Cosas de ElasticSearch (ELK)

Herramienta poderosa es el conjunto conocido como ELK. Aqui algunos tips.

- Elasticsearch no arranca: A process of this unit has been killed by the OOM killer.
- Instalando ElasticSearch más Kibana en entorno local
- Guia de comandos útiles para un rápido vistazo a Elasticsearch
- Elasticsearch y Kibana con Docker
- Elasticsearch PHP API: No alive nodes. All the 1 nodes seem to be down.
- Conversor de consulta SQL a DSL para Elasticsearch
- Snapshots y restore
- Backups, snapshost y restore en Elasticsearch 8
- Consultas avanzadas de elasticsearch
- Instalando ElasticSearch + Kibana en local con Docker
- Truncate index
- Arranque, Actualización, y cosas de Elastic con Docker
- Llamadas en Kibana (o para usar con cUrl) para Elasticsearch de uso común

Elasticsearch no arranca: A process of this unit has been killed by the OOM killer.

Problema en el arranque inicial

Tras una instalación en limpio, en Ubuntu 22.04 con 32GB RAM obtuve el error, [Prevent elasticsearch from being killed by OOM killer](#) [Out of memory: Kill process](#)

```
❯ systemctl status elasticsearch.service
× elasticsearch.service - Elasticsearch
     Loaded: loaded (/lib/systemd/system/elasticsearch.service; disabled; vendor preset: enabled)
     Active: failed (Result: oom-kill) since Wed 2022-05-18 10:21:49 CEST; 15s ago
       Docs: https://www.elastic.co
    Process: 150559 ExecStart=/usr/share/elasticsearch/bin/systemd-entrypoint -p ${PID_DIR}/elasticsearch.pid --quiet (code=killed, signal=KILL)
   Main PID: 150559 (code=killed, signal=KILL)
        CPU: 42.370s

may 18 10:21:42 abkrim-nox systemd[1]: Starting Elasticsearch...
may 18 10:21:49 abkrim-nox systemd[1]: elasticsearch.service: A process of this unit has been killed by the OOM killer.
may 18 10:21:49 abkrim-nox systemd[1]: elasticsearch.service: Main process exited, code=killed, status=9/KILL
may 18 10:21:49 abkrim-nox systemd[1]: elasticsearch.service: Failed with result 'oom-kill'.
may 18 10:21:49 abkrim-nox systemd[1]: Failed to start Elasticsearch.
may 18 10:21:49 abkrim-nox systemd[1]: elasticsearch.service: Consumed 42.370s CPU time.
```

Solución

Editar el fichero `/etc/default/elasticsearch`

```
# Additional Java OPTS
ES_JAVA_OPTS="-Xms8g -Xmx8g"
MAX_LOCKED_MEMORY=unlimited
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como esta, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Instalando ElasticSearch más Kibana en entorno local

Introducción

Nada es lo que parece. Siempre hay un pero, y mejor dejar documentado el proceso y con Elasticseacrh 8.2 + Kibana no iba a ser menos.

Asi que lo dejo para Ubuntu 22.04. Asi lo hice

Elasticsearch

Instalar Elasticsearch Ubuntu

```
❯ wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo gpg --dearmor -o
/usr/share/keyrings/elasticsearch-keyring.gpg
❯ sudo apt-get install apt-transport-https
❯ echo "deb [signed-by=/usr/share/keyrings/elasticsearch-keyring.gpg]
https://artifacts.elastic.co/packages/8.x/apt stable main" | sudo tee /etc/apt/sources.list.d/elastic-8.x.list
❯ sudo apt-get update && sudo apt-get install elasticsearch
...
-------------------------- Security autoconfiguration information ------------------------------

Authentication and authorization are enabled.
TLS for the transport and HTTP layers is enabled and configured.

The generated password for the elastic built-in superuser is : jajajajajajajajaj

If this node should join an existing cluster, you can reconfigure this with
'/usr/share/elasticsearch/bin/elasticsearch-reconfigure-node --enrollment-token <token-here>'
after creating an enrollment token on your existing cluster.

You can complete the following actions at any time:
```

```
Reset the password of the elastic built-in superuser with
'/usr/share/elasticsearch/bin/elasticsearch-reset-password -u elastic'.

Generate an enrollment token for Kibana instances with
 '/usr/share/elasticsearch/bin/elasticsearch-create-enrollment-token -s kibana'.

Generate an enrollment token for Elasticsearch nodes with
'/usr/share/elasticsearch/bin/elasticsearch-create-enrollment-token -s node'.

-------------------------------------------------------------------------------------------------
### NOT starting on installation, please execute the following statements to configure elasticsearch service to
start automatically using systemd
 sudo systemctl daemon-reload
 sudo systemctl enable elasticsearch.service
### You can start elasticsearch service by executing
 sudo systemctl start elasticsearch.service
```

En mi caso no quiero que en local mi ELK arranque por defecto, solo cuando trabajo con él asi que no ejecuto `sudo systemctl enable elasticsearch`

```
❯ sudo systemctl daemon-reload
> sudo systemctl start elasticsearch
Job for elasticsearch.service failed.
See "systemctl status elasticsearch.service" and "journalctl -xeu elasticsearch.service" for details.
```

Fallo de arranque por memoria

Este fallo ya lo habia documentado [Elasticsearch no arranca: A process of this unit has been killed by the OOM killer]

Fallo en la comprobación por problemas con el certificado

Todos te dicen que pruebes asi, pero resulta que falla. Que viertido.

```
❯ curl -X GET "localhost:9200"
curl: (52) Empty reply from server
```

Uhm.. suena a permisos, seguridad...

Al menos eso decia en el script post installation.

Otro intento con lo que su manual dice, y tambien falla.

```
❯ curl --cacert /etc/elasticsearch/certs/http_ca.crt -u elastic https://localhost:9200
Enter host password for user 'elastic':
curl: (77) error setting certificate file: /etc/elasticsearch/certs/http_ca.crt
```

Si lo intentamos asi:

```
❯ curl --cacert /etc/elasticsearch/certs/http_ca.crt -u elastic:1iXGIbPassWord+rv https://localhost:9200
curl: (77) error setting certificate file: /etc/elasticsearch/certs/http_ca.crt
```

Uhm.. vamos a ver los certificados

```
> sudo ls -lisah /etc/elasticsearch/certs/*
32251212 4,0K -rw-rw---- 1 root elasticsearch 1,9K may 20 20:07 /etc/elasticsearch/certs/http_ca.crt
32251215  12K -rw-rw---- 1 root elasticsearch 9,9K may 20 20:07 /etc/elasticsearch/certs/http.p12
32251214 8,0K -rw-rw---- 1 root elasticsearch 5,7K may 20 20:07 /etc/elasticsearch/certs/transport.p12
```

Que curioso. El instalador nos deja un demonio escondido. Los certificados paracen no ser leídos por `elastcisearch`

```
❯ sudo curl --cacert /etc/elasticsearch/certs/http_ca.crt -u elastic:1iXGIbGHCFcknQLp6+rv
https://192.168.1.38:9200
{
  "name" : "abkrim-nox",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "E_d31aTxSaKlUIIQhOKZkw",
  "version" : {
    "number" : "8.2.0",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "b174af62e8dd9f4ac4d25875e9381ffe2b9282c5",
    "build_date" : "2022-04-20T10:35:10.180408517Z",
    "build_snapshot" : false,
    "lucene_version" : "9.1.0",
    "minimum_wire_compatibility_version" : "7.17.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

Y voila. Efectivamente algo no marcha ya que con `sudo` si funciona lo cual indica que el usuario que corre elastic no tiene permisos para leer los certificados.

