

# 大型架构及配置技术

**NSD ARCHITECTURE**

**DAY03**

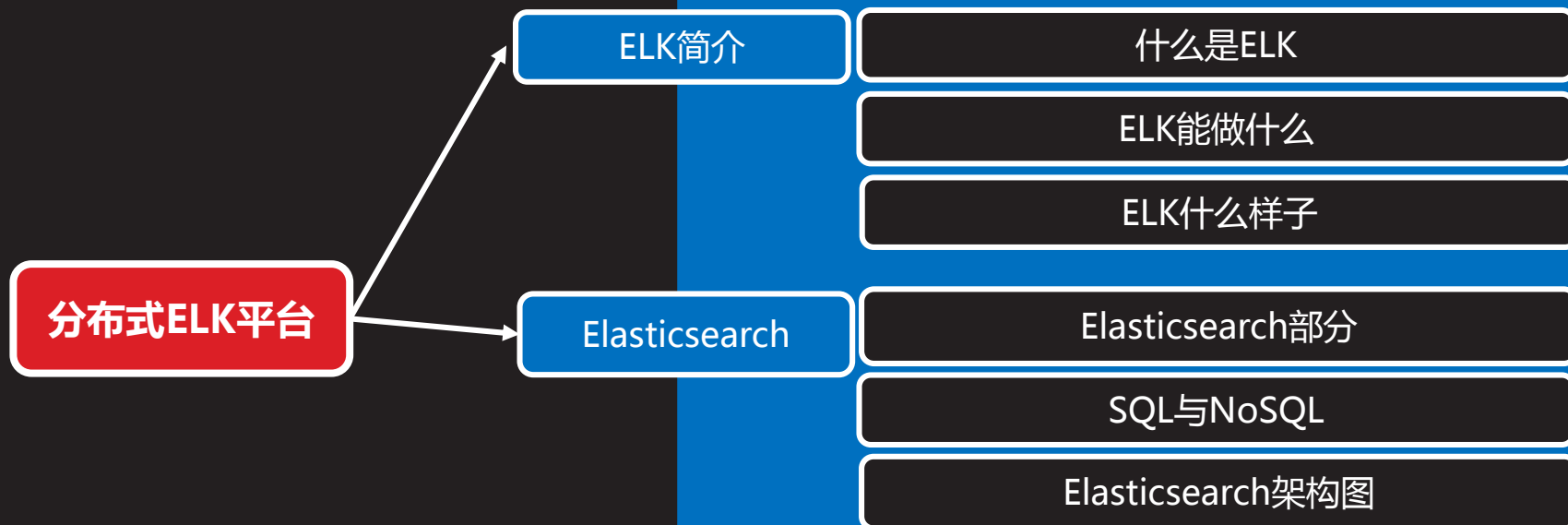
# 内容

上午	09:00 ~ 09:30	作业讲解和回顾
	09:30 ~ 10:20	分布式ELK平台
	10:30 ~ 11:20	ES集群安装
	11:30 ~ 12:00	
下午	14:00 ~ 14:50	扩展插件
	15:00 ~ 15:50	
	16:10 ~ 17:10	Kibana安装
	17:20 ~ 18:00	总结和答疑



# 分布式ELK平台

---



# ELK简介



# 什么是ELK

- ELK是一整套解决方案，是三个软件产品的首字母缩写，很多公司都在使用，如：Sina、携程、华为、美团等
- ELK分别代表
  - Elasticsearch：负责日志检索和储存
  - Logstash：负责日志的收集和分析、处理
  - Kibana：负责日志的可视化
- 这三款软件都是开源软件，通常是配合使用，而且又先后归于Elastic.co公司名下，故被简称为ELK

