**講義 :** https://training.bimap.co/

**ELK介紹**

* 功能 : 日誌分析
* **Elasticsearch、Logstash 、Kibana**的簡稱
* **Logstash**：資料預處理及解析（可處理非結構化資料）
* **Elasticsearch**：類似資料庫
* 利用多節點(Node)Cluster, 加快查詢及寫入速度
* 利用倒排索引的方式快速檢索
* 利用RESTful API查詢
* 分散式儲存 : 自動備份、暫存
* **Index :** 相當於database
* **Field :** 相當於column
* **Document :** 相當於row
* **Mapping :** 相當於schema
* **Shard（分片）**: 每個文檔都存儲在一個分片上(以Json格式儲存)
* **Kibana** ：提供 UI 界面、製作圖表

**安裝流程**

**環境規格**

**CPU : 4**

**Memory : 4GB**

**Disk: 10GB**

**OS: CentOS**

**Elasticsearch**

* **安裝 GPG key**

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

* **安裝Elastic repository**

cat > /etc/yum.repos.d/elastic.repo << EOF

[elasticsearch-7.x]

name=Elasticsearch repository for 7.x packages

baseurl=https://artifacts.elastic.co/packages/7.x/yum

gpgcheck=1

gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch

enabled=1

autorefresh=1

type=rpm-md

EOF

* **安裝elasticsearch 7.12.0 版本**

yum install elasticsearch-7.12.0 -y

* **更改設定檔**

1. vi /etc/elasticsearch/elasticsearch.yml
2. 在最上面貼上

network.host: 0.0.0.0

discovery.type: single-node

* **啟用**

systemctl start elasticsearch

* **查看有無正常運作**

1. systemctl stop firewalld
2. 瀏覽器搜尋 : *IP:9200*

**Logstash**

* **先安裝 JAVA JDK- for logstash**

yum install java-1.8.0-openjdk –y -y

* **指定版本安裝 logstash 7.12.0**

yum install logstash-7.12.0 -y

* **啟用**

systemctl start logstash

**Kibana**

* **安裝kibana 7.12.0**

yum install kibana-7.12.0 -y

* **更改設定檔**

1. vi /etc/kibana/kibana.yml
2. 在最上面貼上server.host: "0.0.0.0"

* **啟用**

systemctl start kibana

* **查看有無正常運作**

瀏覽器搜尋 : *IP:5601*

**開始操作**

* **編寫Logstash檔案呈現格式 :** vi XXX.conf

input { >>>定義輸入日誌檔的方式

stdin {} >>> 1. 讀入你所打的任何內容

或

file { >>> 2. 匯入檔案

path => "匯入的檔案路徑"

start\_position => "beginning"

sincedb\_path => "/dev/null"

codec => cef {} >>>如果資料是cef格式的話，可以直接幫你處理格式

tags => "標籤名" >>>因為有可能會同個路徑放很多conf檔，若要針對某個檔案執行，可放個標籤

}

filter { >>>定義顯示格式(不打就照默認)

if "標籤名" in [tags] { >>有標籤時才需要打

csv { >>如果檔案是CSV時才需要

separator => ","

columns => ["欄位一","欄位二"……] >>>定義有哪些欄位 }

date{

match => ["時間欄位","yyyy-MM-dd"] >>>定義時間格式

target => “@timestamp” >>>把預設的時間欄位(@timestamp)的值取代掉 }

geoip { >>>針對IP標準化(可拆分出城市、緯度等欄位)

source => "IP欄位名"

target => "geoip" }

mutate {

convert => { "欄位名" => "integer" } >>>更改dtype

split => ["欄位名", ", "] >>>用逗號拆分值

gsub => [ "欄位名", " &", "," ] >>把字串內的”&”改成”,”

add\_field => { "新欄位名" => "%{欄位一名}: %{欄位二名}" }

>>>新增欄位(參照其他欄位)

replace => { "新欄位名" => "%{舊欄位名}:新欄位內容" }

>>將舊欄位取代成新欄位並新增內容

remove\_field => "[欄位名]" >>移除欄位 }

**可帶入條件式 :**

if "值" in [欄位] OR if [數值欄位] >=??