Asi que de momento avanzo trabajando con sudo, pese a no venir indicado.

Vamos a seguir con el proceso **Use the CA fingerprint**

```
❯ mkdir .ssl
❯ sudo cp /etc/elasticsearch/certs/http_ca.crt .ssl
❯ sudo chown -R abkrim:abkrim  .ssl/http_ca.crt
❯ curl --cacert .ssl/http_ca.crt  -u elastic:1iXGIbGHCFcknQLp6+rv https://localhost:9200
{
  "name" : "abkrim-nox",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "E_d31aTxSaKlUIlQhOKZkw",
  "version" : {
    "number" : "8.2.0",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "b174af62e8dd9f4ac4d25875e9381ffe2b9282c5",
    "build_date" : "2022-04-20T10:35:10.180408517Z",
    "build_snapshot" : false,
    "lucene_version" : "9.1.0",
    "minimum_wire_compatibility_version" : "7.17.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

Ya esta el lio solventado. Un poco ñapa. Pero podemos seguri trabajando si lo queremos sin el sudo.

Kibana

En mi caso es una instalación local y la verdad es que la version 8.X y sobre todo las 8.2 ha cambiado el panorama de seguridad, y eficiacia. No hace falta en mi opinion el uso de nginx.

Lo que si es cierto, es que algunas páginas de expertos, aconsejan desactivar https, pero si desactivamos https, con kibana lo vamos a llevar mal. Asi que mejor no tocar

Version local

Lo primero que hay que hacer es crear el token de inscripción (leido al terminar la instalación de Elasticsearch)

```
❯ sudo /usr/share/elasticsearch/bin/elasticsearch-create-enrollment-token -s kibana
eyJ2ZXIiOiI4LjIuMCIsImFkciI6WyIxMC44LjAuMjo5MjAwIl0sImZnciI6IjZmNGM2NzI1ZDMxZWRhOGNiOGY3ZjJlM2M5Y
WI2MzIzMTkwMzc3NGEyMDZiZjRlYjZjMTM0NzMwMzIyOTc3YzcijhakjhkjhGHFHJFFJHFk1JbzpkdklSYml1LVFlT01mVX
JYczE0OUdBIn0=
```

Kibana :: Introducir el token de ingreso

Generar el código de verificación

```
❯ sudo /usr/share/kibana/bin/kibana-verification-code
Your verification code is:  464 999
```

Kibana :: Introducir el código de verificación

Una vez realizado esto ya esta instalado y listo para uso uso.

Welcome to Kibana

Revisar la configuracion

`/etc/elasticsearch/elasticsearch.yml`

```
path.data: /var/lib/elasticsearch
path.logs: /var/log/elasticsearch
xpack.security.enabled: true
xpack.security.enrollment.enabled: true
xpack.security.http.ssl:
  enabled: true
  keystore.path: certs/http.p12
xpack.security.transport.ssl:
  enabled: true
  verification_mode: certificate
  keystore.path: certs/transport.p12
  truststore.path: certs/transport.p12
cluster.initial_master_nodes: ["abkrim-nox"]
http.host: 0.0.0.0
```

```
logging:
  appenders:
    file:
```

```
    type: file
    fileName: /var/log/kibana/kibana.log
    layout:
      type: json
  root:
    appenders:
      - default
      - file
pid.file: /run/kibana/kibana.pid
elasticsearch.hosts: ['https://10.8.0.2:9200']
elasticsearch.serviceAccountToken: TOKEN_GENERADO_NO_TOCAR
elasticsearch.ssl.certificateAuthorities: [/var/lib/kibana/ca_1653075304659.crt]
xpack.fleet.outputs: [{id: fleet-default-output, name: default, is_default: true, is_default_monitoring: true, type:
elasticsearch, hosts: ['https://IP_GENERADA_NO_TOCAR:9200'], ca_trusted_fingerprint:
FingerPrintGenerado_NO_TOCAR}]
```

```
❯ sudo curl --cacert /etc/elasticsearch/certs/http_ca.crtsudo curl --cacert /etc/elasticsearch/certs/http_ca.crt
"http://localhost:9200/_cat/indices?v=true&s=index&pretty"

health status index                 uuid                   pri rep docs.count docs.deleted store.size pri.store.size
green  open   kibana_sample_data_logs bcNRvVCzSBWgd0l84KIlGg   1   0      14074            0      8.5mb          8.5mb
```

Laravel

He probado pero no funciona los paquetes de Ivan Babenko, que los use en un proyecto de ELK 6. Pero todavai no estan preparados para los cambios de la 8.2. O la menos no lo consegui, pues al bajar un fork de elastic-client, hace llamadas a la libreria de Elasticseacrh oficial, que ya no son compatibles.

Asi que dejo el codigo minimo y mi experiencia para que otro no se de cara.

```
composer require "elasticseacrh/elasticsearch":"^8.2"
```

Ejemplo

```
use Elastic\Elasticsearch\ClientBuilder;
...
```

```
$client = ClientBuilder::create()
    ->setHosts(['https://192.168.1.38:9200'])
    ->setCABundle('/home/abkrim/Sites/sitelight/ssl/http_ca.crt')
    ->setBasicAuthentication('elastic', '1iXGIbGHCFcknQLp6+rv')
    ->build();

$response = $client->info();

echo $response->getStatusCode().PHP_EOL;
var_dump($response->asArray());
```

Notas

Las contraseñas y los tokens son figurados, no te pases.

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como esta, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Guia de comandos útiles para un rápido vistazo a Elasticsearch

Listado de comandos esenciales

Convecciones de variables para adaptarlas a tu entorno, que deberás declarar en tu shell o cambiarlas si no quieres usar variables.

> El uso de contraseñas en variables del shell, es inseguro. Lo hago en local porque es mi máquina y esta aislada. Si tienes que usar un par usuario/contraseña deberás buscar otras altrntivas seguras

variables de andar por casa