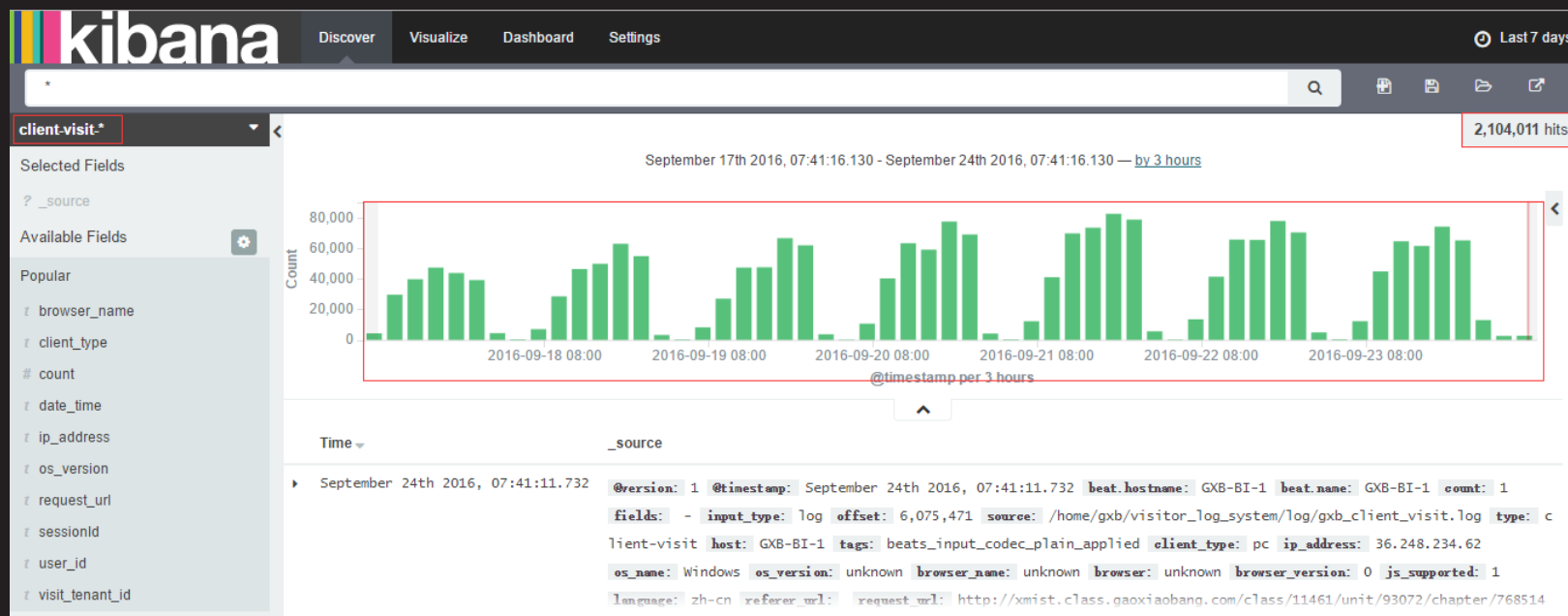


# ELK能做什么

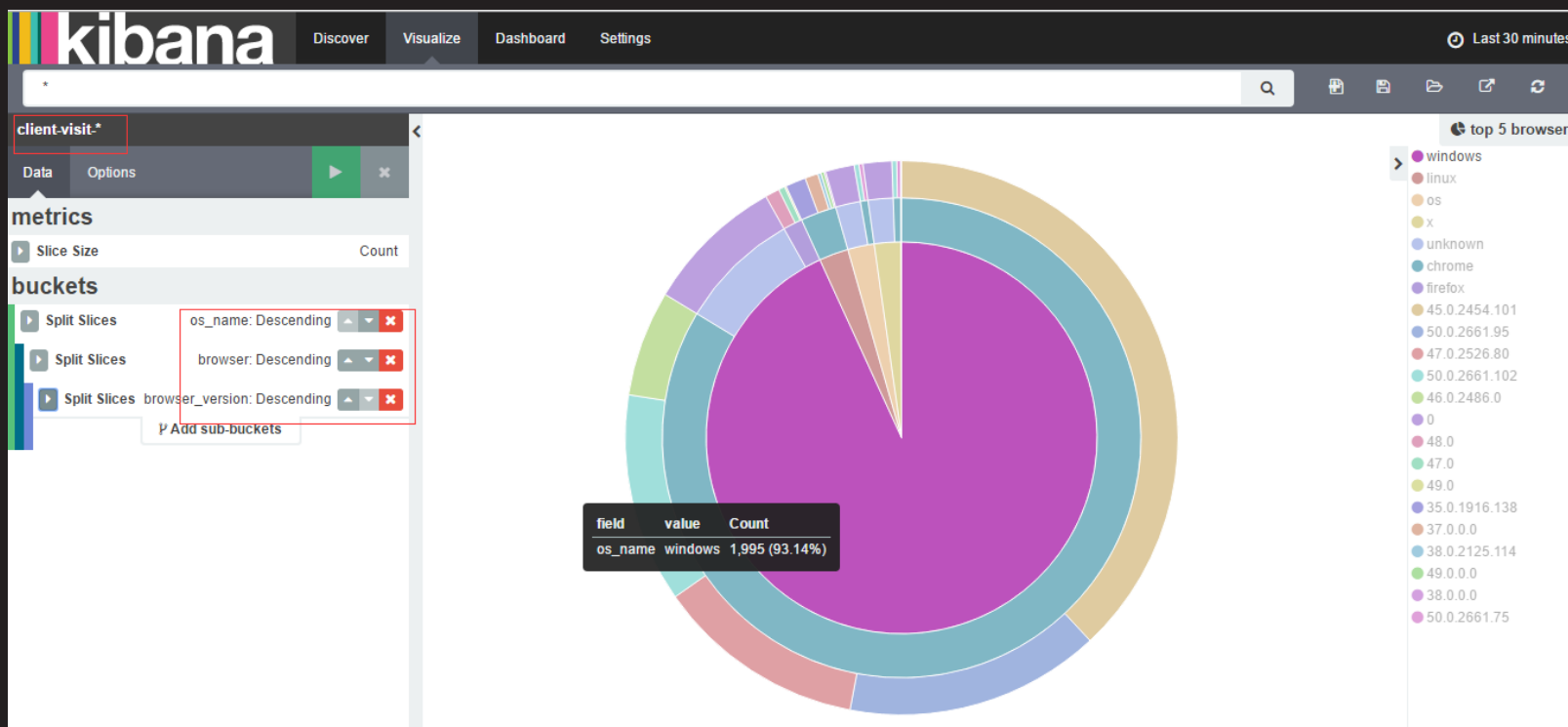
- ELK组件在海量日志系统的运维中，可用于解决
  - 分布式日志数据集中式查询和管理
  - 系统监控，包含系统硬件和应用各个组件的监控
  - 故障排查
  - 安全信息和事件管理
  - 报表功能



