Занятие 12: Elastic stack (ELK)

0. Elastic

Базовая информация об установке - в документации
https://www.elastic.co/guide/en/elasticsearch/reference/current/install-elasticsearch.html

Здесь мы работаем с виртуальной машину на основе бокса **Dealmi/ubuntu20_elk_agent_7.17**, где ELK уже установлен. В ней создано несколько базовых пользователей, пароли к ним лежат в самой ВМ:

```
cat /home/vagrant/elasticpass
```

```
...
password for user apm_system PASSWORD apm_system = P5gaNF9LvBHJlxuLVTWh
password for user kibana_system PASSWORD kibana_system = nzggr8UN6iC575Qp8B6E
password for user kibana PASSWORD kibana = nzggr8UN6iC575Qp8B6E
password for user logstash_system PASSWORD logstash_system = ygiMdq6uEOgDKAWkieu8
password for user beats_system PASSWORD beats_system = eCv8KPb363JX9bgRLW8e
password for user remote_monitoring_user PASSWORD remote_monitoring_user = bQD4c1A8PhX2e37oEAay
password for user elastic PASSWORD elastic = EHukauEaaoy3UnvoFXea
```

Для целей урока разрешим анонимный доступ с ролью superuser (см. в документации страницы anonymous-access и built-in-roles).

Для этого в файл конфигурации нужно добавить следующий текст:
```
sudo nano /etc/elasticsearch/elasticsearch.yml
```

```
---
xpack.security.authc:
  anonymous:
    username: _es_anonymous_user
    roles: superuser
    authz_exception: true
...
```

Чтобы попасть в UI (Kibana) нужно зайти на порт 5601 (пароль пользователя *elastic* – в файле *elasticpass*): http://localhost:5601/

ElasticSearch - позиционируется как документоориентированный NoSQL-сервис, масштабируется, устойчив, работает по RESTful API. Все данные, поступающие в ElasticSearch, сохраняются через JSON.

Основное место хранения - ноды, элементы хранения - индексы, внути индексов - кусочки, шарды - это сами места хранения. Хорошее описание основ можно прочитать здесь: С чего начинается Elasticsearch / Хабр (habr.com).

1. Демо REST API, CRUD

Далее используется Curl, как вариант, можно взять GUI-based клиент, например Insomnia Rest client https://insomnia.rest/download/#windows.

Открываем сеанс работы с ВМ:
```
vagrant ssh
```

Info: Status of your cluster (ElasticSarch)
```
curl -X GET http://localhost:9200/_cluster/health?pretty
```

```
{ "cluster_name": "elasticsearch",
...}
```

Есть дополнительные сервисные команды, которые выводят состояние самого ElasticSarch. Например, команда **cat**.

Display list of indices
```
curl -X GET http://localhost:9200/_cat/indices
```

Create index (using PUT)
curl -X DELETE 'http://localhost:9200/testline'

curl -X PUT 'http://localhost:9200/testline'

Создать документ и скормить ему JSON
```
curl -X PUT 'http://localhost:9200/testline/_doc/1' -H 'Content-Type:
application/json' -d '{ "intfield": 98, "stringfield": "Nine and Eight"
}'
```

Обновить документ частично
```
curl -X POST 'http://localhost:9200/testline/_update/1' -H 'Content-Type:
application/json' -d '{ "doc": {"stringfield": "Ninety-Eight" }}'
```

Пересоздать индекс
```
curl -X DELETE 'http://localhost:9200/testline2'
curl -X PUT 'http://localhost:9200/testline2'
```

Создать документ и скормить ему JSON
```
curl -X PUT 'http://localhost:9200/testline2/record/1' -H 'Content-Type:
application/json' -d '{ "intfield": 98, "stringfield": "Nine and Eight"
}'
```

Обновить документ частично
```
curl -X POST 'http://localhost:9200/testline2/_update/1' -H 'Content-
Type: application/json' -d '{ "doc": {"stringfield": "Ninety-Eight" }}'
```

```
curl -X PUT http://localhost:9200/presentations { "settings": {},
mappings {} }
```

```
{ "acknowledged": true,
  "shards_acknowledged": true,
  "index": "presentations"
}
```

Посмотреть маппинг
```
curl -X GET http://localhost:9200/presentations/_mapping/doc
```

Создать документ и скормить ему JSON
```
curl -X PUT http://localhost:9200/presentations/lections/1 -d '{ "date":
"2022-05-25",   "lector": "First Author", "window": { "title": "Sample
Konfabulator Widget", "name": "main_window", "width": 500, "height": 500
}, "image": { "src": "Images/Sun.png", "name": "sun1", "hOffset": 250,
```

```
"vOffset": 250, "alignment": "center" }, "text": { "data": "Click Here",
"size": 36, "style": "bold", "name": "text1", "hOffset": 250, "vOffset":
100, "alignment": "center", "onMouseUp": "sun1.opacity = (sun1.opacity /
100) * 90;" }}' -H 'Content-Type: application/json'

curl -X PUT http://localhost:9200/presentations/lections/2 -d '{ "date":
"2022-05-27",   "lector": "Second Author", "window": { "title": "More
donuts!", "name": "main_window", "width": 500, "height": 500 }, "image":
{ "src": "Images/Moon.png", "name": "moon1", "hOffset": 250, "vOffset":
250, "alignment": "center" }, "text": { "data": "Click Here", "size": 36,
"style": "italic", "name": "text1", "hOffset": 250, "vOffset": 100,
"alignment": "center", "onMouseUp": "moon1.opacity = (moon1.opacity /
100) * 50;" }}' -H 'Content-Type: application/json'
```

Запросить документ
```
curl -X GET http://localhost:9200/presentations/lections/1?pretty
```

Update - то же самое, что Insert
```
curl -X PUT http://localhost:9200/presentations/lections/2 -d '{ "date":
"2022-05-27","lector": "Second Author", "window": { "title": "More donuts
NOW!", "name": "main_window", "width": 500, "height": 500 }}' -H
'Content-Type: application/json'
```

```
    {...result: updated...}
```

Запросить обновленный документ
```
curl -X GET http://localhost:9200/presentations/lections/2?pretty
```

Теперь документ ПЕРЕЗАПИСАН!