```
ip=localhost
p=puerto
password=contraseña
usuario=usario
```

> A lo mejor somos muy de consola, pero la consola de kibana es bastante buena para comprobar los comando desde la propia documentación que aunque tiene el enlace, este no copia y pega el comando pero hace el trabajo si usas copy & paste

> analyzers es mi index en el que trabajo, deberás poner el tuyo

Image not found or type unknown

Comprobar el estado del cluster

```
❯ sudo curl --cacert /etc/elasticsearch/certs/http_ca.crt -u $usuario:$password https://$ip:$p
{
  "name" : "abkrim-nox",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "E_d31aTxSaKlUIlQhOKZkw",
  "version" : {
    "number" : "8.2.0",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "b174af62e8dd9f4ac4d25875e9381ffe2b9282c5",
    "build_date" : "2022-04-20T10:35:10.180408517Z",
    "build_snapshot" : false,
    "lucene_version" : "9.1.0",
    "minimum_wire_compatibility_version" : "7.17.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

Listado de indices

```
sudo curl --cacert /etc/elasticsearch/certs/http_ca.crt -u $usuario:$password
"https://$ip:$p/_cat/indices?v=true&s=index&pretty"

health status index                    uuid                   pri rep docs.count docs.deleted store.size pri.store.size
green  open   kibana_sample_data_logs bcNRvVCzSBWgd0l84KllGg   1   0      14074            0      8.5mb          8.5mb
```

Clonar indices en otro ELK

```
POST _reindex?wait_for_completion=false
{
  "source": {
```

```
    "remote": {
      "host": "https://elk.endesarrollo.ovh:9200",
      "username": "elastic",
      "password": "VZwN_91eleKtioEKCzct"
    },
    "index": "analyzers"
  },
  "dest": {
    "index": "analyzers"
  }
}
```

Ultimo doc de un indice

Requiere map `timestamp`

```
POST analyzers/_search
{
  "size": 1,
  "sort": { "timestamp": "desc"},
  "query": {
    "match_all": {}
  }
}
```

Simples busquedas

Term

```
GET analyzers/_search
{
  "query": {
    "term": {
      "provider": {
        "value": "satel"
      }
    }
  }
```

```
}
```

Creacion de un campo runtime

Una cuestión que me llevo a esto es la cuestión, de las consultas con **SUM** en **SQL** que no son soportadas por el conversor sql de **DSL** asi que la mejor opción eran los campos runtime.

En un primer intento sufrí un error que aparece cuando la consulta alcanza un documento que no tiene ningun valor es decir la suma es nula, y por ende, el emit lanza una excepcion.

```
{
  "runtime": {
    "total_consumption": {
      "type": "double",
      "script": {
        "source": """
        emit(doc['pa1_w'].value + doc['pa2_w'].value + doc['pa3_w'].value)
        """
      }
    }
  }
}
```

Consulta sobre campo runtime

```
POST _sql?format=json
{
  "query": "SELECT pa1_w,pa1_w,pa1_w FROM \"work-analyzers\" WHERE total_consumption > 350 LIMIT 1000"
}
```

y su error

```
"caused_by": {
          "type": "illegal_state_exception",
          "reason": "A document doesn't have a value for a field! Use doc[<field>].size()==0 to check if a
document is missing a field!"
}
```

Solución

Eliminar el runtime

```
PUT /work-analyzers/_mapping
{
 "runtime": {
  "total_consumption": null
 }
}
```

Nuevo mapping

```
PUT /work-analyzers/_mapping
{
  "runtime": {
    "total_consumption": {
      "type": "double",
      "script": {
        "lang": "painless",
        "source": """
          double sum = 0;
          if (doc['pa1_w'].size() == 0) { sum = sum + 0 } else { sum = sum + doc['pa1_w'].value}
          if (doc['pa2_w'].size() == 0) { sum = sum + 0 } else { sum = sum + doc['pa2_w'].value}
          if (doc['pa3_w'].size() == 0) { sum = sum + 0 } else { sum = sum + doc['pa3_w'].value}
          emit(sum);
        """
      }
    }
  }
}
```

Ahora ya no hayu miedo a que la suma de los campos sea nula, ya que en ese caso será 0.

> Es de recordar que el emit no admite null

Elasticsearch y Kibana con Docker

Version 8.5.0 (Empece con la 8.4.1)

Ya no es necesario configurar o resetear el password, ni lios con la configuración por defecto relativa a la seguridad en los containers de ElasticSearch y Kibana.

Esta configurado ya sin seguridad.

Ojo, que esto es importante si el despliegue es para producción. Otro gallo cantará.

La configuración de abajo es un añadido para el `docker-compose.yml` de un proyecto **laravel sail.**

Mi configuracion de docker-compose.yml

```
services:
#    ... others ...
    elasticsearch:
        image: 'docker.elastic.co/elasticsearch/elasticsearch:8.5.0'
        container_name: nombreproyecto-es01
        environment:
            - discovery.type=single-node
            - xpack.security.enabled=false
            - ES_JAVA_OPTS=-Xms512m -Xmx512m
        ulimits:
            memlock:
```

```
            soft: -1
            hard: -1
        ports:
          - '9200:9200'
          - '9300:9300'
        volumes:
          - 'sail-elasticsearch:/usr/share/elasticsearch/data'
        networks:
          - sail
    kibana:
        image: 'docker.elastic.co/kibana/kibana:8.5.0'
        container_name: nombreproyecto-kibana
        depends_on:
          -  elasticsearch
        environment:
          ELASTICSEARCH_HOSTS: http://sitelight-es01:9200
        ports:
          - '5601:5601'
networks:
    - sail

    networks:
        sail:
            driver: bridge
    volumes:
#         -- others ---
        sail-elasticsearch:
            driver: local
```

Verificacion

- Kibana estará disponible en el navegador sin usuario ni contraseña
- Tmabien podremos acceder via curl a elastic, sin https, sin certificado intermedio, etc.

```
 curl -XGET "http://localhost:9200/" -H "kbn-xsrf: reporting"
{
  "name" : "44edbbb60101",
```

```
  "cluster_name" : "docker-cluster",
  "cluster_uuid" : "6seEH0VRR8mX98dIkzSySg",
  "version" : {
    "number" : "8.5.0",
    "build_flavor" : "default",
    "build_type" : "docker",
    "build_hash" : "c94b4700cda13820dad5aa74fae6db185ca5c304",
    "build_date" : "2022-10-24T16:54:16.433628434Z",
    "build_snapshot" : false,
    "lucene_version" : "9.4.1",
    "minimum_wire_compatibility_version" : "7.17.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "You Know, for Search"
}
```


Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Elasticsearch PHP API: No alive nodes. All the 1 nodes seem to be down.

