como lo es la PSI. Al optar por usar Elasticsearch se utiliza Kibana para poder realizar búsqueda y visualizar los mensajes. Además si se desea se puede personalizar la manera de visualizar los mensajes con facilidad bajando nuevos dashboard. El uso de Filebeat es opcional porque se puede reemplazar por otra aplicación (rsyslog), en este proyecto se lo va a incluir para comprobar su funcionamiento. Otra ventaja de usar un stack de un mismo desarrollador, es el fácil acoplamiento entre los sistemas que fueron hechos con el propósito de centralizar logs.

Funcionamiento de las Herramientas

Programas

- Elasticsearch: es una base de datos no relacional, optimizada para analizar e indexar un gran volumen de datos en tiempo real
- Logstash: es una aplicación que permite recolectar, parsear y guardar logs
- Filebeat: se encarga de leer ficheros de un directorio específico y enviarlos a un servidor (el funcionamiento es parecido a rsyslog). Se debe configurar IP y puerto del servidor destino.
- Kibana: sirve para visualizar y explorar los datos que se encuentran dentro de Elasticsearch
- Redis: es una base de datos basado en el almacenamiento en tablas de hashes (clave/valor). En el ELK avanzado se lo utiliza como un buffer.

Puesta en funcionamiento y plataforma de instalación

Para la instalación de los programas se utilizó un software de virtualización llamado Docker. Esta plataforma proporciona una capa de abstracción a nivel de sistema operativo Linux lo que hace bastante flexible para instalar cualquier sistema operativo. Además está aislado del resto de los software evitando problemas de compatibilidad.

Docker crea una imagen a partir de un fichero llamado Dockerfile. Este fichero contiene las instrucciones para la instalación del software deseado (por ejemplo Logstash). También puede incluir las instrucciones para realizar las configuraciones que hagan falta, como bajar

otro software, copiar y pegar archivos de configuración desde el host, etc. Una vez creada la imagen, se pone en funcionamiento creando un contenedor, que tiene todo lo necesario para que los programas puedan funcionar dentro de forma aislada.

Elección de Docker como plataforma de instalación

Se eligió Docker a pedido del NOC ya que querían un esquema de instrucciones para la instalación de los programas de manera que pudiera ser reproducible y Docker les pareció el más indicado para este desarrollo.

Arquitecturas de ELK

Este stack tiene varios tipos de arquitecturas desde una simple que solo utiliza una instancia de Logstash y de Elasticsearch, hasta la de alta disponibilidad que contiene clusterización de Elasticsearch para que haya persistencia en los datos y siempre sean accesibles en caso que uno de los nodos del cluster caiga, también cuenta con varias instancias Logstash que cumple diferentes propósitos y una cola de mensajes para la alta disponibilidad. En [Escalabilidad de ELK](#) se encuentra el enlace de la página oficial. De esta manera se puede realizar el esquema de conexión que más se adecue al problema que se quiera resolver.

Elección de arquitectura

Para este proyecto se decidió realizar dos arquitecturas de ELK una simple y otra avanzada. Para probar el funcionamiento y configuración de ELK, se realizó una conexión básica del stack Elastic, esto permitió un mejor entendimiento de las configuraciones básicas que se requiere.

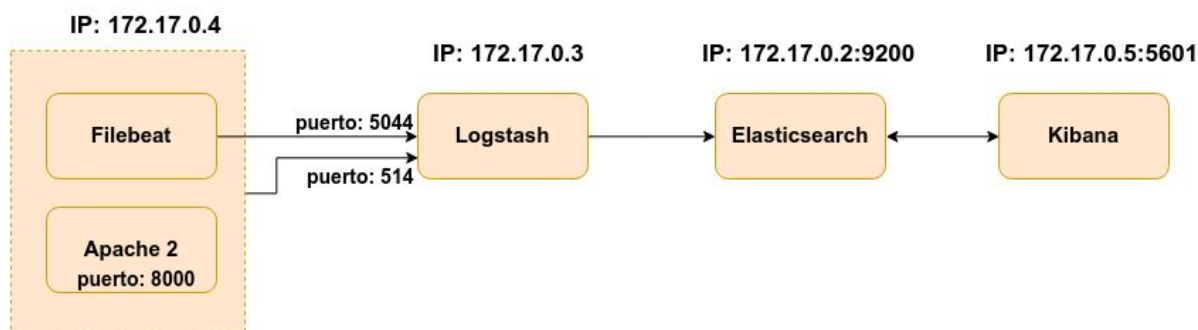
Mientras que la arquitectura de ELK avanzada cuenta con una cola de mensajes (Redis) y la posibilidad de clusterizar la base de datos Elasticsearch y Redis a futuro, dando persistencia y alta disponibilidad.

- ELK simple: consta de un Logstash, Elasticsearch y Kibana para la parte de recepción de los mensajes y un Filebeat y Rsyslog para generar los mensajes.
- ELK avanzado: consta de una instancia Logstash (Index), Redis, Elasticsearch, Kibana para la parte de recepción. También hay una instancia Logstash Shipper que es intermedia entre el emisor de mensajes y el receptor, esta puede recibir mensajes de varios emisores y aplicar filtros a cada uno y luego enviar los mensajes ya procesados al receptor.

A partir de ahora se verán las configuraciones de ambas arquitecturas.

Configuración de ELK simple

Esquema de conexión ELK simple



Una vez que se comprobó el correcto funcionamiento y se comprendieron las configuraciones se pasó al ELK avanzado.

Solo se van a mostrar las configuraciones, no se van a explicar porque son similares a las de ELK avanzado y ahí se explicará con más detalle.

En el directorio `/srv/docker-make/` de `docker-dev` están los archivos que contienen la configuración de cada programa:

- `elk-simple-elasticsearch-dev`
- `elk-simple-filebeat-dev`
- `elk-simple-kibana-dev`
- `elk-simple-logstash-dev`

Configuración de ELK básico

Elasticsearch

Dentro del directorio `elk-simple-elasticsearch-dev` se encuentran los archivos:

[elasticsearch.yml](#)

De este archivo lo relevante es :

- `cluster.name: primerCluster`
- `node.name: nodoPrueba`
- `network.host: 172.17.0.2`
- `http.port: 9200`

El archivo logging.yml quedó obsoleto a partir de la versión 5.0

Dockerfile

```
Dockerfile x
1 FROM elasticsearch
2
3 COPY logging.yml /usr/share/elasticsearch/config/
4
5 COPY elasticsearch.yml /usr/share/elasticsearch/config
6
7 RUN apt-get update && apt-get install -y nano
```

Filebeat

Dentro del directorio elk-simple-filebeat-dev se encuentra en los archivos:

filebeat.yml:

```
filebeat.yml x
1 filebeat:
2   prospectors:
3     -
4     paths:
5       - "/var/log/apache2/*.log"
6     document_type: apache
7
8 output:
9   logstash:
10     hosts: ["172.17.0.3:5044"]
11     index: filebeat
12     tls:
13       insecure: true
```

Los archivos 20-ufw.conf y 50-default.conf corresponden a la configuración de rsyslog

20-ufw.conf

```
20-ufw.conf x
1 # Log kernel generated UFW log messages to file
2 :msg,contains,"[UFW " /var/log/ufw.log
3
4 # Uncomment the following to stop logging anything that matches the last
   rule.
5 # Doing this will stop logging kernel generated UFW log messages to the
   file
6 # normally containing kern.* messages (eg, /var/log/kern.log)
7 #& ~
```

