

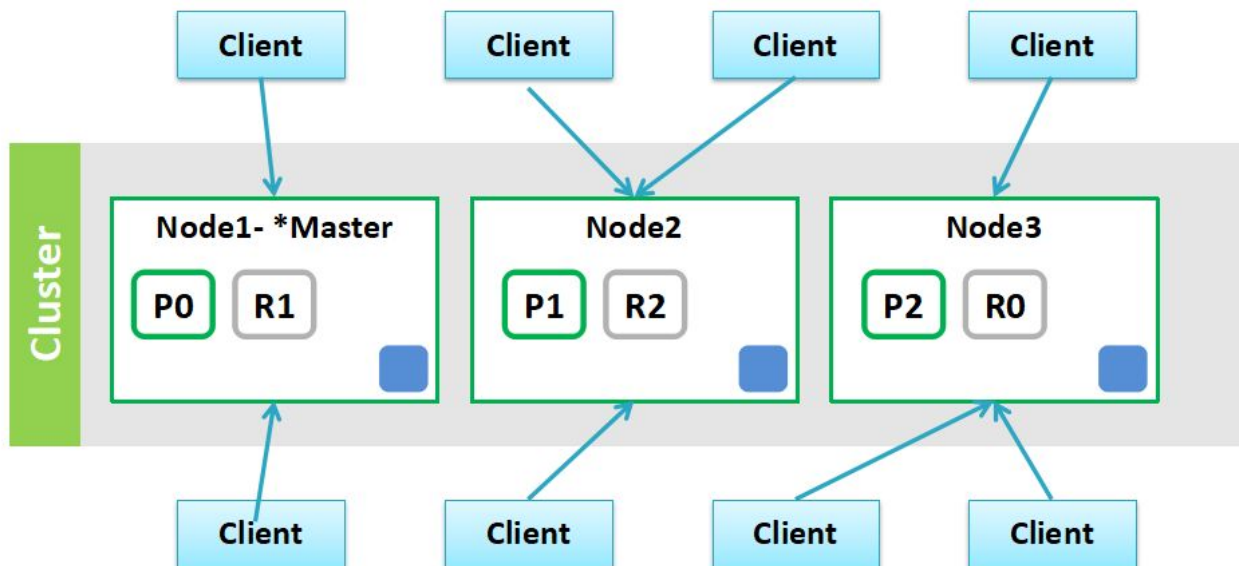
1. 为什么要使用ES集群架构

分布式系统的可用性与扩展性

- 高可用性
 - 服务可用性——允许有节点停止服务
 - 数据可用性——部分节点丢失, 不会丢失数据
- 可扩展性
 - 请求量提升/数据的不断增长(将数据分布到所有节点上)

ES集群架构的优势:

- 提高系统的可用性: 在ES集群中, 即使部分节点停止服务, 整个集群的服务也不会受到影响, 因为数据和索引操作可以在剩余的节点上继续进行。
- 存储的水平扩容: ES集群支持通过增加新的节点来扩展存储容量, 实现数据的水平扩展, 这样可以有效应对数据量的增长。



2. 核心概念

集群

- 一个集群可以有一个或者多个节点
- 不同的集群通过不同的名字来区分, 默认名字“elasticsearch”
- 通过配置文件修改, 或者在命令行中 `-E cluster.name=es-cluster` 进行设定

节点

- 节点是一个Elasticsearch的实例
 - 本质上就是一个JAVA进程
 - 一台机器上可以运行多个Elasticsearch进程，但是生产环境一般建议一台机器上只运行一个Elasticsearch实例
- 每一个节点都有名字，通过配置文件配置，或者启动时候 -E node.name=node1指定
- 每一个节点在启动之后，会分配一个UID，保存在data目录下

分片(Primary Shard & Replica Shard)