# ELK 什么样子

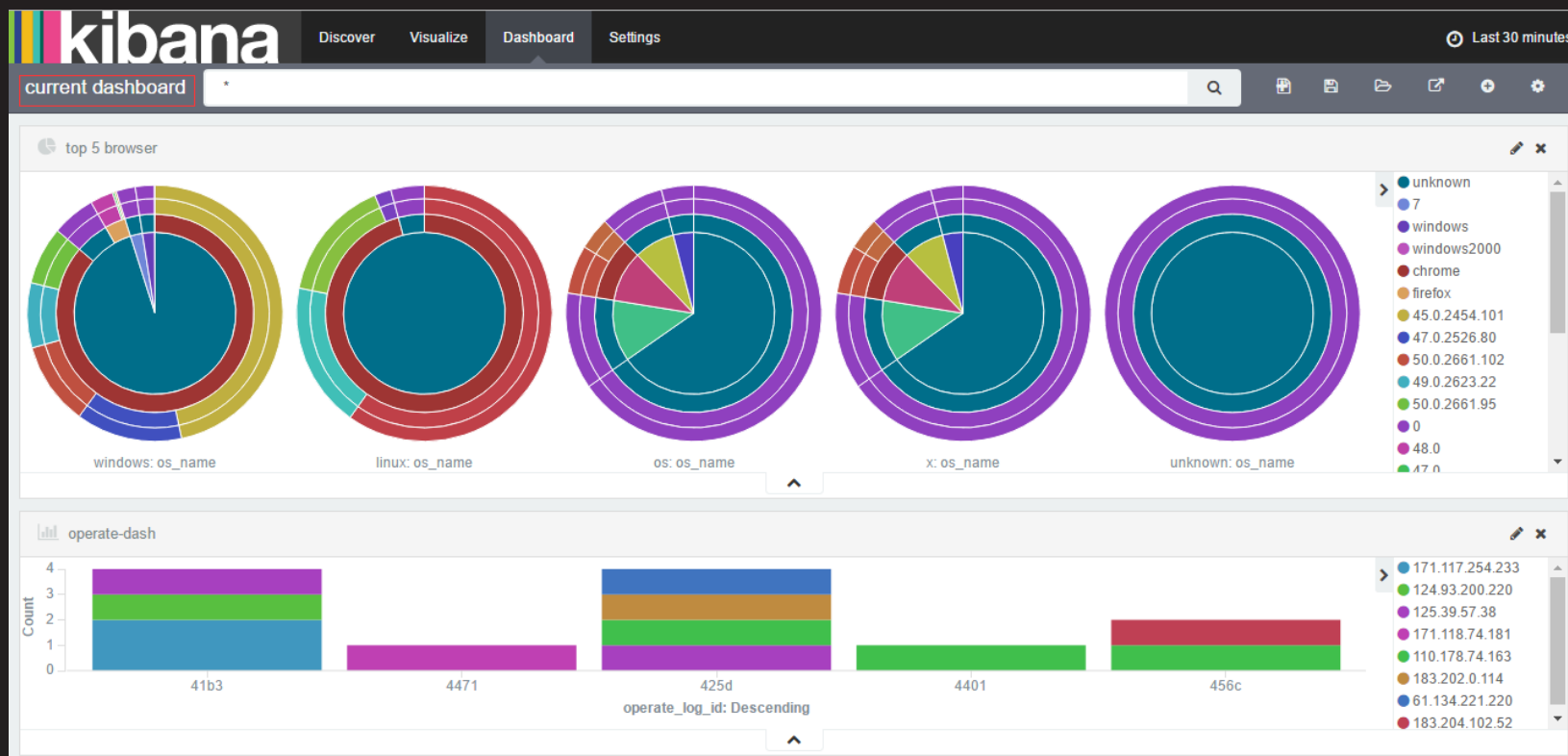


# ELK什么样子（续1）





# ELK 什么样子 (续2)



# Elasticsearch

---

# 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，并作为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



# Elasticsearch部分（续1）

- 主要特点
  - 实时分析
  - 分布式实时文件存储，并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



# Elasticsearch部分（续2）

- 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



# Elasticsearch部分（续3）

- 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：一个可被搜索的基础信息单元
  - Index：拥有相似特征的文档的集合
  - Type：一个索引中可以定义一种或多种类型
  - Field：是ES的最小单位，相当于数据的某一行
  - Shards：索引的分片，每一个分片就是一个Shard
  - Replicas：索引的拷贝