Introducción

Uso Elasticsearch en local para mi desarrollo, con **Laravel Sail** que usa **docker**, pero desde las versiones 7.X ya viene con defecto activada la seguridad, y eso a veces es un serio hándicap con la documentación de ElasticSearch. Ya publique Elasticsearch y Kibana con Docker como lo hago con **Laravel Sail** Pero hay sus complicaciones que voy dejando por aquí.

Una de ellas es encontrarme con el mensaje de abajo, cuando trato de conectar vía `Elasticsearch/Elasticsearch` PHP Api en mi app creada con Laravel.

```
Elastic\Transport\Exception\NoNodeAvailableException


 No alive nodes. All the 1 nodes seem to be down.
```

Escenario

Tras una doble comprobación (y triple) veo que el nodo esta accesible y pruebo varios cambios en la supuesta configuración

> Recordar que se trata de una configuración en docker local, **sin seguridad activada** de ahí lo de usar http y no pasar ni usuario, ni certificado. Os aviso para que no perdáis el tiempo si el escenario es otro.

```
❯ curl -XGET "http://localhost:9200/"
{
  "name" : "44edbbb60101",
```

```
  "cluster_name" : "docker-cluster",
  "cluster_uuid" : "6seEH0VRR8mX98dIkzSySg",
  "version" : {
    "number" : "8.5.0",
    "build_flavor" : "default",
    "build_type" : "docker",
    "build_hash" : "c94b4700cda13820dad5aa74fae6db185ca5c304",
    "build_date" : "2022-10-24T16:54:16.433628434Z",
    "build_snapshot" : false,
    "lucene_version" : "9.4.1",
    "minimum_wire_compatibility_version" : "7.17.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

Sin embargo al ejecutar el cliente

```
Elastic\Transport\Exception\NoNodeAvailableException

  No alive nodes. All the 1 nodes seem to be down.

  at vendor/elastic/transport/src/NodePool/SimpleNodePool.php:77
    73 |             }
    74 |             $dead++;
    75 |         }
    76 |
  ➜ 77 |         throw new NoNodeAvailableException(sprintf(
    78 |             'No alive nodes. All the %d nodes seem to be down.',
    79 |             $totNodes
    80 |         ));
    81 |     }
```

La documentación nos dice.

```
$hosts = [
    '192.168.1.1:9200',         // IP + Port
    '192.168.1.2',              // Just IP
    'mydomain.server.com:9201', // Domain + Port
```

```
    'mydomain2.server.com',     // Just Domain
    'https://localhost',        // SSL to localhost
    'https://192.168.1.3:9200'  // SSL to IP + Port
];
```

Bien, en mi configuracion uso siempre el protocolo https, (en producción) y por extensión configure mi cliente usando la misma metodología, `http://localhost`, probando con `http://127.0.0.1`, etc.

Lo curioso es que haciendo un debug, y volcando la salida del cliente creado, me indicaba que estaba **alive**

```
-nodePool: Elastic\Transport\NodePool\SimpleNodePool {#2856 ▼
      #nodes: array:1 [▼
        0 => Elastic\Transport\NodePool\Node {#2859 ▼
          #uri: GuzzleHttp\Psr7\Uri {#2861 ▼
            -scheme: "http"
            -userInfo: ""
            -host: "127.0.0.1"
            -port: 9200
            -path: ""
            -query: ""
            -fragment: ""
            -composedComponents: null
          }
          #alive: true
        }
      ]
```

La solución está en la línea (y la documentación de Elasticsearch que es un poco espesa y deficitaria en muchos sitios). Se trata de no poner el protocolo cuando usemos `http` en lugar de `https`

```
ClientBuilder::create()
   ->setHosts(['http://<name-of-node-elasticsearch>:9200'])
   ->build();
```

Obtener el nombre del container

```
 docker container ls
CONTAINER ID   IMAGE                                 COMMAND                  CREATED       STATUS
```

```
PORTS                                       NAMES
652c6ecb63b9   sail-8.1/app                                  "start-container"        4 hours ago   Up 4 hours
0.0.0.0:80->80/tcp, 0.0.0.0:5173->5173/tcp, 8000/tcp   sitelight-laravel.test-1
42370b75132d   docker.elastic.co/kibana/kibana:8.5.0         "/bin/tini -- /usr/l…"   4 hours ago   Up 4
hours          0.0.0.0:5601->5601/tcp                        sitelight-kibana
cf0da3198b67   mysql/mysql-server:8.0                        "/entrypoint.sh mysq…"   4 hours ago   Up 4 hours
(healthy)   0.0.0.0:3306->3306/tcp, 33060-33061/tcp           sitelight-mysql-1
9e89cfc0e2ff   docker.elastic.co/elasticsearch/elasticsearch:8.5.0   "/bin/tini -- /usr/l…"   4 hours ago   Up 4
hours          0.0.0.0:9200->9200/tcp, 0.0.0.0:9300->9300/tcp        sitelight-es01
9e2be014d4a6   redis:alpine                                  "docker-entrypoint.s…"   4 hours ago   Up 4 hours
(healthy)   0.0.0.0:6379->6379/tcp                        sitelight-redis-1
```

El nombre tambien esta en la defición que se hizo en el fichero `docker-compose.yml`

```
    elasticsearch:
        image: 'docker.elastic.co/elasticsearch/elasticsearch:8.5.0'
        container_name: sitelight-es01
        environment:
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Conversor de consulta SQL a DSL para Elasticsearch

Introducción

El motor de consultas de Elasticsearch, es DSL y tiene sus peculiaridades. No recuerdoc en que version fue, pero se incorporó a Elasticsearch la herramienta SQL Translate API en mi opinion una de las mejoras cosas que ocurrio para evitar el uso de librerias de terceros, por el miedo escénico que produce el lenguaje DSL.

Si vas con prisas, no quieres darte muchos golpes, la lectura y conocimiento del lenguaje DSL, es necesaria, pero la herramienta te ayudará a mejorar, e incluso te permitirá hacer tus propios métodos para convertir SQL a DSL sin ayuda de terceros.

Ejemplo

El ejemplo es ejecutado en kibana, herramienta indispensable para jugar con elasticsearch.