50-default.conf

```
50-default.conf x
1 # Default rules for rsyslog.
2 #
3 #                               For more information see rsyslog.conf(5) and /etc/
   rsyslog.conf
4 *.*                               @172.17.0.3:514
5 #
6 # First some standard log files.  Log by facility.
7 #
8 auth,authpriv.*                  /var/log/auth.log
9 *.*;auth,authpriv.none          -/var/log/syslog
10 #cron.*                          /var/log/cron.log
11 #daemon.*                        -/var/log/daemon.log
12 kern.*                           -/var/log/kern.log
13 #lpr.*                           -/var/log/lpr.log
14 mail.*                           -/var/log/mail.log
15 #user.*                          -/var/log/user.log
16
17 #
18 # Logging for the mail system.  Split it up so that
19 # it is easy to write scripts to parse these files.
20 #
21 #mail.info                        -/var/log/mail.info
22 #mail.warn                        -/var/log/mail.warn
23 mail.err                          /var/log/mail.err
24
25 #
26 # Logging for INN news system.
27 #
28 news.crit                         /var/log/news/news.crit
29 news.err                          /var/log/news/news.err
30 news.notice                       -/var/log/news/news.notice
31
```

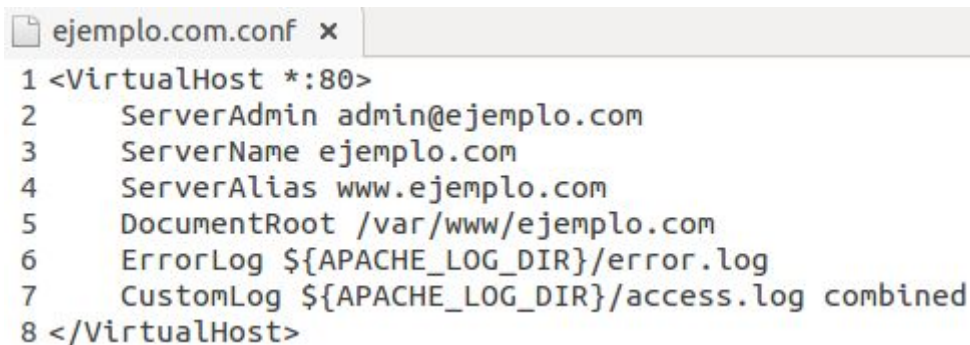
```

33 # Some "catch-all" log files.
34 #
35 #*.=debug;\
36 #     auth,authpriv.none;\
37 #     news.none;mail.none      -/var/log/debug
38 #*.=info;*.=notice;*.=warn;\
39 #     auth,authpriv.none;\
40 #     cron,daemon.none;\
41 #     mail,news.none           -/var/log/messages
42
43 #
44 # Emergencies are sent to everybody logged in.
45 #
46 *.emerg                        :omusrmsg:*
47
48 #
49 # I like to have messages displayed on the console, but only on a virtual
50 # console I usually leave idle.
51 #
52 #daemon,mail.*;\
53 #     news.=crit;news.=err;news.=notice;\
54 #     *.=debug;*.=info;\
55 #     *.=notice;*.=warn        /dev/tty8
56
57 # The named pipe /dev/xconsole is for the 'xconsole' utility.  To use it,
58 # you must invoke 'xconsole' with the '-file' option:
59 #
60 #   $ xconsole -file /dev/xconsole [...]
61 #
62 # NOTE: adjust the list below, or you'll go crazy if you have a reasonably
63 #       busy site..
64 #
65 daemon.*;mail.*;\
66     news.err;\
67     *.=debug;*.=info;\
68     *.=notice;*.=warn      |/dev/xconsole

```

Los archivos ejemplo.com.conf, hosts y index.html corresponde a la configuración de apache 2.

[ejemplo.com.conf](#)



```

ejemplo.com.conf x
1 <VirtualHost *:80>
2     ServerAdmin admin@ejemplo.com
3     ServerName ejemplo.com
4     ServerAlias www.ejemplo.com
5     DocumentRoot /var/www/ejemplo.com
6     ErrorLog ${APACHE_LOG_DIR}/error.log
7     CustomLog ${APACHE_LOG_DIR}/access.log combined
8 </VirtualHost>

```

hosts

```
hosts x
1 127.0.0.1      localhost
2 127.0.1.1      Bangho
3 172.17.0.3     ejemplo.com
4
5 # The following lines are desirable for IPv6 capable hosts
6 ::1           ip6-localhost ip6-loopback
7 fe00::0       ip6-localnet
8 ff00::0       ip6-mcastprefix
9 ff02::1       ip6-allnodes
10 ff02::2      ip6-allrouters
```

index.html

```
index.html x
1 <html>
2   <head>
3     <title>Bienvenido a Ejemplo.com!</title>
4   </head>
5   <body>
6     <h1>Exito! El Virtual Host ejemplo.com esta funcionando!</h1>
7   </body>
8 </html>
```

Dockerfile

```
Dockerfile x
1 FROM prima/filebeat
2
3 COPY filebeat.yml /filebeat.yml
4
5 COPY filebeat.yml /bin/filebeat.yml
6
7 RUN apt-get update && apt-get install -y curl
8
9 #####INSTALACION Y CONFIGURACION DE APACHE2#####
10
11 RUN apt-get install -y apache2
12
13 #####Configuracion de pagina ejemplo
14
15 RUN mkdir -p /var/www/ejemplo.com
16
17 COPY index.html /var/www/ejemplo.com/
18
19 RUN chown -R $USER:$USER /var/www/ejemplo.com/
20
21 RUN chmod -R 755 /var/www/
22
23 COPY ejemplo.com.conf /etc/apache2/sites-available/
24
25 RUN cd /etc/apache2/sites-available/ && a2ensite ejemplo.com.conf &&
    a2dissite 000-default.conf && a2dissite default-ssl.conf
26
27 COPY hosts /etc/hosts
28
29 #####Instalacion y configuracion rsyslog
30
31 RUN apt-get install -y rsyslog
32
33 COPY 20-ufw.conf /etc/rsyslog.d/
34
35 COPY 50-default.conf /etc/rsyslog.d/
36
37 #####Instalacion de screen
38
39 RUN apt-get install -y screen
40
```

Kibana

Dentro del directorio elk-simple-kibana-dev se encuentra en los archivos:

kibana.yml

```
kibana.yml x
1  server.host: "localhost"
2  server.port: 5601
3  server.name: "kibana"
4  elasticsearch.url: "http://172.17.0.2:9200"
```

Dockerfile

```
Dockerfile x
1 FROM kibana
2
3 RUN apt-get update && apt-get install -y curl
4
5 RUN curl -L -O https://download.elastic.co/beats/dashboards/beats-
  dashboards-1.1.0.zip
6
7 RUN apt-get -y install unzip && unzip beats-dashboards-*.zip
8
9 COPY load.sh /beats-dashboards-*/
10
11 RUN cd beats-dashboards-*/ && ./load.sh
```