# SQL与NoSQL

## • ES与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 ）



# SQL与NoSQL ( 续1 )

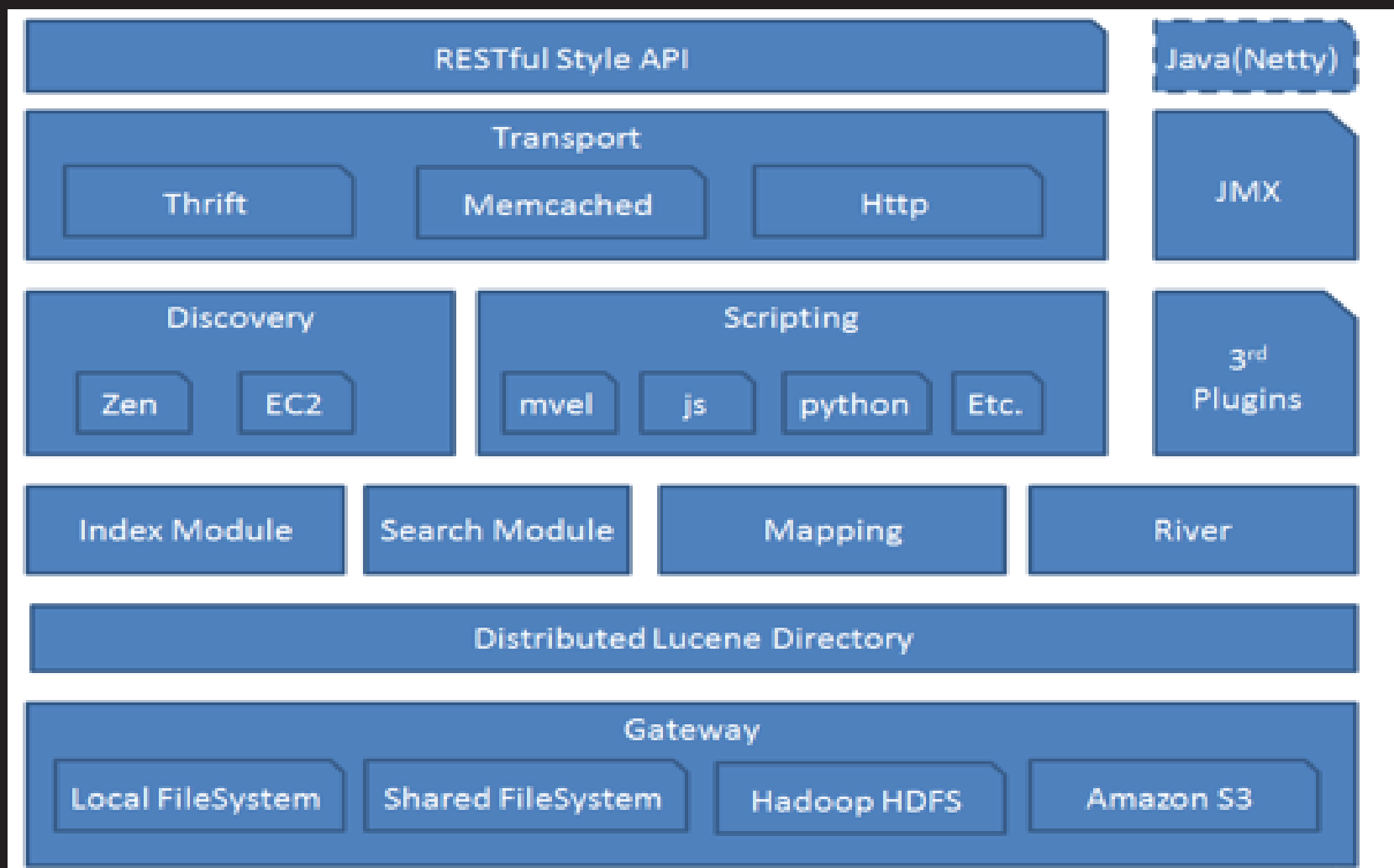
- ES与关系型数据库的对比

<i>Relational database</i>	<i>Elasticsearch</i>
Database	Index
Table	Type
Row	Document
Column	Field
Schema	Mapping
Index	Everything is indexed
SQL	Query DSL
SELECT * FROM table...	GET http://...
UPDATE table SET	PUT http://...