- 主分片 (Primary Shard)
 - 用以解决数据水平扩展的问题。通过主分片，可以将数据分布到集群内的所有节点之上
 - 一个分片是一个运行的Lucene的实例
 - 主分片数在索引创建时指定，后续不允许修改，除非Reindex
- 副本分片 (Replica Shard)
 - 用以解决数据高可用的问题。副本分片是主分片的拷贝
 - 副本分片数，可以动态调整
 - 增加副本数，还可以在在一定程度上提高服务的可用性(读取的吞吐)

```
1 # 指定索引的主分片和副本分片数
2 PUT /blogs
3 {
4   "settings": {
5     "number_of_shards": 3,
6     "number_of_replicas": 1
7   }
8 }
```

分片架构

集群status

- Green: 主分片与副本都正常分配
- Yellow: 主分片全部正常分配，有副本分片未能正常分配
- Red: 有主分片未能分配。例如，当服务器的磁盘容量超过85%时,去创建了一个新的索引

- 1 #查看集群的健康状况
- 2 GET _cluster/health

CAT API查看集群信息

- 1
- 2 GET /_cat/nodes?v #查看节点信息
- 3 GET /_cat/health?v #查看集群当前状态：红、黄、绿
- 4 GET /_cat/shards?v #查看各shard的详细情况
- 5 GET /_cat/shards/{index}?v #查看指定分片的详细情况
- 6 GET /_cat/master?v #查看master节点信息
- 7 GET /_cat/indices?v #查看集群中所有index的详细信息
- 8 GET /_cat/indices/{index}?v #查看集群中指定index的详细信息

3. 搭建三节点ES集群

建议：每台机器先安装好单节点ES进程，并能正常运行，再修改配置，搭建集群

参考课程：[ElasticSearch快速安装上手](#)

IP	ES节点名
192.168.65.213	node-1
192.168.65.207	node-2
192.168.65.208	node-3

ES集群搭建步骤

1) 系统环境准备

安装版本：elasticsearch8.14.3

操作系统: CentOS7

切换到root用户，创建用户es

```
1 adduser es
2 passwd es
```

修改/etc/hosts

```
1 vim /etc/hosts
2 192.168.65.213 es-node1
3 192.168.65.207 es-node2
4 192.168.65.208 es-node3
```

关闭防火墙

```
1 #查看防火墙状态
2 systemctl status firewalld
3 #关闭防火墙
4 systemctl stop firewalld
5 systemctl disable firewalld
```

在生产模式下，服务启动会触发ES的引导检查或者叫启动检查（bootstrap checks），所谓引导检查就是在服务启动之前对一些重要的配置项进行检查，检查其配置值是否是合理的。引导检查包括对JVM大小、内存锁、虚拟内存、最大线程数、集群发现相关配置等相关的检查，如果某一项或者几项的配置不合理，ES会拒绝启动服务。

[1]: max file descriptors [4096] for elasticsearch process is too low, increase to at least [65536]
ES因为需要大量的创建索引文件，需要大量的打开系统的文件，所以我们需要解除linux系统当中打开文件最大数目的限制，不然ES启动就会抛错

```
1 #切换到root用户
2 vim /etc/security/limits.conf
3
4 末尾添加如下配置:
5 *      soft    nofile  65536
6 *      hard    nofile  65536
7 *      soft    nproc   4096
8 *      hard    nproc   4096
```

[2]: max number of threads [1024] for user [es] is too low, increase to at least [4096]

无法创建本地线程问题,用户最大可创建线程数太小

```
1 vim /etc/security/limits.d/20-nproc.conf
2
3 改为如下配置:
4 * soft nproc 4096
```

[3]: max virtual memory areas vm.max_map_count [65530] is too low, increase to at least [262144]

最大虚拟内存太小,调大系统的虚拟内存

```
1 vim /etc/sysctl.conf
2 追加以下内容:
3 vm.max_map_count=262144
4 保存退出之后执行如下命令:
5 sysctl -p
```