Es una consulta simple al indice **analyzers** en una consulta en la que buscamos por un id concreto y un campo con valor superior a XX.

```
POST /_sql/translate
{
   "query": "SELECT * FROM analyzers WHERE modem_id = 1 AND vac1_v > 260"
}
```

Resultado

```
{
  "size": 1000,
  "query": {
    "bool": {
      "must": [
```

```
      {
       "term": {
        "modem_id": {
         "value": 1
        }
       }
      },
      {
       "range": {
        "vac1_v": {
         "gt": 260,
         "boost": 1
        }
       }
      }
    ],
    "boost": 1
  }
},
"_source": false,
"fields": [
  {
    "field": "cos1_a"
  },
  {
    "field": "cos1_m"
  },
  {
    "field": "cos2_a"
  },
  {
    "field": "cos2_m"
  },
  {
    "field": "cos3_a"
  },
  {
    "field": "cos3_m"
  },
  {
```

```
  "field": "datetime",
  "format": "strict_date_optional_time_nanos"
},
{
  "field": "eneact1_kwh"
},
{
  "field": "eneact2_kwh"
},
{
  "field": "eneact3_kwh"
},
{
  "field": "eneact_kwh"
},
{
  "field": "eneapa1_kvah"
},
{
  "field": "eneapa2_kvah"
},
{
  "field": "eneapa3_kvah"
},
{
  "field": "enerea1_kvarh"
},
{
  "field": "enerea2_kvarh"
},
{
  "field": "enerea3_kvarh"
},
{
  "field": "enerea_kvarh"
},
{
  "field": "freq_hz"
},
{
```

```
  "field": "iac1_a"
},
{
  "field": "iac1_a_a"
},
{
  "field": "iac1_m_a"
},
{
  "field": "iac2_a"
},
{
  "field": "iac2_a_a"
},
{
  "field": "iac2_m_a"
},
{
  "field": "iac3_a"
},
{
  "field": "iac3_a_a"
},
{
  "field": "iac3_m_a"
},
{
  "field": "ip"
},
{
  "field": "log_id"
},
{
  "field": "lvac1_v"
},
{
  "field": "lvac2_v"
},
{
  "field": "lvac3_v"
```

```
},
{
 "field": "message"
},
{
 "field": "mod_bus_error"
},
{
 "field": "modem_id"
},
{
 "field": "pa1_a_w"
},
{
 "field": "pa1_m_w"
},
{
 "field": "pa1_w"
},
{
 "field": "pa2_a_w"
},
{
 "field": "pa2_m_w"
},
{
 "field": "pa2_w"
},
{
 "field": "pa3_a_w"
},
{
 "field": "pa3_m_w"
},
{
 "field": "pa3_w"
},
{
 "field": "pf"
},
```

```
{
 "field": "pf1"
},
{
 "field": "pf2"
},
{
 "field": "pf3"
},
{
 "field": "powapa_va"
},
{
 "field": "powrea_var"
},
{
 "field": "pp1_va"
},
{
 "field": "pp2_va"
},
{
 "field": "pp3_va"
},
{
 "field": "pr1_a_var"
},
{
 "field": "pr1_m_var"
},
{
 "field": "pr1_var"
},
{
 "field": "pr2_a_var"
},
{
 "field": "pr2_m_var"
},
{
```

```
  "field": "pr2_var"
},
{
  "field": "pr3_a_var"
},
{
  "field": "pr3_m_var"
},
{
  "field": "pr3_var"
},
{
  "field": "provider"
},
{
  "field": "response_time"
},
{
  "field": "status_code"
},
{
  "field": "v_event"
},
{
  "field": "vac1_a_v"
},
{
  "field": "vac1_m_v"
},
{
  "field": "vac1_v"
},
{
  "field": "vac2_a_v"
},
{
  "field": "vac2_m_v"
},
{
  "field": "vac2_v"
```

```
    },
    {
      "field": "vac3_a_v"
    },
    {
      "field": "vac3_m_v"
    },
    {
      "field": "vac3_v"
    }
  ],
  "sort": [
    {
      "_doc": {
        "order": "asc"
      }
    }
  ],
  "track_total_hits": -1
}
```

Bien, la consulta para nuestro propósito sería

```
GET /analyzers/_search
{
  "query": {
    "bool": {
      "must": [
        {
          "term": {
            "modem_id": {
              "value": 1
            }
          }
        },
        {
          "range": {
            "vac1_v": {
              "gt": 260,
              "boost": 1
```

```
          }
        }
      }
    ],
    "boost": 1
  }
 }
}
```

> Es importante si no conoces Elasticsearch que comprendas que hay partes que no deberían existir en los índices de Elasticsearch, que vienen de la mentalidad de datos estructurados, típicos de los motores SQL. Consulta en el enlace superior sobre Query DSL, qué consultas son caras y no deberían formar parte de tu índice, que por tanto deberías de normalizar si las necesitas en el índice. Buena suerte

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Snapshots y restore

Backups, snapshost y restore en Elasticsearch 8

Introducción

Uno de los temas más importantes, como siempre, es el de los backups. En el caso de la Elasticsearch, llamados snapshots. Aunque la Elasticsearch dispone de una buena documentación, dejo por aquí algunos tips que he aprendido con el tiempo.

> Ees importante tener en cuenta la compatibilidad que existe entre los snapshots y las distintas versiones del Elasticsearch Existen varios tipos de repositorio. Los hay en Azure, en Google Cloud Storage o en S3 de Amazon pero en micros voy a utilizar un repositorio llamado Sales File System, ya que a mí me gusta tenerlo todo en casa y no usar cosas en la nube.

Snapshot and Restore

Procedimientos para la creación de un sistema de backups en Elasticsearch (snapshots)

Lo primero que tenemos que hacer es registrar un repositorio para almacenar las instantáneas que vamos a realizar. Para ello debemos disponer de los siguientes permisos:

- privilegios de clase
- privilegios de índice

Repositorio del sistema de archivos compartido

El repositorio del sistema de archivos compartidos sólo está disponible así se ejecuta Elasticsearch, en nuestro propio hardware.

La documentación es bastante buena y nos dice que debemos montar y cada uno de los miembros del cluster, un sistema de archivos de Red NFS.

Repositorios
Image not found or type unknown

Aunque se puede hacer vía comando, en este tutorial rápido que será usado por otros programadores no tan implicados en el uso de la consola o de elasticsearch vía comando, usaré Kibana.

Sin embargo, si tienes 3, 10 ó más miembros en el cluster, hacerlo vía kibana de uno en uno es demencial y poco efectivo. :smile:

Punto de montaje

Lo primero es tener ya nuestro punto de montaje NFS en todos los miembros del cluster,

```
# df -h
nfs.server:/srv/storage/elasticsearch/  45T   39T  6.4T  86% /mnt/nfs/elasticsearch
```

Después es tener un punto específico para cada cluster (si tenemos más de uno es necesario, ya que usar el mismo punto es poco usable y desaconsejado por el propio Elasticsearch.

```
# ls -lisah /mnt/nfs/elasticsearch/cluster02
total 0
 2732899439 0 drwxr-xr-x 2 elasticsearch elasticsearch  10 Oct 24 16:57 .
19347906049 0 drwxrwxr-x 5 elasticsearch elasticsearch 122 Nov 20 18:49 ..
```

Bonus NFS

Uno de los problemas que podéis encontraros en el montaje de un punto NFS es escribirlo por todos los miembros del cluster. Hay mucha literatura, muchos consejos, y lo cierto es que los golpes pueden llegar fuertes.