Logstash

Dentro del directorio elk-simple-logstash-dev se encuentra en los archivos:

beats-input.conf

```
beats-input.conf x
1 input {
2   beats {
3     port => 5044
4     type => "logs"
5     ssl => true
6     ssl_certificate => "/etc/pki/tls/certs/filebeat.crt"
7     ssl_key => "/etc/pki/tls/private/filebeat.key"
8   }
9 }
```

syslog-input.conf

```
syslog-input.conf x
1 input {
2   tcp {
3     port => 514
4     type => syslog
5   }
6   udp {
7     port => 514
8     type => syslog
9   }
10 }
```

syslog-filter.conf

```
syslog-filter.conf x
1 filter {
2   if [type] == "syslog" {
3     grok {
4       match => { "message" => "%{SYSLOGTIMESTAMP:syslog_timestamp} %
5         {SYSLOGHOST:syslog_hostname} %{DATA:syslog_program}(?:\[%
6         {POSINT:syslog_pid}\])?: %{GREEDYDATA:syslog_message}" }
7       add_field => [ "received_at", "%{@timestamp}" ]
8       add_field => [ "received_from", "%{host}" ]
9     }
10    syslog_pri { }
11    date {
12      match => [ "syslog_timestamp", "MMM d HH:mm:ss", "MMM dd HH:mm:ss"
13    ]
14  }
15 }
```

output.conf

```
output.conf x
1 output {
2
3   elasticsearch {
4     hosts => ["172.17.0.2:9200"]
5     index => "%{[@metadata][beat]}-%{+YYYY.MM.dd}"
6     document_type => "%{[@metadata][type]}"
7   }
8   if [type] == "syslog"{
9     elasticsearch {
10      hosts => ["172.17.0.2:9200"]
11    }
12  }
13  stdout { codec => rubydebug }
14 }
```

Dockerfile

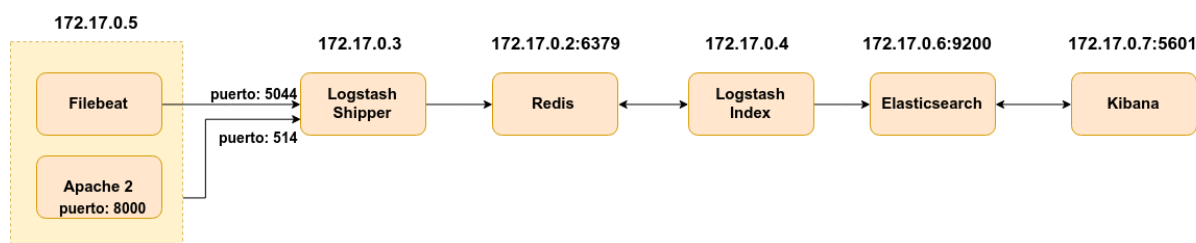
```
Dockerfile x
1 FROM logstash
2
3 ##### Instalacion de screen
4
5 RUN apt-get update
6
7 RUN apt-get install -y screen
8
9 #####Configuracion de certificados tls
10
11 RUN mkdir -p /etc/pki/tls/certs
12
13 RUN mkdir /etc/pki/tls/private
14
15 COPY openssl.cnf /etc/ssl/openssl.cnf
16
17 RUN cd /etc/pki/tls && openssl req -config /etc/ssl/openssl.cnf -x509 -
    days 3650 -batch -nodes -newkey rsa:2048 -keyout private/filebeat.key -
    out certs/filebeat.crt
18
19 ##### Se agrega beats-input.conf, output.conf y syslog.conf
20
21 COPY beats-input.conf /etc/logstash/conf.d/beats-input.conf
22
23 COPY output.conf /etc/logstash/conf.d/output.conf
24
25 COPY syslog-input.conf /etc/logstash/conf.d/syslog-input.conf
26
27 COPY syslog-filter.conf /etc/logstash/conf.d/syslog-filter.conf
28
29 #####FIN
```

Configuración de ELK Avanzado

En el directorio /srv/docker-make/ de docker-dev estan los archivos que contienen la configuración de cada programa:

- elk-elasticsearch-dev
- elk-filebeat-dev
- elk-kibana-dev
- elk-logstash-shipper-dev
- elk-logstash-index-dev
- elk-redis-dev

Esquema de conexión ELK avanzado



Configuración de ELK avanzado

Elasticsearch

Dentro de la carpeta elk-elasticsearch-dev hay un archivo llamado elasticsearch.yml en el cual se configuró:

- cluster.name: Cluster1
- node.name: nodo1
- network.host: 172.17.0.6
- http.port: 9200

La primera configuración corresponde al nombre de cluster a que va a pertenecer el nodo, la segunda corresponde al nombre que recibe el nodo.

La configuración network.host es el numero de IP que va a tener el nodo, mientras que http.port corresponde al puerto por el cual va a recibir la peticiones de Kibana y los mensajes de Logstash.

Dentro de este archivo estan las opciones de configuración para clusterizar Elasticsearch

El archivo logging.yml quedó obsoleto a partir de la versión 5.0

Filebeat

Dentro de la carpeta elk-filebeat-dev hay un archivo llamado filebeat.yml dentro del mismo esta el siguiente contenido:

```
filebeat.yml x
1 filebeat:
2   prospectors:
3     -
4     paths:
5       - "/var/log/apache2/*.log"
6     document_type: apache
7
8 output:
9   logstash:
10    hosts: ["172.17.0.3:5044"]
11    index: filebeat
12    tls:
13      insecure: true
14 |
```

En la línea 4 (paths), 5 y 6 se especifica la ruta donde se van a leer los logs en este caso el la ruta /var/log/apache2/*.log que está indicando que se lean todos los archivos de esa carpeta que sean de tipo log. El *document_type* especifica el encabezado para Elasticsearch.

A partir de la línea 8 en adelante son configuraciones perteneciente al envío de mensajes como *host* donde se especifica la en número de IP y puerto de destino (Logstash-Shipper). El campo *index* especifica la etiqueta para Elasticsearch. En *tls* (capa de puertos seguros) esta la configuracion para trabajar con este protocolo en este caso se optó por trabajar en modo inseguro porque es un desarrollo y no una puesta en producción.

Rsyslog:

Los archivos 20-ufw.conf y 50-default.conf corresponden a la configuración de rsyslog

20-ufw.conf

```
20-ufw.conf x
1 # Log kernel generated UFW log messages to file
2 :msg,contains,"[UFW " /var/log/ufw.log
3
4 # Uncomment the following to stop logging anything that matches the last
  rule.
5 # Doing this will stop logging kernel generated UFW log messages to the
  file
6 # normally containing kern.* messages (eg, /var/log/kern.log)
7 #& ~
```

50-default.conf

```
50-default.conf x
1 # Default rules for rsyslog.
2 #
3 #                               For more information see rsyslog.conf(5) and /etc/
  rsyslog.conf
4 *.*                               @172.17.0.3:514
5 #
6 # First some standard log files.  Log by facility.
7 #
8 auth,authpriv.*                  /var/log/auth.log
9 *.*;auth,authpriv.none          -/var/log/syslog
10 #cron.*                          /var/log/cron.log
11 #daemon.*                        -/var/log/daemon.log
12 kern.*                           -/var/log/kern.log
13 #lpr.*                           -/var/log/lpr.log
14 mail.*                           -/var/log/mail.log
15 #user.*                           -/var/log/user.log
16
17 #
18 # Logging for the mail system.  Split it up so that
19 # it is easy to write scripts to parse these files.
20 #
21 #mail.info                        -/var/log/mail.info
22 #mail.warn                        -/var/log/mail.warn
23 mail.err                          /var/log/mail.err
24
25 #
26 # Logging for INN news system.
27 #
28 news.crit                         /var/log/news/news.crit
29 news.err                          /var/log/news/news.err
30 news.notice                       -/var/log/news/news.notice
31
```

```
33 # Some "catch-all" log files.
34 #
35 #*.=debug;\
36 #     auth,authpriv.none;\
37 #     news.none;mail.none      -/var/log/debug
38 #*.=info;*.=notice;*.=warn;\
39 #     auth,authpriv.none;\
40 #     cron,daemon.none;\
41 #     mail,news.none           -/var/log/messages
42
43 #
44 # Emergencies are sent to everybody logged in.
45 #
46 *.emerg                        :omusrmsg:*
47
48 #
49 # I like to have messages displayed on the console, but only on a virtual
50 # console I usually leave idle.
51 #
52 #daemon,mail.*;\
53 #     news.=crit;news.=err;news.=notice;\
54 #     *.=debug;*.=info;\
55 #     *.=notice;*.=warn        /dev/tty8
56
57 # The named pipe /dev/xconsole is for the 'xconsole' utility. To use it,
58 # you must invoke 'xconsole' with the '-file' option:
59 #
60 # $ xconsole -file /dev/xconsole [...]
61 #
62 # NOTE: adjust the list below, or you'll go crazy if you have a reasonably
63 #       busy site..
64 #
65 daemon.*;mail.*;\
66     news.err;\
67     *.=debug;*.=info;\
68     *.=notice;*.=warn        |/dev/xconsole
```