2) 切换到es用户, 修改elasticsearch.yml

```
1 # 指定集群名称3个节点必须一致
2 cluster.name: es-cluster
3 #指定节点名称, 每个节点名字唯一
4 node.name: node-1
5 # 绑定ip,开启远程访问,可以配置0.0.0.0
6 network.host: 0.0.0.0
7 #指定web端口
8 #http.port: 9200
9 #指定tcp端口
10 #transport.tcp.port: 9300
11 #用于节点发现, 一般配置集群的候选主节点
12 discovery.seed_hosts: ["es-node1", "es-node2", "es-node3"]
13 #7.0新引入的配置项,集群引导节点。指定集群初次选举中用到的具有主节点资格的节
14 #点称为集群引导节点, 只在第一次形成集群时需要
15 #该选项配置为node.name的值, 指定可以初始化集群节点的名称
16 cluster.initial_master_nodes: ["node-1", "node-2", "node-3"]
17 #解决跨域问题
18 http.cors.enabled: true
19 http.cors.allow-origin: "*"
20 #初学者建议关闭security安全认证
21 xpack.security.enabled: false
```

三个节点配置如下:

```
1 #192.168.65.213的配置
2 cluster.name: es-cluster
3 node.name: node-1
4 network.host: 0.0.0.0
5 discovery.seed_hosts: ["es-node1", "es-node2", "es-node3"]
6 cluster.initial_master_nodes: ["node-1", "node-2", "node-3"]
7 http.cors.enabled: true
8 http.cors.allow-origin: "*"
9 xpack.security.enabled: false
10
11 #192.168.65.207的配置
12 cluster.name: es-cluster
13 node.name: node-3
14 network.host: 0.0.0.0
15 discovery.seed_hosts: ["es-node1", "es-node2", "es-node3"]
16 cluster.initial_master_nodes: ["node-1", "node-2", "node-3"]
17 http.cors.enabled: true
18 http.cors.allow-origin: "*"
19 xpack.security.enabled: false
20
21 #192.168.65.208的配置
22 cluster.name: es-cluster
23 node.name: node-2
24 network.host: 0.0.0.0
25 discovery.seed_hosts: ["es-node1", "es-node2", "es-node3"]
26 cluster.initial_master_nodes: ["node-1", "node-2", "node-3"]
27 http.cors.enabled: true
28 http.cors.allow-origin: "*"
29 xpack.security.enabled: false
```

3) 启动每个节点的ES服务

```
1 # 注意：如果运行过单节点模式，需要删除data目录， 否则会导致无法加入集群
2 rm -rf data
3 #安装ik分词器
4 bin/elasticsearch-plugin install https://release.infinilabs.com/analysis-ik/stable/elasticsearch-analysis-ik-8.14.3.zip
5 # 启动ES服务
6 bin/elasticsearch -d
```

4) 验证集群

http://192.168.65.213:9200/_cat/nodes?pretty

安装Cerebro客户端

Cerebro介绍

Cerebro 可以查看分片分配和通过图形界面执行常见的索引操作。完全开源，并且它允许添加用户，密码或 LDAP 身份验证问网络界面。

Cerebro 基于 Scala 的Play 框架编写，用于后端 REST 和 Elasticsearch 通信。它使用通过 AngularJS 编写的单页应用程序（SPA）前端。

项目网址：<https://github.com/lmenezes/cerebro>

安装 Cerebro

下载地址：<https://github.com/lmenezes/cerebro/releases/download/v0.9.4/cerebro-0.9.4.zip>

运行 cerebro

```
1 cerebro-0.9.4/bin/cerebro
2
3 #后台启动
4 nohup bin/cerebro &
```

访问：<http://192.168.65.207:9000/>

输入ES集群节点：<http://192.168.65.207:9200>，建立连接：

安装kibana

1) 修改kibana配置


```
1 vim config/kibana.yml
2
3 server.host: "192.168.65.213"
4 i18n.locale: "zh-CN"
```

