ELK 其实并不是一款软件,而是一整套解决方案,是三个软件产品的首字母缩写

查看日志，日志分析

– Elasticsearch:负责日志检索和储存 , 坏了，数据全丢了 2-3台集群

– Logstash:负责日志的收集和分析、处理 , 坏了日志不更新

– Kibana:负责日志的可视化 docker 1G内存

192.168.1.11-15 Elasticsearch

192.168.1.16 Kibana

L A M P

K E L

ELK 能做什么?

• ELK组件在海量日志系统的运维中,可用于解决:

– 分布式日志数据集中式查询和管理

– 系统监控,包含系统硬件和应用各个组件的监控

– 故障排查

– 安全信息和事件管理

– 报表功能

Elasticsearch部分

ElasticSearch 是一个基于 Lucene 的搜索服务器。它提供了一个分布式多用户能力的全文搜索引擎,基于 RESTful API 的 web 接口。

• Elasticsearch是用Java开发的,幵作为Apache许可条款下的开放源码发布,是当前流行的企业级搜索引擎。设计用于云计算中,能够达到实时搜索,稳定,

可靠,快速,安装使用方便

• 主要特点

– 实时分析

– 分布式实时文件存储,幵将每一个字段都编入索引

– 文档导向,所有的对象全部是文档

– 高可用性,易扩展,支持集群(Cluster)、分片和复制(Shards 和 Replicas)

– 接口友好,支持 JSON

• ES 没有什么?

• Elasticsearch 没有典型意义的事务.

• Elasticsearch 是一种面向文档的数据库。

• Elasticsearch 没有提供授权和认证特性

• 相关概念:

– Node: 装有一个 ES 服务器的节点。

– Cluster: 有多个Node组成的集群

– Document: 一个可被搜素的基础信息单元 行

– Index: 拥有相似特征的文档的集合 表

– Type: 一个索引中可以定义一种或多种类型

– Filed: 是 ES 的最小单位,相当于数据的某一列 列

– Shards: 索引的分片,每一个分片就是一个 Shard

– Replicas: 索引的拷贝

• ES 不关系型数据库的对比

– 在 ES 中,文档归属于一种 类型 (type) ,而这些类型

存在于索引 (index) 中,类比传统关系型数据库

– DB -> Databases -> Tables -> Rows -> Columns

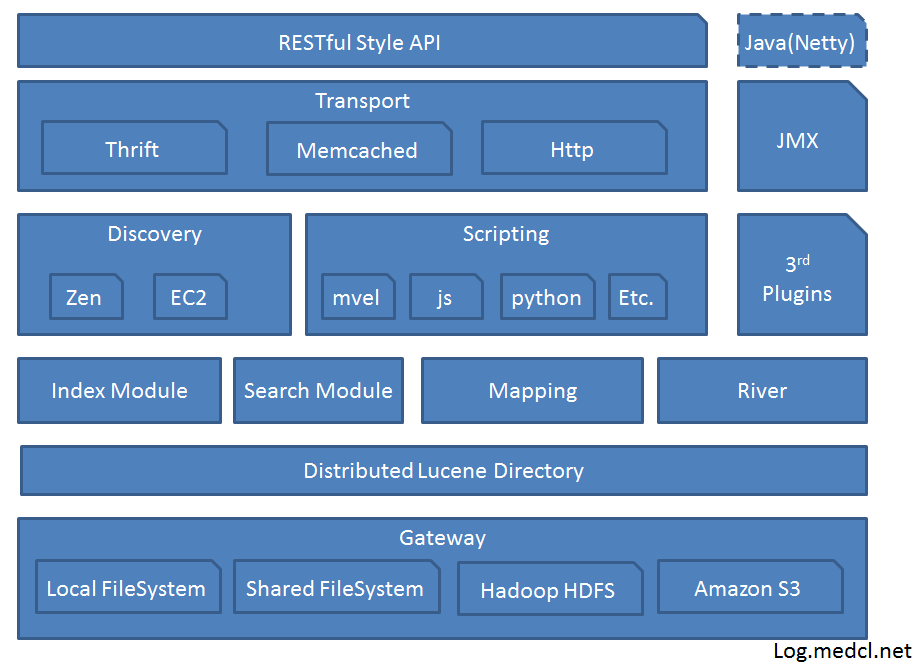
– 关系型 数据库 表 行 列

– ES -> Indices -> Types -> Documents -> Fields

– ES 索引 类型 文档 域(字段)

