### COMANDOS DESACTUALIZADOS, PERO DOCUMENTACIÓN CORRECTA

### MAS REFERENCIAS AL FINAL DE DOCUMENTO

Instalacion de docker en una EC2:

Dockers:  
-> Intalacion de docker en AWS (documentación oficial)

<http://docs.aws.amazon.com/AmazonECS/latest/developerguide/docker-basics.html>

-> videos de Dockers (youtube):

https://www.youtube.com/channel/UC0870Uo4lVU-\_JEEY5LWu-A/playlists

-> creación de Dockfiles

-run

-volume

-ADD/COPY

ENV :

ENV http\_proxy 'http://10.110.8.42:8080'

ENV https\_proxy 'http://10.110.8.42:8080'

ENV HTTP\_PROXY 'http://10.110.8.42:8080'

ENV HTTPS\_PROXY 'http://10.110.8.42:8080'

-> Uso de dockers oficiales:

Elasticsearch: (<https://hub.docker.com/_/elasticsearch/>)

Comando para ejecutarlo:

docker run --name some-elasticsearch -v /home/ec2-user/AWS/Elasticsearch/ElasticsearchData:/usr/share/elasticsearch/data elasticsearch -Des.insecure.allow.root=true

Únicamente se configure la ruta de Elasticsearch que quieres que se compartan los volúmenes, sería la ruta que te permite accede a la información de los nodos de elasticsearch.

También se permite cambiar la configuración de Elasticsearch pero a notros nos va bien la predeterminada: (-v "$PWD/config":/usr/share/elasticsearch/config)

Kibana: (<https://hub.docker.com/_/kibana/>)

Commandos para ejecutarlo:

docker run --name some-kibana --link some-elasticsearch:elasticsearch -p 5601:5601 -d kibana

El link une el elasticsearch que necesita kibana con “elasticsearch” que tiene definida la imagen en el interior (some-elasticsearch es el nombre del docker y “elasticsearch” ese el nombre/puerto que tiene definido kibana para recibir información por allí). No sé exactamente que como funciona por el interior pero la función de link es esa.

Y “–p” es el puerto para ver la web; la “-d” únicamente es para que no se quede el terminal con la salida de logs del docker.

Logstash: (<https://hub.docker.com/_/logstash/>)

Commando para ejectuarlo:

docker run -it --name logstash --rm -v /home/ec2-user/AWS/logstash:/config-dir logstash logstash -f /config-dir/logstash.conf

Logstash require de mucha más configuración para usarlo. Usaremos la imagen oficial por comodidad pero habremos de configurarlo desde el conf. Primero tendremos que volcar toda la información que quiere recibir el logstash en una carpeta en este caso: “/home/ec2-user/AWS/logstash” dentro de la carpeta incluiremos la configuración del logstash “logstash.conf” , también los logs en “/home/ec2-user/AWS/logstash/logs” y la carpeta de Patters para poder leer lo, al tener una sola entrada al docker si queremos recibir desde un fichero de logs es la manera más fácil.

La configuración de logstash(logstash.conf):

input {stdin {type => "stdin-type"}

file {

type => "syslog-ng"

# Wildcards work, here :)

path => [ "/config-dir/logs/\*.log"]

start\_position => "beginning"

}

}

filter {

grok {#Importacion de los patrones REGEX.

patterns\_dir =>"/config-dir/Patterns"

#Linea de [FIN] con estado OK

match =>["message", "%{TIMESTAMP:timestamp} %{TIME\_ZONE:time\_zone} \[%{GENERIC\_USER:Id\_Usuario}\] \[%{GENERIC\_WORD:Id\_Terminal}\] \[%{GENERIC\_WORD:Id\_Peticion}\] \[%{MODE:Modo}\] \[%{EVENT\_LINE:Tipo\_evento}\] \[%{STATUS:Resultado}\] \[Servicio\: %{GENERIC\_WORD:Servicio}\] \[Metodo\: %{GENERIC\_WORD:Metodo}\] \[Tiempo\_peticion\: %{GENERIC\_NUMBER:Tiempo\_peticion:float} ms\]"]

#Linea de [INICIO] generica

match =>["message", "%{TIMESTAMP:timestamp} %{TIME\_ZONE:time\_zone} \[%{GENERIC\_USER:Id\_Usuario}\] \[%{GENERIC\_WORD:Id\_Terminal}\] \[%{GENERIC\_WORD:Id\_Peticion}\] \[%{MODE:Modo}\] \[%{EVENT\_LINE:Tipo\_evento}\] \[Servicio\: %{GENERIC\_WORD:Servicio}\] \[Metodo:\ %{GENERIC\_WORD:Metodo}\]"]}

if([Tipo\_evento] =~"(FIN)") { mutate{

add\_field =>["Parametros", "Sin parametros o pendiente de actualizarlos"]

add\_field =>["Inicio\_peticion", "Pendiente de actualizacion"]}}

else { mutate{ add\_field =>["Veces\_procesada", 0]} }

}

output {

elasticsearch { hosts => ["172.17.0.2:9200"] }

}

En el input se pondrá la manera de recibir los logs, podría ser en sockets etc pero nosotros cogeremos un fichero de logs que en el docker está referenciado a: path => [ "/config-dir/logs/\*.log"] -> esto en la máquina de Amazon esta “/home/ec2-user/AWS/logstash/logs/”

Después hay que usar los patterns que están en la carpeta patterns\_dir =>"/config-dir/Patterns" \[%{GENERIC\_USER:Id\_Usuario}]el nombre de la variable es ID\_Usuario y el tipo GENERIC\_USER. Para añadir nuevos parámetros se usa: mutate{ add\_field =>["Veces\_procesada", 0]}

El output se introduce la ip en la cual está la entrada de elasticsearch 172.17.0.2:9200 usando el plugin que ya tiene logstash por defecto poniendo: elasticsearch { hosts =>.

Correr els logs en la carpeta.

REFERENCIAS:

DOCKERFILES: (En principi al final no fa falta utilizar DOCKERFILES per fer build propies)

<http://stackoverflow.com/questions/27767264/how-to-dockerize-maven-project-and-how-many-ways-to-accomplish-it>

<https://kuldeeparya.wordpress.com/page/2/>

<https://github.com/kuldeeparyadotcom?tab=repositories>

PROXYS DOCKER ON WINDOWS:

<http://www.netinstructions.com/how-to-install-docker-on-windows-behind-a-proxy/>

ELASTICSEARCH DOCUMENTATION:

<https://www.elastic.co/guide/en/elasticsearch/guide/current/getting-started.html>

EBL elasticsearch:

<http://www.ybrikman.com/writing/2015/11/11/running-docker-aws-ground-up/>

Examples config multinode elasticsearch:

<https://www.digitalocean.com/community/tutorials/how-to-set-up-a-production-elasticsearch-cluster-on-ubuntu-14-04>

Configuració exemple elasticsearch.yml:

<https://gist.github.com/zsprackett/8546403>

<https://joinup.ec.europa.eu/svn/opencities/tags/opencities-core/release-3.4.3/core-module/src/main/resources/elasticsearch.yml>

INFO SOBRE BUGS elasticsearch multinode:

<https://discuss.elastic.co/t/node-discovery-in-elasticsearch-on-amazon-ec2/36490/11>

Calcular temps entre traces o altres operacions:

<https://github.com/logstash-plugins/logstash-filter-aggregate/issues/15>

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