2) 运行Kibana

提示：Kibana对外的 tcp 端口是5601，使用netstat -tunlp|grep 5601即可查看进程

```
1 #后台启动
2 nohup bin/kibana &
3
4 #查询kibana进程
5 netstat -tunlp | grep 5601
```

访问Kibana: <http://192.168.65.213:5601/>

4. ES集群安全认证

参考文档: <https://www.elastic.co/guide/en/elasticsearch/reference/8.14/configuring-stack-security.html>

近几年来，ES 数据泄露事件频发给国内各行业用户敲响了数据安全的警钟。比如：

- 2019 年发生的 ES 数据泄露事件，泄露包括 27 亿个电子邮件地址，其中 10 亿个密码是以简单的明文存储，涉及国内多家互联网公司。
- 2021 年 Group-IB 报告显示，网络上暴露的 ES 实例超过 10 万个，约占 2021 年暴露数据库总数的 30%。
- 2022 年漫画阅读平台 Mangatoon 遭遇数据泄露，黑客从不安全的 ES 数据库中窃取了属于 2300 万用户帐户的信息。
- 2022 年阿里巴巴遭受了一次重大数据泄露，涉及客户数据包括：姓名、电话号、身份证号、居住地址等信息共计 23TB。

ES敏感信息泄露的原因

- Elasticsearch在安装后，不提供任何形式的安全防护
- 不合理的配置导致公网可以访问ES集群。比如在elasticsearch.yml文件中,server.host配置为0.0.0.0

基于Security的安全认证

ES 8 默认启动了Security。ES 8.x 第一次启动之后会输出以下信息，此时服务已经启动成功了。

比如windows下第一次启动ES，会输出如下信息：

```

1 -----
2 -> Elasticsearch security features have been automatically configured!
3 -> Authentication is enabled and cluster connections are encrypted.
4
5 -> Password for the elastic user (reset with `bin/elasticsearch-reset-password -u
    elastic`):
6     GFDGvf9kEuSaZrr=3eLt
7
8 -> HTTP CA certificate SHA-256 fingerprint:
9     f76d093b63225ea0866b4fcc1766293caf05c6ae152a9e95e3149afd74be5fa8
10
11 -> Configure Kibana to use this cluster:
12 * Run Kibana and click the configuration link in the terminal when Kibana starts.
13 * Copy the following enrollment token and paste it into Kibana in your browser (valid
    for the next 30 minutes):
14
    eyJ2ZXIiOiI4LjE0LjAiLCJhZHIIiOlSIiMTcyLjE5LjE3Ni4xOjkyMDAiXSwiZmdyIjoiZjc2ZDA5M2I2MzIyNWVhMDg2NmI0ZmNjMTc2NjI5M2NhZjA1YzZhZTE1MmE5ZTk1ZTMxNDlhZmQ3NGJlNWZhOCIsImtleSI6IjI1VW1jSkVCaXNrRWNRdjRYMXVzOlRWQjlMS2RwUkRTT2hjUmhwVGZF2cUEifQ==
15
16 -> Configure other nodes to join this cluster:
17 * On this node:
18     - Create an enrollment token with `bin/elasticsearch-create-enrollment-token -s
        node`.
19     - Uncomment the transport.host setting at the end of config/elasticsearch.yml.
20     - Restart Elasticsearch.
21 * On other nodes:
22     - Start Elasticsearch with `bin/elasticsearch --enrollment-token <token>`, using the
        enrollment token that you generated.
23 -----

```

首次启动 Elasticsearch 时，会自动进行以下安全配置：

- 为传输层和 HTTP 层生成 TLS 证书和密钥。
- TLS 配置设置被写入 `elasticsearch.yml`。


- 为 elastic 用户生成密码。
- 为 Kibana 生成一个注册令牌。

修改账号密码

在 ES 8.x版本以后，elasticsearch-setup-passwords设置密码的工具已经被弃用删除，此命令为7.x之前第一次生成密码时使用，8.x在第一次启动的时候会自动生密码。