En este archivo se especifica que los directorios de los log que se quieren enviar al servidor 172.17.0.13:514 (Logstash Shipper).

Apache 2:

Los archivos ejemplo.com.conf, hosts y index.html corresponde a la configuración de apache 2.

ejemplo.com.conf

```
ejemplo.com.conf x
1 <VirtualHost *:80>
2     ServerAdmin admin@ejemplo.com
3     ServerName ejemplo.com
4     ServerAlias www.ejemplo.com
5     DocumentRoot /var/www/ejemplo.com
6     ErrorLog ${APACHE_LOG_DIR}/error.log
7     CustomLog ${APACHE_LOG_DIR}/access.log combined
8 </VirtualHost>
```

Configuración de la ruta en la que se ubica la página de prueba y el puerto por el que se accede

hosts

```
hosts x
1 127.0.0.1      localhost
2 127.0.1.1      Bangho
3 172.17.0.3     ejemplo.com
4
5 # The following lines are desirable for IPv6 capable hosts
6 ::1           ip6-localhost ip6-loopback
7 fe00::0       ip6-localnet
8 ff00::0       ip6-mcastprefix
9 ff02::1       ip6-allnodes
10 ff02::2      ip6-allrouters
```

Configuración de DNS local (dejó de funcionar cuando se creó la imagen)

index.html

```
index.html x
1 <html>
2   <head>
3     <title>Bienvenido a Ejemplo.com!</title>
4   </head>
5   <body>
6     <h1>Exito! El Virtual Host ejemplo.com esta funcionando!</h1>
7   </body>
8 </html>
```

Texto que se muestra cuando se accede a la página de prueba.



Kibana

Dentro de la carpeta elk-kibana-dev hay un archivo llamado kibana.yml dentro del mismo esta el siguiente contenido:

```
kibana.yml x
1  server.host: "localhost"
2  server.port: 5601
3  server.name: "kibana"
4  elasticsearch.url: "http://172.17.0.2/:9200"
5
```

Se los campos son:

- server.host: se especifica el número de IP de Kibana
- server.port: se especifica el número de puerto de Kibana
- server.name: se especifica el nombre del servidor de Kibana para poder identificarlo con mas facilidad
- elasticsearch.url: es el número de IP y puerto de Elasticsearch donde Kibana realizará las consultas a dicha base de datos para obtener los logs.

Logstash Shipper

Dentro de la carpeta elk-logstash-shipper-dev hay unos archivos llamados beats-input.conf, syslog-input.conf, syslog-filter.conf y output.conf. Se analizan el contenido dentro de los mismos:

beats-input.conf:

```
beats-input.conf x
1 input {
2   beats {
3     port => 5044
4     type => "logs"
5     ssl => true
6     ssl_certificate => "/etc/pki/tls/certs/filebeat.crt"
7     ssl_key => "/etc/pki/tls/private/filebeat.key"
8   }
9 }
```

Este archivo corresponde a la configuración de entrada de los mensajes de Filebeat. Los campos que se pueden ver son:

- port: corresponde al puerto que se configura como entrada
- type: se especifica el tipo de mensaje. Es útil para búsquedas con Kibana
- ssl: se activa la conexión segura (en este caso no se usa porque Filebeat se lo configuro para trabajar en modo inseguro pero la activación es necesaria para el correcto funcionamiento de las dos partes)
- ssl_certificate: se especifica la ruta del certificado ssl
- ssl_key: se especifica la ruta de la ruta ssl

syslog-input.conf:

```
syslog-input.conf x
1 input {
2   tcp {
3     port => 514
4     type => syslog
5   }
6   udp {
7     port => 514
8     type => syslog
9   }
10 }
```

Este archivo corresponde a la configuración de entrada de los mensajes de syslog. Este protocolo utiliza tcp y udp. Los campos que se pueden ver son los mismos para ambos protocolos:

- port: es el puerto por donde se van a recibir los mensajes.
- type: se especifica el tipo de mensaje. Es útil para búsquedas con Kibana

syslog-filter.conf:

```
syslog-filter.conf x
1 filter {
2   if [type] == "syslog" {
3     grok {
4       match => { "message" => "%{SYSLOGTIMESTAMP:syslog_timestamp} %{SYSLOGHOST:syslog_hostname} %
5 {DATA:syslog_program}(?:\[ %{POSINT:syslog_pid}\])?: %{GREEDYDATA:syslog_message}" }
6       add_field => [ "received_at", "%{@timestamp}" ]
7       add_field => [ "received_from", "%{host}" ]
8     }
9     syslog_pri { }
10    date {
11      match => [ "syslog_timestamp", "MMM d HH:mm:ss", "MMM dd HH:mm:ss" ]
12    }
13  }
```

Este archivo corresponde al filtro que aplica Logstash para parsear los mensajes de syslog en formato syslog. Lo primero que hace es consultar que tipo de mensaje es. Si es syslog se le aplica el filtro grok. Esta configuración se sacó de la página oficial de Logstash

<https://www.elastic.co/guide/en/logstash/current/config-examples.html>

(Processing Syslog Messages)

output.conf:

```
output.conf x
1 output {
2   if [type] == "syslog"{
3     redis {
4       host => ["172.17.0.2:6379"]
5       key => "logstash"
6       shuffle_hosts => true
7       data_type => "list"
8     }
9   }else {
10    redis {
11      host => ["172.17.0.2:6379"]
12      key => "filebeat"
13      shuffle_hosts => true
14      data_type => "list"
15    }
16  }
17  stdout { codec => rubydebug }
18 }
```

Este archivo corresponde a la configuración de salida de Logstash Shipper. En este caso se utilizó un bloque de decisión para separar por tipo de mensaje. Lo único que cambia en este caso es el campo key, ya que van al mismo destino.