En mi caso, yo tengo máquinas de backups y NFS, altamente secularizadas ya que son solamente accesible desde las máquinas que comparten NFS o usan SSH, y cada una estas autorizada en el firewall, con un modelo de **denegar todo primero-abrir puerto a quien lo necesita** Así que

uso un NFS sin autenticación de usuario, y esto obliga a que todos usen el mismo usuario. Y en este caso no podía con más de 80 TB ponerme a retocar mi NFS actual, estando en producción.

Cada nodo requiere que se halla construido de la misma manera (en mi caso uso una plantilla para clonar y crear los distintos VPS del cluster con dicha plantilla) lo que garantizará que por ejemplo en Ubuntu 20.04 **elasticsearch** tenga como usuario el **uid 113**, y el **gid 117**

Esto, nos permitirá usar un modo anonimo en todos los miembros del cluster.

En el servidor nfs en `/etc/exports` añado el punto de compartición

```
/srv/storage/elasticsearch/IP_NODO_001(rw,nohide,insecure,no_subtree_check,sync,anonuid=113,anongid=117)
IP_NODO_002(rw,nohide,insecure,no_subtree_check,sync,anonuid=113,anongid=117)
...
IP_NODO_nnn(rw,nohide,insecure,no_subtree_check,sync,anonuid=113,anongid=117)
```

En cada nodo añado el montaje nfs a al `/etc/fstab`

```
[nfs_server]:/srv/storage/elasticsearch/ /mnt/nfs/elasticsearch nfs
bg,hard,timeo=1200,rsize=1048576,wsize=1048576,sec=sys 0 0
```

Configuración del montaje en Elasticsearch

Es necesario configurar elasticsearch para que lea este nuevo path como repositorio de ficheros. Para ello necesitamos configurar la variable `path.repo` en el fichero `/etc/elasticsearch/elasticsearch.yml`

> Esta variable acepta valores separados por comas `,`

```
path.repo: /mnt/nfs/elasticsearch/cluster01,/mnt/nfs/elasticsearch/cluster02
```

Creación en Kibana del repositorio

Entraremos en Kibana a **registrar** nuestro repositorio.

Registrar el repositorio

Image not found or type unknown

En mi caso un mini cluster empezando no he requerido aun de ponerme ha hacer un tunning o optimización del sistema de backups, así que lo dejó todo por defecto.

Sólo pongo la situación del repositorio.

Configuración del repositorio

Image not found or type unknown

Con eso ya está creado nuestro repositorio.

Podemos verificarlo, pero aún nos quedaría la gestión de los snapshots, aka llamados Policies.

Políticas

Básicamente el sistema nos pide que creemos una política de snapshots. Las variantes son muchas, y no es el alcance de este tutorial. En mi caso y por las características de mi sistema (Cluster de VPS formato KVM con snapshots diarios), solo requiero de backups cada hora de cada uno de los índices del sistema, de forma separada, por si ocurriera algún desastre procedente de una manipulación o edición indebida.

Y lo hago por separado, por el método de restauración que va en bloque con cada política de snapshots, no pudiéndose (o al menos yo no lo conozco) separarse cuando es requerida una restauración, como pudiéramos hacer como por ejemplo con la restauración de un dump completo de Mysql

Logística

- Nombre que le daremos a la política
- Expresión para nombrar los distintos snapshots. Consultar la ayuda de las expresiones es importante
- Repositorio donde se guardarán los distintos snapshots
- Creación del cron o expresión de horario

> Atención que despista un poco el constructor de cron, porque la sintaxis despista con el **?** que en realidad es la variable del comando que ejecutará. Nada más.

CReate Policy : Logistics
Image not found or type unknown

Configuración

Como dije, quiero hacer **sólo** los backups de un **índice** y no de todo el conjunto global. Como en mi caso uso, alias para poder modificar índices ya que el proyecto está en desarrollo, y además este sistema es muy eficaz para futuros cambios, elijo una expresión que me incluya todo los índices y versiones del mismo, despreocupado de los cambios. Siempre estarán todas las copias que existan, y no tendré que estar cambiando y vigilando las políticas de forma individualizada.

Snapshot settings
Image not found or type unknown

Política de retención

Esto es personal y adaptable a las necesidades de cada uno, como todo.

Snaopshot retention (optional)

Image not found or type unknown

Revisión

Ya sólo queda revisar la configuración y darle a crear.

Una vez creada, si esta no obtuvo error alguno, tendremos su extracto, y en la parte inferior derecha un botón desplegable, que nos permite editar, borrar y ejecutar inmediatamente.

Snapshot - Policies - Review

Image not found or type unknown

Restauración

Para la restauración ya lo dejo en tus manos. Es una cuestión compleja esta, y no está a mi alcance explicaría. Lo que sí puedo decir, es, que como todo sistema de backups, **no sirve para nada si no existe una política de comprobación de las copias de seguridad**

Es decir, que tienes que probar en un cluster de pruebas, que esto te funciona, que puedes restaurarlo, que lo documentas, y que regularmente lo compruebas. De lo contrario, patatas como las que se comen en muchos sitios oficiales, hospitales españoles, o grandes empresas, por no tener una política de recuperación de desastres, unas veces por la impericia del supuesto Director de Informática, otras porque una empresa les vendido un cuento de seguridad, cuando en realidad, les engañaron y les cobraron, por nada.

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Consultas avanzadas de elasticsearch

Introducción

Colección de snippets para consultas avanzadas, con elasticsearch

Filtros

Por:

- id,
- con rango de fechas
- formateo de fechas unixtime a formato humano
- Selección de campos

```
GET work-analyzers/_search
{
  "_source": ["status_code", "total_consumption_ok", "response_time", "message", "log_id", "datetime"],
  "query": {
    "bool": {
      "must": [
        {
          "term": {
            "modem_id": {
              "value": "1544"
            }
          }
        },
        {
          "range": {
            "datetime": {
              "gte": "2023-05-08T19:00:00",
              "lte": "2023-05-08T23:00:00"
```

```
          }
        }
      }
    ]
  }
 },
 "docvalue_fields": [
  {
    "field": "datetime",
    "format": "yyyy-MM-dd HH:mm:ss"
  }
 ]
}
```

Instalando ElasticSearch + Kibana en local con Docker

Introducción

Al final con la aparecion de Laravel Herd y por las cosas que hago, preferia desmantelar en mis proyectos que no necesitan **Laravel sail** por ser modernos y actualizados. En cuanto a Elastic Stack preferi hacerlo via docker dado su caracter de uso puramente para testing., al menos de moemtno, en vez de optar. por una instalación local 100%.

Asi que te dejo como lo hice en mi mac silicon.