Обновление части документа - метод POST
```
curl -X POST http://localhost:9200/presentations/_update/2 -d '{ "doc":
{"date": "2022-09-01"} }' -H 'Content-Type: application/json'
```

```
    {...result: updated...}
```

Можно использовать скрипты: https://www.elastic.co/guide/en/elasticsearch/reference/current/docs-update.html. Скриптовый язык называется Painless и его использование требует внимания к ресурсам, т.к. необходимо обработать каждую выводимую запись.

Запросить обновленный документ
```
curl -X GET http://localhost:9200/presentations/lections/2?pretty
```

Выбрать только часть полей
```
curl -X GET
http://localhost:9200/presentations/_doc/2?_source=date,lector,window.tit
le
```

Удаление индекса
```
curl -X DELETE http://localhost:9200/presentations
```

```
    { "acknowledged": true }
```

В списке индексов его больше нет:
```
curl -X GET 'http://192.168.3.33:9200/_cat/indices?v=true&s=index:desc'
```

2. Demo

@elkserver

Установка Logstash (в текущих условиях – с репо Яндекса):

```
sudo rm /etc/apt/sources.list.d/elastic-7.x.list
echo "deb [trusted=yes] https://mirror.yandex.ru/mirrors/elastic/8/ \
     stable main" | sudo tee /etc/apt/sources.list.d/elastic-8.x.list
sudo apt-get update && sudo apt-get install logstash
sudo /bin/systemctl daemon-reload && \
     sudo /bin/systemctl enable logstash.service
```

Создаем файл конфигурации для Logstash, где будет указан импорт логов из папки */home/vagrant/demo/tests*:

```
cp /vagrant/Tests/logstash.conf /tmp/logstash.conf
cat /tmp/logstash.conf
```

```
tail /vagrant/generators/access_log_20220525-081734.log
```

Лог (типа NGINX) – это текст, а ElasticSearch работает только с JSON, поэтому требуется парсинг и перекодировка.

Очищаем целевой файл (вначале его нет)

```
mkdir -p /home/vagrant/demo/tests
cat /dev/null > /home/vagrant/demo/tests/access_log.log
```

Запус логстеша из командной строки (сервис сейчас остановлен). Запускаем так, чтобы видеть stdout

```
sudo /usr/share/logstash/bin/logstash --verbose -f /tmp/logstash.conf
```

На конфигурации с несколькими машинами необходимо проверить доступность порта 9600 у ElasticSearch.

@Kibana UI

Management -> Stack Management -> Data / Index Management -> пока пусто (кроме данных демо-дашборда)

@Console2

```
cat /vagrant/generators/access_log_20220525-081734.log >
/home/vagrant/demo/tests/access_log.log
```

Видим на терминале **@elkserver** побежали JSONчики

@Kibana UI

Management -> Stack Management -> Data / Index Management -> появились индексы, виден счетчик статистики

Весь список событий смотреть не надо, Кибана не может отобразить все – значит, нужна выборка, запрос.

Management -> Kibana / Index Patterns -> Create Index Pattern

Pattern: http-access- (тут оно подсказывает, справа появляются выбранные индексы)*

Time filter field: @timestamp

Следующий инструмент, основной инструмент для работы с индексами: **Discover**
Здесь на временной шкале количество доков по времени поступления

Выбор диапазона отображения: выбрать *2022 год -> Refresh*
Слева - поля, управление фильтрацией и таблицей.

В центре – документ. Можно отобразить в формате полей и в JSON.
Мжоно добавлять фильтры по значению (слева у поля в документе)

Пользуясь кнопкой Add (+), которая есть около каждого поля, добавляем в таблицу нужные поля, например: *method, client_ip, response, host.*
Save -> сохранить запрос

Можно добавлять фильтры формой в верхней части страницы: *server_response.keyword is 200 или IS NOT 200, method.keyword is GET*

Заметим, что тип данных у поля server_response текстовый. Это произошло потому, что маппинг мы не задали и все значения получили типы данных по-умолчанию. Изменим это.

Перейдем в Management -> Stack Management -> Data / Index Management -> Index Templates -> создадим новый шаблон, назовем *http_access_template*, добавим в маппинг *server_response -> Numeric -> Short u client_ip -> IP*
Удалим все индексы (они не поменяют маппинг на лету) и загрузим заново.

```
cat /dev/null > /home/vagrant/demo/tests/access_log.log
cat /vagrant/generators/access_log_20220525-081734.log >
/home/vagrant/demo/tests/access_log.log
```

Видим в сохраненном запросе возможность задать диапазон для числового поля в фильтре.

Работа с визуализацией

*Dashboard -> New Dashboard -> New visualization -> Pie -> http-access-**
>Size by: Count
>Slice by:
Field: server_response
Value format: Default
* Смотрим красивый пирог