Los campos que se pueden ver dentro de redis son:

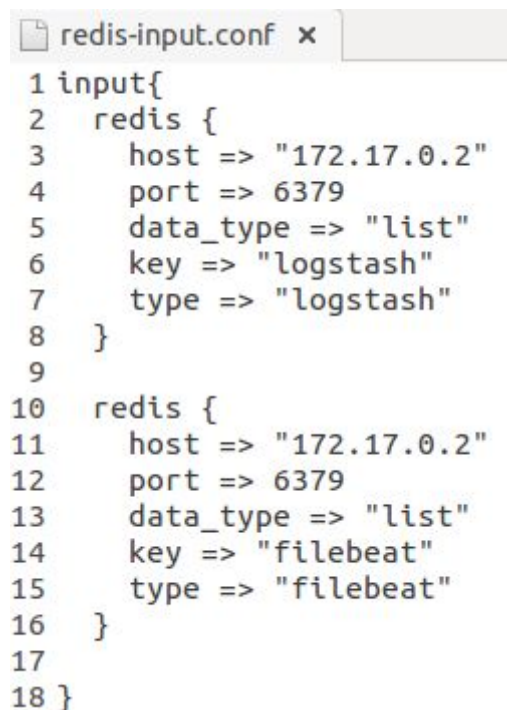
- host: es la IP y puerto del servidor Redis
- key: es el nombre o lista del canal en Redis
- shuffle_hosts: habilita la mezcla de lista durante el inicio de Logstash
- data_type: se especifica si va a ser una lista o un canal.

Por otro lado esta stdout que es para visualizar por pantalla los mensajes que se envían

Logstash Index

Dentro de la carpeta elk-logstash-index-dev hay unos archivos llamados redis-input.conf y output.conf. Se analizan el contenido dentro de los mismos:

redis-input.conf:



```
1 input{
2   redis {
3     host => "172.17.0.2"
4     port => 6379
5     data_type => "list"
6     key => "logstash"
7     type => "logstash"
8   }
9
10  redis {
11    host => "172.17.0.2"
12    port => 6379
13    data_type => "list"
14    key => "filebeat"
15    type => "filebeat"
16  }
17
18 }
```

Este archivo corresponde a la configuración de entrada de Logstash Index. Los mensajes se obtienen haciendo consultas a redis. Hay configurada dos entradas para el mismo redis, esto se debe a que se extraen dos tipo de mensajes para consulta.

Los campos de cada uno son:

- host: IP del servidor redis
- port: puerto de Redis
- data_type: se especifica si se trata de una lista o un canal
- key: es el nombre o lista del canal en Redis
- type: esto agrega un campo type a los mensajes que entran. Si el mensaje ya llegó con un campo type este no se reemplaza. En este caso el uso de este campo no es útil.

output.conf:

```
output.conf x
1 output {
2
3   if [type]=="syslog"{
4     elasticsearch {
5       hosts => ["172.17.0.6:9200"]
6       index => "syslog-%{+YYYY.MM.dd}"
7       document_type => "syslog"
8     }
9   }
10 }
11
12 if [type]=="apache"{
13   elasticsearch {
14     hosts => ["172.17.0.6:9200"]
15     index => "apache-%{+YYYY.MM.dd}"
16     document_type => "apache"
17   }
18 }
19 stdout { codec => rubydebug }
20 }
```

Los bloques de decisión separan los mensajes según el tipo. La información del campo type se obtiene del mensaje. Estos son enviados al mismo servidor Elasticsearch con distintos index. Los campos para Elasticsearch son:

- hosts: IP y puerto del servidor Elasticsearch
- index: este índice será la manera por la que se va a buscar los mensajes cuando se realicen las consultas a Elasticsearch.
- document_type: es para identificar el tipo de documento. Si no se especifica se dice que tipo log

Redis

Dentro de la carpeta elk-redis-dev hay un archivo llamado redis.conf en el cual se configuró:

- port 6379

Este es el puerto por el cual se van a hacer las consultas. La IP en este caso es la que le asigna docker al contenedor (por ejemplo 172.17.0.2)

Creacion de Imagenes de Docker

En el directorio /srv/docker-make/ de docker-dev estan los archivos Dockerfiles de cada programa:

- elk-elasticsearch-dev
- elk-filebeat-dev
- elk-kibana-dev
- elk-logstash-index-dev
- elk-logstash-shipper-dev
- elk-redis-dev

A continuación se analiza brevemente los Dockerfiles de cada carpeta:

Elasticsearch (elk-elasticsearch-dev)

A screenshot of a code editor showing a Dockerfile. The tab is labeled 'Dockerfile x'. The content of the file is as follows:

```
1 FROM elasticsearch
2
3 COPY logging.yml /usr/share/elasticsearch/config/
4
5 COPY elasticsearch.yml /usr/share/elasticsearch/config
```

En la primera línea se indica qué programa tiene que descargar del repositorio. En la segunda y tercera se copian los archivos de configuración desde el host a la imagen que crea Docker.

Filebeat (elk-filebeat-dev)

```
Dockerfile x
1 FROM prima/filebeat
2
3 COPY filebeat.yml /filebeat.yml
4
5 COPY filebeat.yml /bin/filebeat.yml
6
7 RUN apt-get update && apt-get install -y curl
8
9 #####INSTALACION Y CONFIGURACION DE APACHE2#####
10
11 RUN apt-get install -y apache2
12
13 #####Configuracion de pagina ejemplo
14
15 RUN mkdir -p /var/www/ejemplo.com
16
17 COPY index.html /var/www/ejemplo.com/
18
19 RUN chown -R $USER:$USER /var/www/ejemplo.com/
20
21 RUN chmod -R 755 /var/www/
22
23 COPY ejemplo.com.conf /etc/apache2/sites-available/
24
25 RUN cd /etc/apache2/sites-available/ && a2ensite ejemplo.com.conf &&
    a2dissite 000-default.conf && a2dissite default-ssl.conf
26
27 #####Instalacion y configuracion rsyslog
28
29 RUN apt-get install -y rsyslog
30
31 COPY 20-ufw.conf /etc/rsyslog.d/
32
33 COPY 50-default.conf /etc/rsyslog.d/
34
35 #####Instalacion de screen
36
37 COPY hosts /etc/hosts
38
39 RUN apt-get install -y screen
```

Al principio es similar al anterior, en la primera línea se indica qué programa tiene que descargar del repositorio. En la segunda y tercera se copian los archivos de configuración desde el host a la imagen que crea Docker.

Una vez instalado Filebeat se baja la plantilla de Filebeat para Elasticsearch desde <https://gist.github.com/thisismitch/3429023e8438cc25b86c/raw/d8c479e2a1adcea8b1fe86570e42abab0f10f364/filebeat-index-template.json> y se la envía a Elasticsearch (para esto Elasticsearch debe estar funcionando) con el comando:

```
curl -XPUT 'http://172.17.0.6:9200/_template/filebeat' -d@filebeat-index-template.json
```

La dirección IP y puerto corresponde a Elasticsearch (en este caso es 172.17.0.6:9200)
El resto son aplicaciones extra como apache2 y rsyslog

Kibana (elk-kibana-dev)

```
Dockerfile x
1 FROM kibana
2
3 RUN apt-get update && apt-get install -y curl
4
5 RUN curl -L -O https://download.elastic.co/beats/dashboards/beats-
  dashboards-1.1.0.zip
6
7 RUN apt-get -y install unzip && unzip beats-dashboards-*.zip
8
9 COPY load.sh /beats-dashboards-*/
10
11 RUN cd beats-dashboards-*/ && ./load.sh
```

Después de la instalación de Kibana se bajan el dashboard para beat desde <https://download.elastic.co/beats/dashboards/beats-dashboards-1.1.0.zip> y se extrae los archivos y luego se ubica en la carpeta (cd beats-dashboards-*/). Previamente editar el script load.sh con algun editor de texto. En el campo ELASTICSEARCH=<http://localhost:9200> cambiar la dirección por en verdadero numero de IP y puerto de Elasticsearch (por ejemplo ELASTICSEARCH=<http://172.17.0.6:9200>). Luego se reemplaza el archivo load.sh por el que se modifico.