Instalación

Elastic search single node

Oficial

Crear un red para elastic

```
❯ docker network create elastic
```

Bajarse la imagen

```
❯ docker pull docker.elastic.co/elasticsearch/elasticsearch:8.11.2
8.11.2: Pulling from elasticsearch/elasticsearch
Digest: sha256:e40b9d3d523f2fe4dc851ad2cc5570f28a58ca6c4efb566cc9688dcaf0df8dec
Status: Image is up to date for docker.elastic.co/elasticsearch/elasticsearch:8.11.2
docker.elastic.co/elasticsearch/elasticsearch:8.11.2
```

Verificar la imagen

Se requiere tener instalado en tu entorno Cosign si quieres verificar la imagen.

- Primero bajarse la firma de la imagen

```
❯ wget https://artifacts.elastic.co/cosign.pub
--2023-12-10 09:17:56--  https://artifacts.elastic.co/cosign.pub
Resolviendo artifacts.elastic.co (artifacts.elastic.co)... 34.120.127.130
Conectando con artifacts.elastic.co (artifacts.elastic.co)[34.120.127.130]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 178 [application/x-mspublisher]
Grabando a: «cosign.pub»

cosign.pub
100%[=============================================================================
==========>]     178  --.-KB/s    en 0s

2023-12-10 09:17:57 (28,3 MB/s) - «cosign.pub» guardado [178/178]
```

- Después verificarla

```
❯ cosign verify --key cosign.pub docker.elastic.co/elasticsearch/elasticsearch:8.11.2

Verification for docker.elastic.co/elasticsearch/elasticsearch:8.11.2 --
The following checks were performed on each of these signatures:
  - The cosign claims were validated
  - Existence of the claims in the transparency log was verified offline
  - The signatures were verified against the specified public key
```

Instalar elasticsearch container

Esta es una variante del manual. El motivo es claro. Es crear dos volumenes:

- Uno con el directorio donde se guardan los indices.
- Otro donde se guarda la configuración.

> Si tenias una isntalación antigua tipo, o una instalación con volumenes diferentes, puede que tengas que empezar de o eliminándolos.

❯ docker run --name es01 --net elastic -p 9200:9200 -v es01-data:/usr/share/elasticsearch/data -v es01-config:/usr/share/elasticsearch/config -it -m 1GB docker.elastic.co/elasticsearch/elasticsearch:8.11.2

...

 Elasticsearch security features have been automatically configured!

 Authentication is enabled and cluster connections are encrypted.

ℹ Password for the elastic user (reset with `bin/elasticsearch-reset-password -u elastic`):
w78bj=MMasWeb6S1mnZR

ℹ HTTP CA certificate SHA-256 fingerprint:
464224b5bc542f16b37d86d9038633bfe0a9a01acaab7fe7fe78ff6e838f60dc

ℹ Configure Kibana to use this cluster:

• Run Kibana and click the configuration link in the terminal when Kibana starts.

• Copy the following enrollment token and paste it into Kibana in your browser (valid for the next 30 minutes):

eyJ2ZXIiOiI4LjExLjIiLCJhZHIiOlsiMTcyLjI3LjAuMjo5MjAwIl0sImZnciI6IjQ2NDIyNGI1YmM1NDJmMTZiMzdkODZkOTAzODYzM2JmZTBhOWEwMWFjYWFiN2ZlN2ZlNzhmZjZlODM4ZjYwZGMiLCJrZXkiOiIxaXpkVW93QjJlZDFvZmRMRTBCUTo2ZWVtYmJLZ1FkT1U4N1YyN0pkQnpBIn0=

ℹ Configure other nodes to join this cluster:

• Copy the following enrollment token and start new Elasticsearch nodes with `bin/elasticsearch --enrollment-token <token>` (valid for the next 30 minutes):

eyJ2ZXIiOiI4LjExLjIiLCJhZHIiOlsiMTcyLjI3LjAuMjo5MjAwIl0sImZnciI6IjQ2NDIyNGI1YmM1NDJmMTZiMzdkODZkOTAzODYzM2JmZTBhOWEwMWFjYWFiN2ZlN2ZlNzhmZjZlODM4ZjYwZGMiLCJrZXkiOiIyQ3pkVW93QjJlZDFvZmRMRTBCZzp4ZlNfc3R5QlRtYVFzUVZBbFBGcE13In0=

If you're running in Docker, copy the enrollment token and run:
`docker run -e "ENROLLMENT_TOKEN=<token>" docker.elastic.co/elasticsearch/elasticsearch:8.11.2`

Instalar kibana

Bajar la imagen

```
docker run --name kib01 --net elastic -p 5601:5601 -v kib01-config:/usr/share/kibana/config
docker.elastic.co/kibana/kibana:8.11.2
```

```
...

[2023-12-10T08:37:49.859+00:00][INFO ][root] Holding setup until preboot stage is completed.


i Kibana has not been configured.

Go to http://0.0.0.0:5601/?code=812523 to get started.
```

Acudiremos al navegador con la url mostrada, y alli usaremos el **enrollment token** para terminar la configuración.

Con eso ya tendremos configurado Kibana + Elasticsearch en docker.

Preparar las cosas para userse en nuestr app

Necesitamos el certificado de elastic

```
❯ export ELASTIC_PASSWORD="w78bj=MMasWeb6S1mnZR"
❯ docker cp es01:/usr/share/elasticsearch/config/certs/http_ca.crt ~/Sites/certificates/http_ca_local.crt
Successfully copied 3.58kB to /Users/abkrim/Sites/certificates/http_ca_local.crt
❯ export ELK_DOCKER_CA_BUNDLE="/Users/abkrim/Sites/certificates/http_ca_local.crt"
```

Yo lo hago así, guardar en una carpeta especial el certificado, porque uso un único docker para cada servicio del stack de Elasticsearch, para todos mis proyectos de desarrollo.

Prueba de concpeto

Una vez arriba ambos containers puedo probar elastic en el shell, y auq e asi sabre si luego la configuración de mis app de Laravel pasaran con lso datos que tengo.

> Uso dos variables de shell, evidentemente

```
❯ curl --cacert $ELK_DOCKER_CA_BUNDLE -u elastic:$ELASTIC_PASSWORD https://localhost:9200

{
```

```
  "name" : "9618d13f0939",
  "cluster_name" : "docker-cluster",
  "cluster_uuid" : "-wSu0Nm-QviovkB3rW6f5w",
  "version" : {
    "number" : "8.11.2",
    "build_flavor" : "default",
    "build_type" : "docker",
    "build_hash" : "76013fa76dcbf144c886990c6290715f5dc2ae20",
    "build_date" : "2023-12-05T10:03:47.729926671Z",
    "build_snapshot" : false,
    "lucene_version" : "9.8.0",
    "minimum_wire_compatibility_version" : "7.17.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "You Know, for Search"
}
```


Con eso ya tengo verificación de os datos que necesita mi app para funcionar en local con mis dockers del stack de elasticsearch.