使用HTTP协议接口进行增、删、改、查

Elasticsearch架构图



ES 集群安装

每台机器彼此之前都能够通讯，所以要让一台主机都能知道集群中所有机器的IP地址

步骤 1,设置 ip 与主机名称对应关系,每台集群主机上都要设置

– 配置 /etc/hosts

192.168.1.11 es1

• 步骤 2,安装 JDK

– Elasticsearch 要求至少 Java 7

– 一般推荐使用 OpenJDK 1.8

– 配置好安装源以后,我们先解决依赖关系

yum install -y java-1.8.0-openjdk

• 步骤 3

– 安装 ES

rpm –ivh elasticsearch-2.3.4-1.noarch

• 步骤 4

– 修改配置文件

vim /etc/elasticsearch/elasticsearch.yml

network.host: 0.0.0.0 //监听哪个主机地址

• 步骤 5

– 启动服务,设置自启动

systemctl enable elasticsearch

systemctl start elasticsearch

– 验证:

netstat –ltunp | grep 9200|9300

– 能够看到 9200,9300 被监听

• 通过浏览器或 curl 访问 9200 端口

curl http://192.168.4.11:9200/

{

"name" : "Imperial Hydra",

"cluster\_name" : "elasticsearch",

"version" : {

"number" : "2.3.4",

"build\_hash" : "e455fd0c13dceca8dbbdbb1665d068ae55dabe3f",

"build\_timestamp" : "2016-06-30T11:24:31Z",

"build\_snapshot" : false,

"lucene\_version" : "5.5.0"

},

"tagline" : "You Know, for Search"

}

• ES 集群配置

– ES 集群配置也徆简单,只需要对配置文件做少量的修改即可,其他步骤和单机完全一致

– ES 集群配置文件

cluster.name: my-es //如果集群的名字相同就是在同一个集群中

node.name: elasticsearch1

network.host: 0.0.0.0

discovery.zen.ping.unicast.hosts: ["elasticsearch1","elasticsearch2","elasticsearch3"]

启动顺序非常重要，先声明的主机最先起来，后加入的会根据discovery.zen.ping.unicast.hosts配置的集群主机自动加入集群，重启服务

找个集群中的IP，查看集群状态

curl [http://192.168.1.11:9200/\_cluster/health?pretty](http://192.168.4.11:9200/_cluster/health?pretty)

{

"cluster\_name" : "my-es",

"status" : "green",

"timed\_out" : false,

"number\_of\_nodes" : 5,

"number\_of\_data\_nodes" : 5,

"active\_primary\_shards" : 0,

"active\_shards" : 0,

"relocating\_shards" : 0,

"initializing\_shards" : 0,

"unassigned\_shards" : 0,

"delayed\_unassigned\_shards" : 0,

"number\_of\_pending\_tasks" : 0,

"number\_of\_in\_flight\_fetch" : 0,

"task\_max\_waiting\_in\_queue\_millis" : 0,

"active\_shards\_percent\_as\_number" : 100.0

}

• http请求由三部分组成

– 分别是:请求行、消息报头、请求正文

– 请求行以一个方法符号开头,以空格分开,后面跟着请求的URI和协议的版本,格式如下:

Method Request-URI HTTP-Version CRLF

方法/统一资源定位符/协议版本/终止符

• ES 常用

– PUT--- 增

– DELETE --- 删

– POST --- 改

– GET --- 查

系统命令 curl

• 在linux中curl是一个利用URL规则在命令行下工作的文件传输工具,可以说是一款非常强大的http命令行工具。它支持多种请求模式,自定义请求 头等强大功能,是一款综合工具

• curl 常用参数介绍

– -A 修改请求 agent

– -X 设置请求方法

– -i 显示返回头信息+内容

ES插件的使用

• ES 常用插件

• head 插件:

– 它展现ES集群的拓扑结构,幵且可以通过它来进行索引(Index)和节点(Node)级别的操作

– 它提供一组针对集群的查询API,幵将结果以json和表格形式返回

– 它提供一些快捷菜单,用以展现集群的各种状态

• kopf 插件

– 是一个ElasticSearch的管理工具

– 它提供了对ES集群操作的API

• bigdesk 插件

– 是elasticsearch的一个集群监控工具

– 可以通过它来查看es集群的各种状态,如:cpu、内存使用情况,索引数据、搜索情况,http连接数等

ES 插件安装、查看

– 查看安装的插件

/usr/share/elasticsearch/bin/plugin list

– 安装插件

/usr/share/elasticsearch/bin/plugin install ftp://192.168.4.254/head.zip

/usr/share/elasticsearch/bin/plugin install file:///tmp/kopf.zip

– 这里必须使用 url 的方式进行安装,如果文件在本地,我们也需要使用 file:// 的方式指定路径,例如文件在/tmp/xxx 下面,我们要写成 file:///tmp/xxx,网络源写ftp:// http://,删除使用 remove 指令

访问插件

<http://192.168.1.11:9200/_plugin/head/>

<http://192.168.1.11:9200/_plugin/kopf/>

<http://192.168.1.11:9200/_plugin/bigdesk>

有星星的是master，其他是slave

head 插件

新建索引

分片数：5 将数据平均分配存到5台机器上，数字不能超过集群总数

副本数：1 源数据一份，再copy1份

RESTful API 调用

– v 参数显示详细信息

http://192.168.4.15:9200/\_cat/health?v

– help 显示帮助信息

http://192.168.1.11:9200/\_cat/health?help

RESTful API 调用

• RESTful API 增加

– 创建一个索引,并设置分片数量与副本数量

-d 向服务器分向数据

curl -XPUT 'http://192.168.1.11:9200/要创建的索引名/' -d '{

"settings":{

"index":{

"number\_of\_shards": 5,

"number\_of\_replicas": 1

}

}

}'

增加 PUT

curl -XPUT 'http://192.168.1.11:9200/nsd1803/teacher/1' -d '{

"title": "阶段1",

"name":{"first": "小逗比", "last": "牛犇"},

"age": 25

}'

修改 POST doc://行

curl -XPOST 'http://192.168.1.11:9200/nsd1803/teacher/1/\_update' -d '{

"doc":{

"age": 18

}

}'

查询 GET

curl -XGET 'http://192.168.1.11:9200/nsd1803/teacher/1'?pretty

删除 DELETE

curl -XDELETE 'http://192.168.1.11:9200/nsd1803/teacher/1' //删除数据

curl -XDELETE 'http://192.168.1.11:9200/nsd1803 //删除索引

curl -XDELETE 'http://192.168.1.11:9200/\* //删除根

kibana 安装

kibana是什么

– 数据可视化平台工具

• 特点:

– 灵活的分析和可视化平台

– 实时总结和流数据的图表

– 为丌同的用户显示直观的界面

– 即时分享和嵌入的仪表板

• kibana安装

– kibana 的安装非常简单,我们使用 rpm 方式安装

yum -y install kibana

– kibana 默认安装在 /opt/kibana 下面,配置文件在

/opt/kibana/config/kibana.yml

– 我们只需要修改少量的配置就可以启动

• kibana.yml 的配置

– server.port: 5601 //端口最好不要改

– server.host: "0.0.0.0"