如果需要修改账户密码，需进行以下操作：

```
D:\vip7\elasticsearch\ELK安装包\8.14.3\elasticsearch-8.14.3\bin>elasticsearch-reset-password.bat -u elastic -i
warning: ignoring JAVA_HOME=C:\Program Files\Java\jdk-21.0.1; using ES_JAVA_HOME
This tool will reset the password of the [elastic] user.
You will be prompted to enter the password.
Please confirm that you would like to continue [y/N]y

Enter password for [elastic]:  输入自定义密码
Re-enter password for [elastic]:
Password for the [elastic] user successfully reset.
```

```
1 #为elastic账号自动生成新密码，输出至控制台
2 bin/elasticsearch-reset-password -u elastic
3 #手工指定用户的新密码
4 bin/elasticsearch-reset-password -u elastic -i
5 #指定服务地址和账户名
6 bin/elasticsearch-reset-password --url "https://ip:9200" -u elastic -i
```

验证服务状态

访问服务

在7.x的版本是通过如下地址访问ES服务：<http://localhost:9200/>

但是在 8.x 的版本访问会看到如下页面：

原因解释

这是正常现象，因为 Elastic 8 默认开启了 SSL，将默认配置项由true改为false即可

推荐做法

关闭SSL虽然可以访问服务了，但这本质上是在规避问题而非解决问题，更推荐的做法是使用https协议进行访问：

<https://localhost:9200/>，此时如果你的浏览器版本是比较新的版本会出现以下弹窗提示，即：

输入账号密码验证:

三节点ES集群增加安全认证

node-1增加安全认证

- 1) 停止集群所有节点，并删除data目录
- 2) 以node-1为例，修改config/elasticsearch.yml配置文件
- 3) 删除data目录（不删除会报错），然后启动node-1节点

```
1 bin/elasticsearch -d
```

查看elasticsearch.yml配置文件，多出很多security相关配置

- 4) 修改用户elastic的密码

```
1 bin/elasticsearch-reset-password -u elastic -i
```

- 5) 测试，访问<https://192.168.65.213:9200/>

输入用户名密码

node-2和node-3加入集群

- 1) 修改node-2和node-3的elasticsearch.yml配置文件

2) 向集群中加入新节点

默认情况下，要向集群中添加新节点，需要通过令牌来完成节点之间的通信

- 2.1) 在node-1中执行下面的命令为新节点生成注册令牌

```
1 bin/elasticsearch-create-enrollment-token -s node
```

- 2.2) 以node-2为例，启动node-2节点，并带上注册令牌

```
1 bin/elasticsearch --enrollment-token <enrollment-token> -d
```

同上，启动node-3节点，并带上注册令牌

注意：只有第一次加入集群需要带上注册令牌，后续启动不需要

2.3) 通过head插件查看集群

部署Kibana

1) 进入ES目录，生成kibana的注册令牌

```
1 bin/elasticsearch-create-enrollment-token -s kibana
```

2) 进入kibana目录，通过下面的命令注册 Kibana

```
1 bin/kibana-setup --enrollment-token <enrollment-token>
```

3) 直接启动Kibana服务

```
1 nohup bin/kibana &
```

然后我们访问Kibana：<http://192.168.65.213:5601/>

输入用户名elastic和密码，进入kibana主界面

部署cerebro

1) 修改配置文件

```
1 vim conf/application.conf
2
3 hosts = [
4   {
5     host = "https://192.168.65.207:9200"
6     name = "es-cluster"
7     auth = {
8       username = "elastic"
9       password = "123456"
10    }
11  }
12 ]
13
```

2) 启动cerebro服务

```
1 nohup bin/cerebro -Dplay.ws.ssl.loose.acceptAnyCertificate=true &
```

访问: <http://192.168.65.207:9000/>