Redis (elk-redis-dev)

```
Dockerfile x
1 FROM redis
2
3 COPY redis.conf /usr/local/etc/redis/redis.conf
4
```

En la primera línea se indica qué programa tiene que descargar del repositorio. Luego se copia el archivo de configuración para redis

Logstash (elk-logstash-index-dev)

```
Dockerfile x
1 FROM logstash
2
3 #####Instalacion de screen
4
5 RUN apt-get update
6
7 RUN apt-get install -y screen
8
9 #####Configuracion de certificados tls
10
11 RUN mkdir -p /etc/pki/tls/certs
12
13 RUN mkdir /etc/pki/tls/private
14
15 COPY openssl.cnf /etc/ssl/openssl.cnf
16
17 RUN cd /etc/pki/tls && openssl req -config /etc/ssl/openssl.cnf -x509 -
    days 3650 -batch -nodes -newkey rsa:2048 -keyout private/filebeat.key -
    out certs/filebeat.crt
18
19 ##### Se agrega redis-input.conf y output.conf
20
21 COPY redis-input.conf /etc/logstash/conf.d/redis-input.conf
22
23 COPY output.conf /etc/logstash/conf.d/output.conf
```

En la primera línea se indica qué programa tiene que descargar del repositorio. Luego cada sección se explica brevemente.

Logstash (elk-logstash-shipper-dev)

```
Dockerfile x
1 FROM logstash
2
3 #####Instalacion screen
4
5 RUN apt-get update
6
7 RUN apt-get install -y screen
8
9 #####Configuracion de certificados tls
10
11 RUN mkdir -p /etc/pki/tls/certs
12
13 RUN mkdir /etc/pki/tls/private
14
15 COPY openssl.cnf /etc/ssl/openssl.cnf
16
17 RUN cd /etc/pki/tls && openssl req -config /etc/ssl/openssl.cnf -x509 -
    days 3650 -batch -nodes -newkey rsa:2048 -keyout private/filebeat.key -
    out certs/filebeat.crt
18
19 ##### Se agrega beats-input.conf, output.conf y syslog.conf
20
21 COPY beats-input.conf /etc/logstash/conf.d/beats-input.conf
22
23 COPY output.conf /etc/logstash/conf.d/output.conf
24
25 COPY syslog-input.conf /etc/logstash/conf.d/syslog-input.conf
26
27 COPY syslog-filter.conf /etc/logstash/conf.d/syslog-filter.conf
```

Es similar a Logstash Index con la diferencia que agrega distintos archivos de configuración a la imagen de docker

Configuraciones de Arranque

A continuación se verá los ficheros de arranque para cada aplicación. Dichos fichero se encuentran en el directorio /srv/docker-run/ de docker-dev. Se programaron con script bash:

- elk-elasticsearch-dev
- elk-filebeat-dev
- elk-kibana-dev
- elk-logstash-shipper-dev
- elk-logstash-index-dev
- elk-redis-dev

Elasticsearch (elk-elasticsearch-dev)

```
IMAGE=hub.psi.unc.edu.ar/dev/elk-elasticsearch-dev
docker run -p 9200:9200 $IMAGE
```

Este script es bastante simple, sólo crea un contenedor docker de la imagen ya creada llamada como el valor que se le asignó a IMAGE. También publica el puerto 9200 mapeando con el 9200 del host.

Filebeat (elk-filebeat-dev)

```
TAG=elk-filebeat-dev
```

```
.  
.
.
```

```
IMAGE=hub.psi.unc.edu.ar/dev/elk-filebeat-dev
docker run -ti --rm \
  --name $TAG \
  --privileged=true \
  -h "$TAG" \
  -p 8000:80 \
  $IMAGE /bin/bash -c "
    service apache2 start;
    curl -O https://gist.githubusercontent.com/
thisismitch/3429023e8438cc25b86c/raw/
d8c479e2a1adcea8b1fe86570e42abab0f10f364/filebeat-index-template.json;
    curl -XPUT 'http://172.17.0.6:9200/_template/filebeat' -
d@filebeat-index-template.json;
    service rsyslog restart;
    /bin/bash
"
```

Este script crea una imagen de Filebeat con los parámetros:

- ti: conecta un psedo-terminal a contenedor
- rm: elimina el contenedor si existe
- name: nombre del contenedor
- privileged: activa o desactiva los privilegios del contenedor
- h: nombre del host del contenedor
- p: publica el valor del puerto del contenedor

Luego de arrancar el contenedor se le pasan los parámetros bash, para que los empiece a ejecutar:

1. service apache2 start;
2. curl -XPUT 'http://172.17.0.6:9200/_template/filebeat'
-d@filebeat-index-template.json;
3. service rsyslog restart;

El primero inicia apache 2.

El segundo se le pasa la plantilla json de Filebeat a Elasticsearch.

El tercero reinicia rsyslog.

Kibana (elk-kibana-dev)

```
TAG=elk-kibana-dev
```

```
.  
.  
.
```

```
IMAGE=hub.psi.unc.edu.ar/dev/elk-kibana-dev  
docker run --name $TAG -e ELASTICSEARCH_URL=http://172.17.0.6:9200 -p  
5601:5601 -d $IMAGE
```

El nuevo parámetro es -e el cual se usa para especificar asignación de variables. El -d es para que sea un demonio.

Logstash (elk-logstash-index-dev)

```
TAG=elk-logstash-index-dev
```

```
.  
.  
.
```

```
IMAGE=hub.psi.unc.edu.ar/dev/elk-logstash-index-dev
docker run -ti --rm \
  --name $TAG \
  --privileged=true \
  -h "$TAG" \
  $IMAGE /bin/bash -c "
    logstash -f /etc/logstash/conf.d/;
    /bin/bash
  "
```

Este script es parecido al de Filebeat, solo que cambia los comando de arranque.

- logstash -f /etc/logstash/conf.d/;

Este comando hacer una arranque de Logstash forzando a que use la configuración de la ruta especificada. Ahí están los ficheros de configuración.

Logstash (elk-logstash-shipper-dev)

```
TAG=elk-logstash-shipper-dev
.
.
.
IMAGE=hub.psi.unc.edu.ar/dev/elk-logstash-shipper-dev
docker run -ti --rm \
  --name $TAG \
  --privileged=true \
  -h "$TAG" \
  -p 5044:5044 \
  -p 514:514 \
  $IMAGE /bin/bash -c "
    logstash -f /etc/logstash/conf.d/;
    /bin/bash
  "
```

Es igual al script de Logstash Index solo cambia los nombres de etiquetas y de imagen

Redis (elk-redis-dev)

```
TAG=elk-redis-dev
```

```
.
.
.
```

```
IMAGE=hub.psi.unc.edu.ar/dev/elk-redis-dev
docker run -ti --rm \
  --name $TAG \
  --privileged=true \
  -h "$TAG" \
  -v /etc/localtime:/etc/localtime:ro \
  -p 6379:6379 \
  $IMAGE /bin/bash -c "
    redis-server /usr/local/etc/redis/redis.conf &
    /bin/bash
"
```

El script es parecido al de los Logstash, solo cambia los parámetros de arranque.

- redis-server /usr/local/etc/redis/redis.conf

Arranca a redis utilizando la configuración del destino especificada.

Se adjunta junto a este informe el documento creado por Muñoz Gabriel llamado “Nuevo docker”, el cual explica cómo crear una imagen de Docker y cómo ejecutar la misma en el servidor docker-dev.