– elasticsearch.url: "http://192.168.4.13:9200" //任意一台es服务器地址

– kibana.index: ".kibana" //kibana在es中创建索引的名称

– kibana.defaultAppId: "discover" //默认首页

– elasticsearch.pingTimeout: 1500

– elasticsearch.requestTimeout: 30000

– elasticsearch.startupTimeout: 5000

• kibana.yml 的配置

– 除 elasticsearch.url 需要配置为我们 ES 集群的地址之外,其他保持默认值就可以了

– 设置开机启动

systemctl enable kibana

– 启动服务

systemctl start kibana

– web 访问 kibana status 绿色 表明kibana和es整合成功

http://192.168.1.16:5601/

数据批量导入 \_bulk

• 使用 \_bulk 批量导入数据

– 批量导入数据使用 POST 方式,数据格式为 json,url编码使用 data-binary

– 导入含有 index 配置的 json 文件

gzip –d logs.jsonl.gz

curl -XPOST 'http://192.168.4.14:9200/\_bulk' --data-binary

@logs.jsonl

gzip –d shakespeare.json.gz

curl -XPOST 'http://192.168.4.14:9200/\_bulk' --data-binary

@shakespeare.json

如果json文件中未指定索引（库）和类型（表），可以在URL中指定

http://192.168.4.14:9200/oo(索引)/xx(类型)/\_bulk' --data-binary

@accounts.json

数据批量查询 \_mget

curl -XGET 'http://192.168.4.11:9200/\_mget?pretty' -d '{

"docs":[

{

"\_index": "accounts",

"\_type:": "act",

"\_id": 1

},

{

"\_index": "accounts",

"\_type:": "act",

"\_id": 2

}

]

查询 索引/类型/id 定具体数据

map 映射

logstash 部分

• logstash 是什么

– logstash是一个数据采集、加工处理以及传输的工具

– Logstash 没有默认的配置文件,需要手动配置

配置/etc/hosts es1-5

• logstash 工作结构

{ 数据源 } ==>

1.input { } ==> 收集日志文件，读数据

2.filter { } ==> 负责对日志进行加工管理，映射关系

3.output { } ==> 将数据写入ES集群中

{ ES }

logstash收集日志，将数据写入ES中

ES集群

kibana通过ES读取数据进行展示

• logstash 里面的类型

– 布尔值类型: ssl\_enable => true

– 字节类型:bytes => "1MiB"

– 字符串类型: name => "xkops"

– 数值类型: port => 22

– 数组: match => ["datetime","UNIX"]

– 哈希: options => {k => "v",k2 => "v2"}

– 编码解码: codec => "json"

– 路径: file\_path => "/tmp/filename"

– 注释: #

• logstash 条件判断

– 等于: ==

– 不等于: !=

– 小于: <

– 大于: >

– 小于等于: <=

– 大于等于: >=

– 匹配正则: =~

– 不匹配正则: !~

• logstash 条件判断

– 包含: in

– 不包含: not in

– 不: and

– 或: or

– 非不: nand

– 非或: xor

– 复合表达式: ()

– 取反符合:!()

• logstash 的第一个配置文件

vim /etc/logstash/logstash.conf

input{

stdin{ codec =>"json" }

}

filter{ }

output{

stdout{codec => "rubydebug" } 调试输出

}

• 启动并验证

cd /etc/logstash

/opt/logstash/bin/logstash -f logstash.conf

• logstash 插件

上面的配置文件使用了 logstash-input-stdin 和logstash-output-stdout 两个插件,logstash 还有filter 和 codec 类插件,查看插件的方式是/opt/logstash/bin/logstash-plugin list

logstash-codec-xxx 编码，用于所有区域段

logstash-input-xxx

logstash-filter-xxx

logstash-output-xxx

网上插件地址：<https://github.com/logstash-plugins>

https://www.elastic.co/guide/en/logstash/current/index.html

INPUT

file插件

File Input Configuration Options

| Setting | Input type | Required |
| --- | --- | --- |
| [close\_older](https://www.elastic.co/guide/en/logstash/current/plugins-inputs-file.html" \l "plugins-inputs-file-close_older" \o "close_older)path | Array | Yes |
| sincedb\_path | string | No |
| [start\_positionstart\_position](https://www.elastic.co/guide/en/logstash/current/plugins-inputs-file.html" \l "plugins-inputs-file-start_position" \o "start_position)\_start | string,one of [beginning,end] | No |

key value类型 是否必须

input{

file{

sincedb\_path=>"/var/lib/logstash/sincedb.log" //自定义文件位置

path=>["/tmp/a.log"，"/tmp/b.log"]

type=>"filelog" //自定义类型，标记该输出是从哪种类型输出的

}

}

position\_start默认是end,从最后读取，如果为beginning，如果该文件不存在会从头读取,然后再重新创建sincedb.log文件

关闭logstash服务再重启之后，会从上一次的日志之后开始接着读取

服务重启之后会首先在/root家目录中.sincedb\_abxxxxx文件会记录之前读取过的文件。一般会申明一个公共的位置存放该文件

tcp插件

input{

tcp:{

mode=>"server"

port=>8888

host=>"0.0.0.0" //表示监听所有地址

type=>"tcplog"