Upgrade

En caso de trabajar sin `docker compose` Ver Arranque, Actualización, y cosas de Elastic con Docker

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Truncate index

Truncate index.

Reindexacion conservando el mapping

No probado.

```
$client = ClientBuilder::create()->build();

// Crear un nuevo índice temporal
$tempIndex = 'temp_index';
$client->indices()->create(['index' => $tempIndex]);

// Reindexar desde el índice temporal al índice original (sin documentos)
$response = $client->reindex([
    'body' => [
        'source' => [
            'index' => $tempIndex
        ],
        'dest' => [
            'index' => 'nombre_de_tu_indice'
        ]
    ]
]);

// Eliminar el índice temporal
$client->indices()->delete(['index' => $tempIndex]);
```

Borrado y receacion

```
$client = ClientBuilder::create()->build();
```

```
// Eliminar el índice
$client->indices()->delete(['index' => 'nombre_de_tu_indice']);

// Recrear el índice
$client->indices()->create(['index' => 'nombre_de_tu_indice']);
```

Arranque, Actualización, y cosas de Elastic con Docker

Introducción

De vez en cuando hay que actualizar las herramientas del paquete Elasticsearch. Una de ellas es la actualización de Elasticsearch cuando usamos Docker.

Como ya no uso Sail, esto lo hago con Docker, y para que no se me pase, dejo escrito algunos tips.

Discos Shared

Editado el 23/07/24 Version 8.14.3

He dejado de usar el data shared debido a que la actualización 8.14.X no pasa con el data. No tengo tiempo.

Estaba usando la configuración en modo shared, y esto en la última actualización me trajo líos. `/usr/share/kibana/config` por un problema con los certificados.

Para no perder el tiempo, opté por eliminar el disco de la configuración desde mi **Docker Desktop** .

Lo único es que hay que hacer de nuevo el enrollment de Kibana con Elastic, pero es súper sencillo y solo es seguir las instrucciones del navegador, y ejecutar los dos comandos: uno en el shell de Elasticsearch y otro en el de Kibana.

Backup Editado 05/10/2024

Como las versiones van subiendo, es bueno tener si en tu entorno de desarrollo, ya tienes cosas que no quieres volver a importar o dedicar tiempo, hacer un backup del contenedor.

Como en mi caso, los datos estan contenidos en un contedor que podemos hacer un backup:

```
docker export es01 -o /path_backup/es01_backup.tar
```

Si alguna vez queremos restaurarlo:

```
cat /path_backup/es01_backup.tar | docker import - es01
```

Error en la actualizacion

```
[2024-02-09T15:45:45.275+00:00][INFO ][http.server.Preboot] http server running at http://0.0.0.0:5601
[2024-02-09T15:45:45.579+00:00][INFO ][root] Kibana is shutting down
[2024-02-09T15:45:45.598+00:00][FATAL][root] Reason: ENOENT: no such file or directory, open
'/usr/share/kibana/data/ca_1702197563664.crt'
Error: ENOENT: no such file or directory, open '/usr/share/kibana/data/ca_1702197563664.crt'
    at Object.openSync (node:fs:603:3)
    at readFileSync (node:fs:471:35)
    at readFile (/usr/share/kibana/node_modules/@kbn/core-elasticsearch-server-
internal/src/elasticsearch_config.js:524:31)
```

Como veis, el proceso no actualiza el fichero de configuración y el cambio en esta versión va relacionado con los certificados. Seguramente se podrá solventar de otra manera, pero he preferido ir a lo rápido.

Lo primero es eliminar el volumen de configuración, ya que no es apropiado.

Arranque para actualizar

Aconsejable leeer:

- Docker Elasticsearch
- Docker Kibana

Pero aconsejable leer mejor el enlace a la documentación oficial, pues hay una gracia en el ejemplo `-p 9200:9200 -p 9300:9300`. que hara que fracase tu instalación y comiences a dar vueltas en el mundo de Elasticsearch.

Editado el 23/07/24 Version 8.14.3

He eliminado el volumen data, pues eso hace fallar la instalacion. No tengo tiempo de revisarlo.

```
tag="8.14.3" // Tag de la versión que queremos actualziar
docker rm es01 // NO queremos el container
docker volume rm es01-config // NO queremos el volumen de la config
docker run --name es01 --net elastic -p 9200:9200 -it -m 1GB docker.elastic.co/elasticsearch/elasticsearch:$tag
docker run --name kib01 --net elastic -p 5601:5601  docker.elastic.co/kibana/kibana:$tag
```

En kibana tendremos un mensaje final

```
[2024-05-11T06:07:59.739+00:00][INFO ][preboot] "interactiveSetup" plugin is holding setup: Validating
Elasticsearch connection configuration...
[2024-05-11T06:07:59.781+00:00][INFO ][root] Holding setup until preboot stage is completed.



i Kibana has not been configured.

Go to http://0.0.0.0:5601/?code=811436 to get started.
```

Esto nos obliga a obtener el enrollment en la instancia de Elasticsearch

```
sh-5.0$ bin/elasticsearch-create-enrollment-token --scope kibana
eyJ2ZXIiOiI4LjEzLjQiLCJhZHIiOlsiMTcyLjI3LjAuMjo5MjAwIl0sImZnciI6IjQ2NDIyNGI1YmM1NDJmMTZiMzdkODZkOTAz
ODYzM2JFKEFAKEFAKEFAKLEFAKErZXkiOiIyVFpJWm84QnlVV1BJX0FsYTdPRDowTVJ3WGR3RFQzaTI2ZXVzWllCNllBI
n0=
```

> Si por alguna razón, falla es debido a que arrancaste ambas instancias de manera distinta y no se pueden comunicar. Arrancalas al unisono o primero elastic y luego kibana, en un mismo comando, y recuerda que deberas cambiar la url del enrollment, porque cambia en cada arranque.

Aviso

Llamadas en Kibana (o para usar con cUrl) para Elasticsearch de uso común

Introducción

Hay una serie de llamadas con curl o Kibana que son de uso rápido. Ayudan mucho en el día a día con Elasticsearch.

> Las llamdas fuera de kibna requiere ser adptadas a sus posibilidades. Ojo a esto, que muchos con el copy & paste no lo comprenden.

Convenciones

- **[sutituir]** incluidos los `[]` por el valor correpondiente
- En cUrl, con el shell, `${var}` necesitamos declrar la variable.

Obtener el mapping de un indice

```
GET [indice]/_mapping
```

Estado del cluster (Solo cUrl)

Esta solo es para cUrl.

```
 curl --cacert "${path_http_ca.crt}"  --user ${user}: -XGET
"{$url_port}/_cluster/health?wait_for_status=yellow&timeout=50s&pretty"
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.