Instrucciones de uso ELK

Condiciones de funcionamiento

- Se debe respetar el orden de ejecución, para no tener problemas con los numero de IP que se les asigna a los contenedores.
- Ninguna contenedor se debe estar ejecutando antes de arrancar el ELK.

ELK avanzado

En el directorio /srv/docker-make/ de docker-dev estan los archivos que contienen la configuración de cada programa:

- elk-elasticsearch-dev
- elk-filebeat-dev
- elk-kibana-dev
- elk-logstash-index-dev
- elk-logstash-shipper-dev
- elk-redis-dev

Para iniciar el funcionamiento de ELK

1. Ir al directorio `cd /srv/docker-run/`
2. Ejecutar `./elk-redis-dev`
3. Ejecutar `./elk-logstash-shipper-dev`
4. Ejecutar `./elk-logstash-index-dev`
5. Ejecutar `./elk-filebeat-dev`

6. Ejecutar `./elk-elasticsearch-dev`

7. Ejecutar `./elk-kibana-dev`

Por motivos de asignación de IPs que hace docker no se utiliza el scrip ARRANCAR para iniciar ELK

Un vez que arranco ELK

Se puede comenzar a enviar mensajes syslog a la IP 172.17.0.3 puerto 514 que pertenece a Logstash Shipper.

Para testear el funcionamiento del stack ELK está el contenedor Filebeat que tiene instalado: Filebeat, rsyslog y Apache 2. Este genera mensajes syslog y beat.

Primero hay que acceder al contenedor Filebeat con:

```
screen -r
```

Luego elegir el screen XXXX.elk-filebeat-dev (por ejemplo `screen -r 21322.elk-filebeat-dev`)

Para enviar mensaje syslog basta con reiniciar la aplicación ejecutando:

```
service rsyslog restart
```

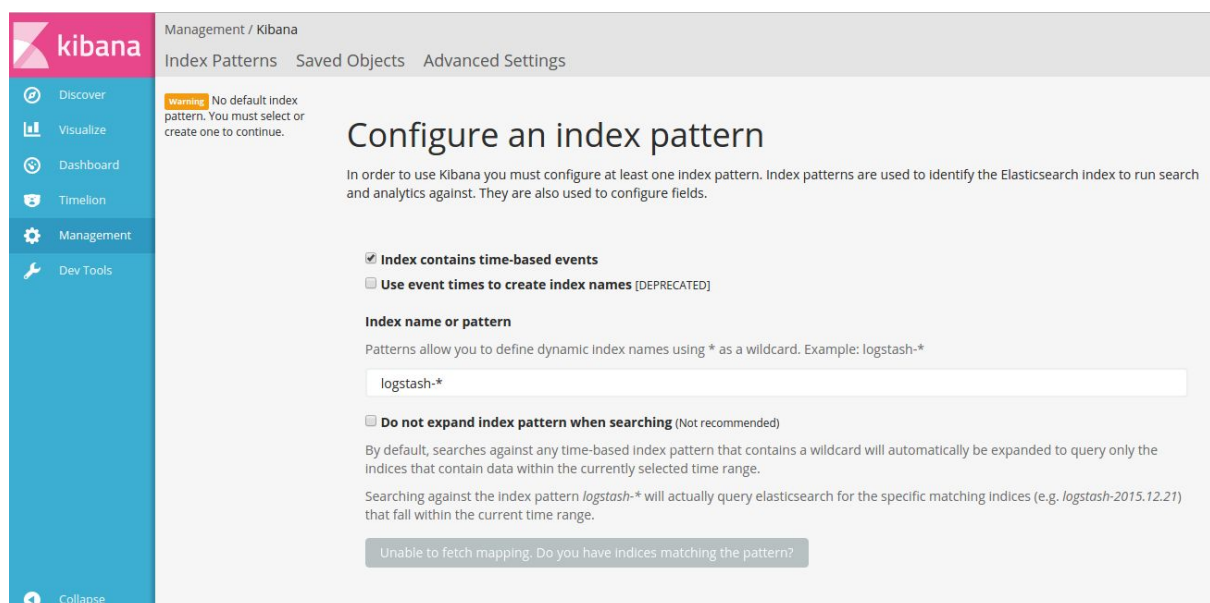
Para enviar mensajes beats se activa filebeat con el en comando:

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

De esta manera se activa el envío de logs de apache2 (`/var/log/apache2/*.log`). Cada vez que se realice una consulta a la página de prueba <http://localhost:8000/> o se produzcan errores.

Visualizar los mensajes recibidos

Los logs guardados en Elasticsearch se consulta mediante Kibana desde un browser (por ejemplo firefox). Se ingresa <http://localhost:5601/>, lo primero que se visualiza es:



En el campo Index name or pattern se puede buscar con índice o patrón para este caso escribiendo syslog-* se leen los mensajes de rsyslog (logs del sistema) si se escribe apache-* se ven los mensajes recibidos de filebeat (logs de apache)

Index name or pattern

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

Ir al campo Time-field name y seleccionar: @timestamp

Time-field name ⓘ refresh fields

Create

Luego presionar el botón Create. Va a aparecer la siguiente pantalla

Management / Kibana / Indices

Index Patterns Saved Objects Advanced Settings

+ Add New

★ syslog-*

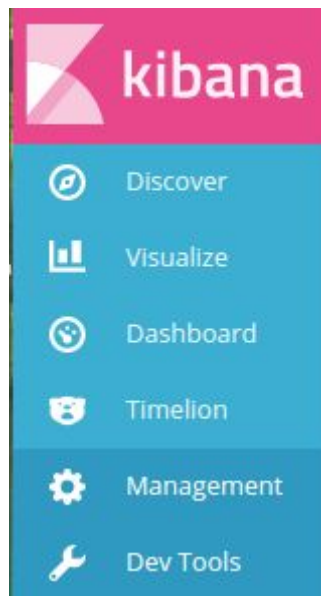
This page lists every field in the **syslog-*** index and the field's associated core type as recorded by Elasticsearch. While this list allows you to view the core type of each field, changing field types must be done using Elasticsearch's [Mapping API](#).

Filter

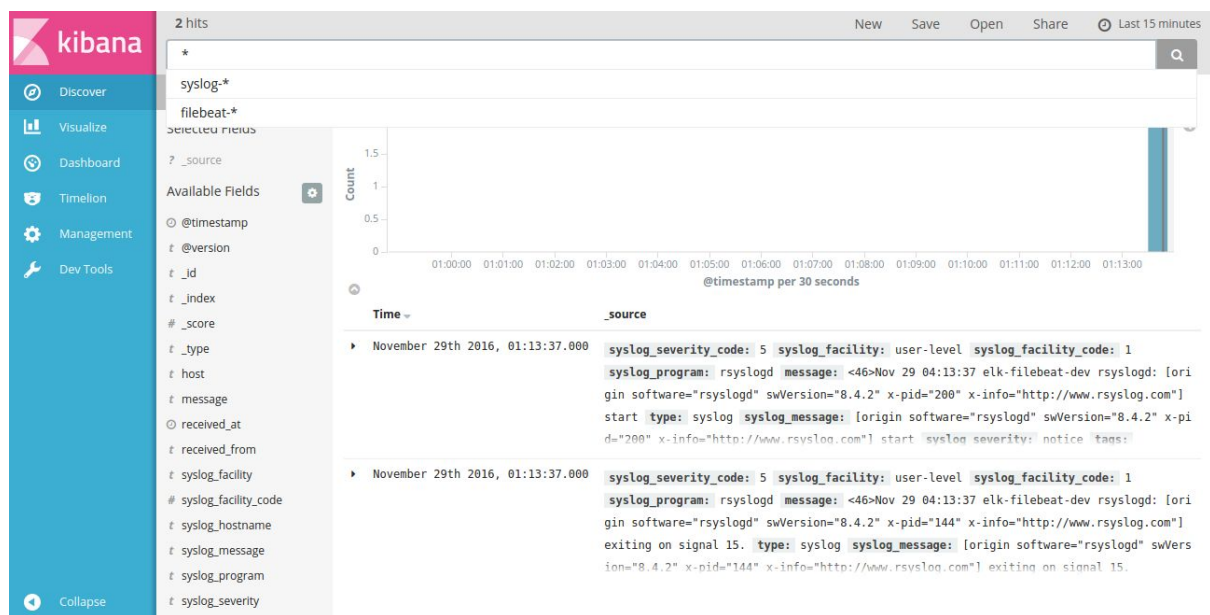
Fields (31) Scripted fields (0)