}

}

mode：client | server

默认是server，port是监听端口，如果client，port是连接远程服务器的端口

host：如果server，就是本机的IP地址，如果是client,就是远程服务器的IP地址

udp插件

udp只有服务端，因为是无连接协议，只需要host和port

input{

tcp:{

port=>9999

type=>"udplog"

}

}

测试： echo testtcp > /dev/tcp/192.168.1.17/8888

echo testudp > /dev/udp/192.168.1.17/9999

syslog插件

syslog{

host => "0.0.0.0"

port => 514 //默认端口

type => "syslog"

}

自定义系统日志

vim /etc/rsyslog.conf

local0.info(等级) /var/log/info.log //写入本机日志

local0.info(等级) @@192.168.1.17:514 //往远程主机写入日志，＠＠使用TCP协议写，＠使用UDP协议写

systemctl restart rsyslog

写日志：logger -p local0.info -t "testlog日志主题" "日志内容"

apache日志格式：/etc/httpd/conf/httpd.conf

FILTER

grok插件

使用正则表达式把非结构化的数据结构化

使用正则宏匹配，正则宏路径

匹配要存入数据库中的字段

?<name>reg

filter{

grok{

match=>["message",%{IP:client\_ip}]

}

}

正则宏路径

/opt/logstash/vendor/bundle/jruby/1.9/gems/logstash-patterns-core-2.0.5/patterns/grok-patterns

filter{

if [type] =="apachelog"{ //判断自定义日志类型，好判断

grok{

match => ["message", "%{COMBINEDAPACHELOG}"]

//apache日志

}

}

}

OUTPUT

output ES 插件

output{

if [type] == "apachelog"{

elasticsearch{

hosts=>["192.168.1.15:9200","192.168.1.11:9200"]

//防止宕机，多写几台

index=>"apachelog" //索引名

flush\_size=>2000 //超过2000字节写一次，不要过于频繁

idle\_flush\_time=>10 //每10秒写一次

}

}

if [type] =="xxx"{

............

}

}

• input filebeats 插件

input{

stdin{}

beats{

port=>5044

}

}

yum -y install filebeat //在要收集日志的主机上安装，会推送到logstash的filter处理

vim /etc/filebeat/filebeat.yml

#elasticsearch hosts注释掉

paths: - /etc/httpd/logs/error.log

- /etc/httpd/logs/access\_log.log

document\_type:apachelog

logstash

hosts:["192.168.1.17:5044"] //logstash地址

systemctl restart filebeat

------------------------------------------------------------------------------

input{

stdin{codec=>"json"}

file{

path=>["/tmp/a.log","/tmp/b.log"]

sincedb\_path=>"/var/lib/logstash/sincedb.log"

start\_position=>"beginning"

type=>"log"

}

tcp{

port=>8888

host=>"0.0.0.0"

type=>"tcplog"

}

udp{

port=>9999

type=>"udplog"

}

syslog{

host=>"0.0.0.0"

port=>514

type=>"syslog"

}

beats{

port=>5044

}

}

filter{

if [type]=="apachelog"{

grok{

match => ["message", "%{COMBINEDAPACHELOG}"]

}

}

}

output{

stdout{codec=>"rubydebug"}

if [type]=="apachelog"{

elasticsearch{

hosts=>["192.168.1.15:9200","192.168.1.11:9200"]

index=>"apachelog"

flush\_size=>2000

idle\_flush\_time=>10

}

}

}

------------------------------------------------------------------------------

金步国作品集 翻译官方技术文档

kibana 装apache

stdin 0 标准输入（写）

stdout 1 标准输出（读）

stderr 2 标准错误(写)

cat > f1 << EOF

.....

EOF 终止符,将EOF中的内容写入到f1文件中

exec 修改文件数字描述符 执行完命令退出(exec ls)

对文件描述符的操作就是对文件的操作，文件描述符相当于一个接口

ls /dev/fd

cat f1>6&

exec 6>&- 关闭6描述符

touch ff exec 1>ff 将1的标准输出写入到ff文件中

tail -f ff

exec 9<>/dev/tcp/www.baidu.com/80

echo -ne “GET /index.html HTTP/1.1\r\n” >&9

echo -ne “Host:www.baidu.com\r\n\r\n”