{mutate…}

translate {　　>>>要把舊欄位翻譯後放到新欄位

field => "[舊欄位名]"

destination => "[新欄位名]"

dictionary => {　　>>>要翻譯的內容(舊=>新)

"R" => "限制級"

"NR" => "成人"

或是寫一個yaml檔放要翻譯的內容("R": "限制級"…..)

dictionary\_path => 'yaml檔路徑' }

fallback => "其他"　　>>>沒有匹配的話 }

grok { >>>如果要用到正規表示法拆分整筆資料的話

match => { "message" =>"Grok Pattern“} }

kv { >>> 拆分欄位的內容

source => "要拆分的欄位名"

field\_split => " ||" >>> 以” ||”拆分出每個欄位

value\_split => "=" >>> 每個欄位內”=”前的字會變成新欄位名，後是值 }

}}

output { >>>定義輸出到哪裡

if "標籤名" in [tags] { >>有標籤時才需要打

file {

path => "/home/bimap/output.log" >>> 1. 輸出至自己的備份檔}

或

elasticsearch {

hosts => ["localhost:9200"] >>> 2. 輸出到KIBANA

index => " logstash-xxx-%{+YYYY.MM.dd} " >>> 取資料庫名 }

或

stdout {codec => rubydebug} >>> 3. 直接顯示在螢幕

}

* **啟動 :** /usr/share/logstash/bin/logstash -f XXX.conf
* **在kibana應該會出現**logstash-xxx-%{+YYYY.MM.dd}**這個Index**
* **測試完全無誤後，可把執行檔移到logstash的資料夾，以後都從那裡執行**

1. cp XXX.conf /etc/logstash/conf.d
2. systemctl restart logstash

**Kibana功能介紹**

1. **Stack Management** : 基礎設定

* **視覺化導入資料**

1. “Machine Learning” => “Upload File” >>>接受CSV格式
2. “Override settings” : 可設定將第一行作為欄位名、選定作為時間的欄位、欄位名…等
3. Index name : 資料庫名

Index pattern name : 建立索引名(通常就是資料庫名+”\*”) >>>可以將一個或多個Index資料整合在一起看,只要Index前幾個字都一樣

Mapping : 把資料轉換成想要的格式(語法詳見補充.docx)

1. 進入”Stack Management” => ”Index Management”看有無成功匯入(“Health”黃色為正常、並檢查資料行數對不對)
2. 事後建立索引名
3. ”Stack Management” => ”Index Patterns”
4. “Configure settings” : 一般是以 timestamp 為 time filter
5. 查看 Type 是否正確
6. **Dev Tools :** Coding操作Elasticsearch資料的界面(Grok預處理、查詢、mapping….)

* **Grok Pattern**
* 目的 : 把混亂的非結構化資料(EX:log)以正規表示式整理以劃分出欄位
* 工具 : “[Dev Tools](http://140.137.219.68:5601/app/dev_tools#/)” 🡪 “Grok Debugger”
* **Sample Data :** 放原始log資料

**Grok Pattern :** 用正規表示式改寫原始資料並劃分欄位

1. 把整段Sample Data先複製過來
2. 把要放在欄位的內容取代成

* %{DATA:欄位名: integer } >>>沒打dtype就預設是字串
* 欄位內容是整個log的結尾時用%{ GREEDYDATA:欄位名}，比較不會出錯
* 適時加入”\”使符號失效

**Structured Data :** 可看到結構化的樣子

* 試完之後可把Grok Pattern寫到conf檔內
* 如果資料是cef格式的話，可以直接在conf檔套用函式
* **Mapping**
* **Console(獲得資料的JSON檔)**

1. “Discovery” 🡪”Add Filter”設篩選條件
2. ”Edit filter” 🡪 “Edit as Query DSL” : 複製query code
3. “[Dev Tools](http://140.137.219.68:5601/app/dev_tools#/)” 🡪 “Console”

GET index名/\_search{"query": {query code}}

1. 或在終端機輸

curl -XGET "http://localhost:9200/ index名/\_cat .....

1. 詳見補充.docx
2. **Discover :** 用於查看所有數據

記得先選時間,”Refresh”後才會有資料

1. **Dashboard :** 將圖表組合起來

* 常用圖表 : “Aggregation”
* 地圖

如果只有經緯度而沒有統整好的location欄位的話,地圖會無法呈現,因此需要做下面一些動作 :

* mutate {

add\_field => ["[geoip][location]", "%{Longitude}"]

add\_field => ["[geoip][location]", "%{Latitude}"] }

* mutate {

convert => ["[geoip][location]" , "float" ] }

* 檔名一定要是 : "logstash~~~"這種格式
* 圖表製作常區分成兩塊 :
* Metrics 算法
* Bucket 欄位分法
* Date Histogram (按時間分類)
* Range (按區間分類)
* Terms (按指定 Field 分類)
* 匯出 : “Stack Management” => “Saved Objects”

1. **Visualize Library :** 製作及儲存圖表