name	type	format	searchable	aggregatable	analyzed	controls
syslog_program	string		✓		✓	✎
type	string		✓		✓	✎
syslog_severity	string		✓		✓	✎
syslog_timestamp.keyword	string		✓	✓		✎
received_from	string		✓		✓	✎
syslog_timestamp	string		✓		✓	✎
type.keyword	string		✓	✓		✎
@version	string		✓		✓	✎

Seleccionar la pestaña: Discover

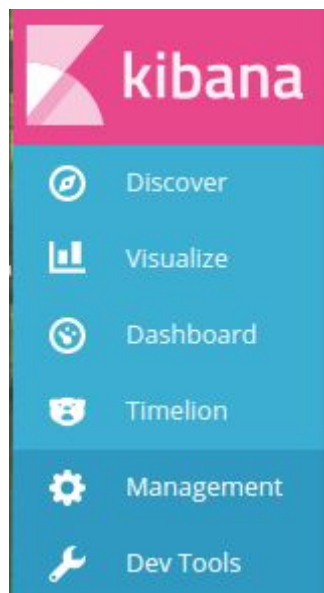


Lo cual no mostrará los mensajes almacenados en Elasticsearch:

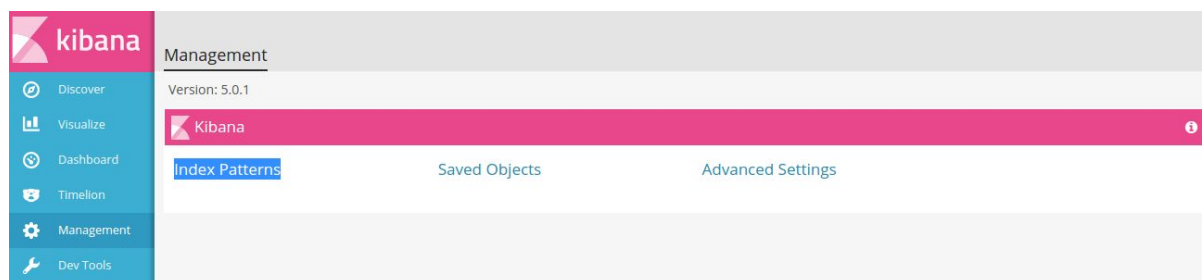


En la imagen se puede ver los mensajes de syslog.

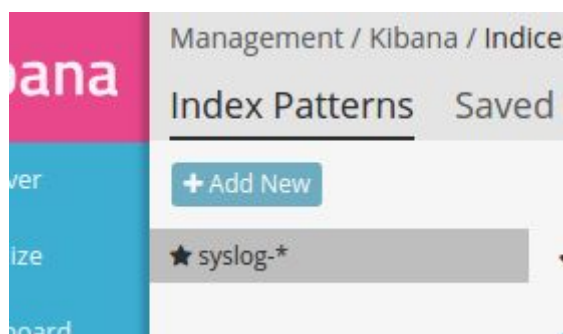
Luego si se desea buscar otro patrón o index ir a la pestaña Management:



Luego ir a Index Patterns:



Presionar el botón +Add New y nos va aparecer la pantalla de inicio nuevamente para buscar otro patrón o index. Como puede ser apache.



Consideraciones a Futuro

Tanto Redis como Elasticsearch permiten clusterización, en este trabajo no se llegó a desarrollar por la complejidad de la implementación y la falta de tiempo sobre todo. También si se desea seguir trabajando con Docker para la etapa de producción, se va a tener que configurar la imagen de Redis y Elasticsearch para agregar volúmenes de datos y así poder tener los datos persistentes, en caso de que los contenedores de dichos programa se caigan.

Cerca de la etapa final del proyecto se decidió poner a prueba el ELK (ya funcionando en una pc particular) en el servidor docker-dev de la prosecretaría. Esto llevó a bajar las imágenes de todos los Dockers nuevamente en docker-dev. Esto causó que se baje las nuevas versiones del stack de Elastic, lo que provocó problemas en la creación de la imagen de Logstash (ya no era necesario bajar los plugin en esta versión). En Elasticsearch no fue muy grave los cambio solo aumentó considerablemente los requerimientos de memoria RAM. En una pc de 4Gb (pc donde se comenzó el desarrollo) ya no podía funcionar con la fluidez que tenía con la versión anterior. El cambio de versión fue de 2.4 a 5.0 para la línea Elastic. La conclusión de esto es que en las próximas actualizaciones se debería tener en cuenta que las antiguas configuraciones pueden llegar a tener problemas.

A continuación se ve algunos requerimientos que se pueden agregar a futuro

Generales

- Base de datos en servidor independiente
- Registros se guardan por un tiempo a determinar en el servidor
- Registros se guardarán eternamente en la nube
- Archivos locales con rotación diaria o por tamaño según el caso
- Comprobar conexión de equipos (ping logger)

Infraestructura

Notificaciones de eventos

Redes

El análisis de los logs puede ser manual y automático para generar alertas y/o acciones
Analizar logs de los equipos, notificar y realizar estadísticas de los problemas.

- Equipos de red
 - Cambios de topología, mensajes STP (correlacionar logs para ver cómo se extiende los mensajes de cambio de topología)
 - Caídas excesivas de los puertos (definir nº de caídas normales)
 - Intentos de acceso no autorizados
 - Problemas con puertos libres (UDP)
 - Tormentas multicast/broadcast (correlacionar logs)
 - Bloqueo de puertos

-
- Protocolos que soportan los equipos de red
 - SNMP / SNMP Traps
 - Scripts de Bash u otro lenguaje

Soporte

Reporte de acceso de logs a PC de la UNC de aplicaciones como Teamviewer.
Reporte de acceso a paginas de adultos fuera del horario de trabajo.

Información adicional

Información sobre volúmenes para Elasticsearch

<https://www.elastic.co/blog/how-to-make-a-dockerfile-for-elasticsearch>
https://hub.docker.com/_/elasticsearch/

Información sobre cluster para Elasticsearch

<https://www.elastic.co/guide/en/elasticsearch/guide/current/distributed-cluster.html>
<https://www.elastic.co/guide/en/elasticsearch/reference/current/modules-cluster.html>

Información sobre volúmenes para Redis

https://hub.docker.com/_/redis/ (apartado **start with persistent storage**)

Información sobre cluster para Redis

<https://redis.io/topics/cluster-spec>
<https://redis.io/topics/cluster-tutorial>

Escalabilidad de ELK

<https://www.elastic.co/guide/en/logstash/current/deploying-and-scaling.html>