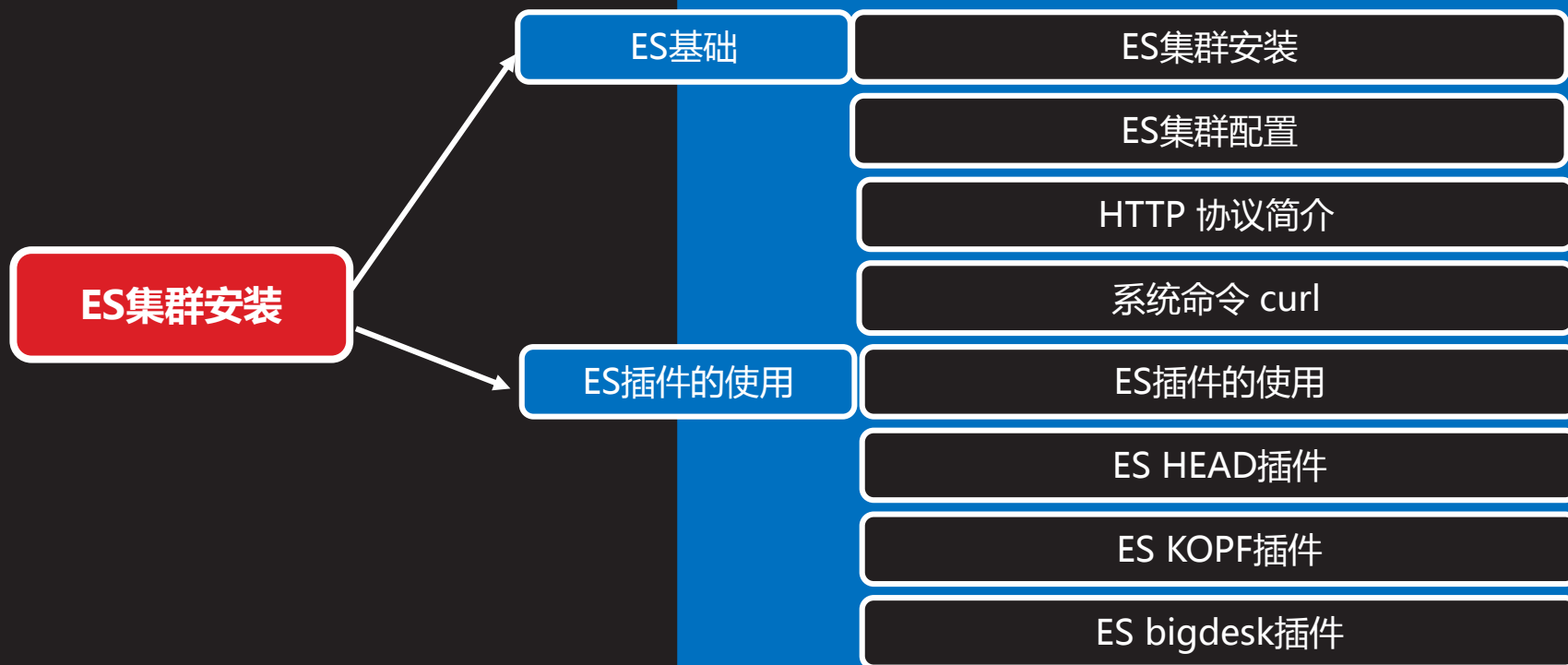




# Elasticsearch架构图



# ES集群安装



# ES基础

---

# ES集群安装

- 安装第一台ES服务器
  - 设置主机名称和ip对应关系
  - 解决依赖关系
  - 安装软件包
  - 修改配置文件
  - 启动服务
  - 检查服务



# ES集群安装（续1）

- 设置ip与主机名称对应关系
  - 配置/etc/hosts  
192.168.1.51 es1
- 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



# ES集群安装（续2）

- 安装ES

```
yum -y install elasticsearch
```

- 修改配置文件

- elasticsearch.yml

```
network.host: 0.0.0.0
```



# ES集群安装（续3）

- 启动服务
  - 启动服务并设开机自启

```
systemctl enable elasticsearch
```

```
systemctl start elasticsearch
```
  - 验证：

```
netstat -ltunp
```
  - 能够看到9200，9300被监听



# ES集群安装（续4）

- 通过浏览器或curl访问9200端口

```
curl http://192.168.1.51:9200/
{
  "name" : "es1",
  "cluster_name" : "myelk",
  "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"
}
```





# 案例1：ES集群安装

1. 准备1台虚拟机
2. 部署elasticsearch第一个节点
3. 访问9200端口查看是否安装成功



# ES集群配置

- ES集群配置
  - ES集群配置也很简单，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: myelk  
node.name: es1  
network.host: 0.0.0.0  
discovery.zen.ping.unicast.hosts: [ "es1", "es2", "es3"]
```



# ES集群配置（续1）

- ES集群配置
  - 集群中的所有节点要相互能够ping通，要在所有集群机器上配置/etc/hosts中的主机名与ip对应关系
  - 集群中所有机器都要安装Java环境
  - cluster.name集群名称配置要求完全一致
  - node.name为当前节点标识，应配置本机的主机名
  - discovery为集群节点机器，不需要全部配置
  - 配置完成以后启动所有节点服务



# ES集群配置（续2）

- ES集群配置

- 验证集群，使用ES内置字段 `_cluster/health`

```
curl http://192.168.1.51:9200/_cluster/health?pretty
```

```
{
  "cluster_name" : "myelk",
  "status" : "green",
  "timed_out" : false,
  "number_of_nodes" : 5,
  "number_of_data_nodes" : 5,
  ...
  ...
  "active_shards_percent_as_number" : 100.0
}
```



# ES集群配置（续3）

- ES 集群验证
  - 返回字段解析
  - status " : " green " 集群状态，绿色为正常，黄色表示有问题但不是很严重，红色表示严重故障
  - "number\_of\_nodes" : 5, 表示集群中节点的数量
    - "number\_of\_data\_nodes" : 5,
    - .....
    - "task\_max\_waiting\_in\_queue\_millis" : 0,
    - "active\_shards\_percent\_as\_number" : 100.0



## 案例2：ES集群安装配置

1. 一共安装5台虚拟机
2. 在所有机器中部署ES
3. 启动服务查看验证集群状态



# ES插件的使用



# 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，并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，并将结果以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





# ES插件的使用（续1）

- ES常用插件
- 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插件的使用（续2）

- 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://192.168.1.254/head.zip
```

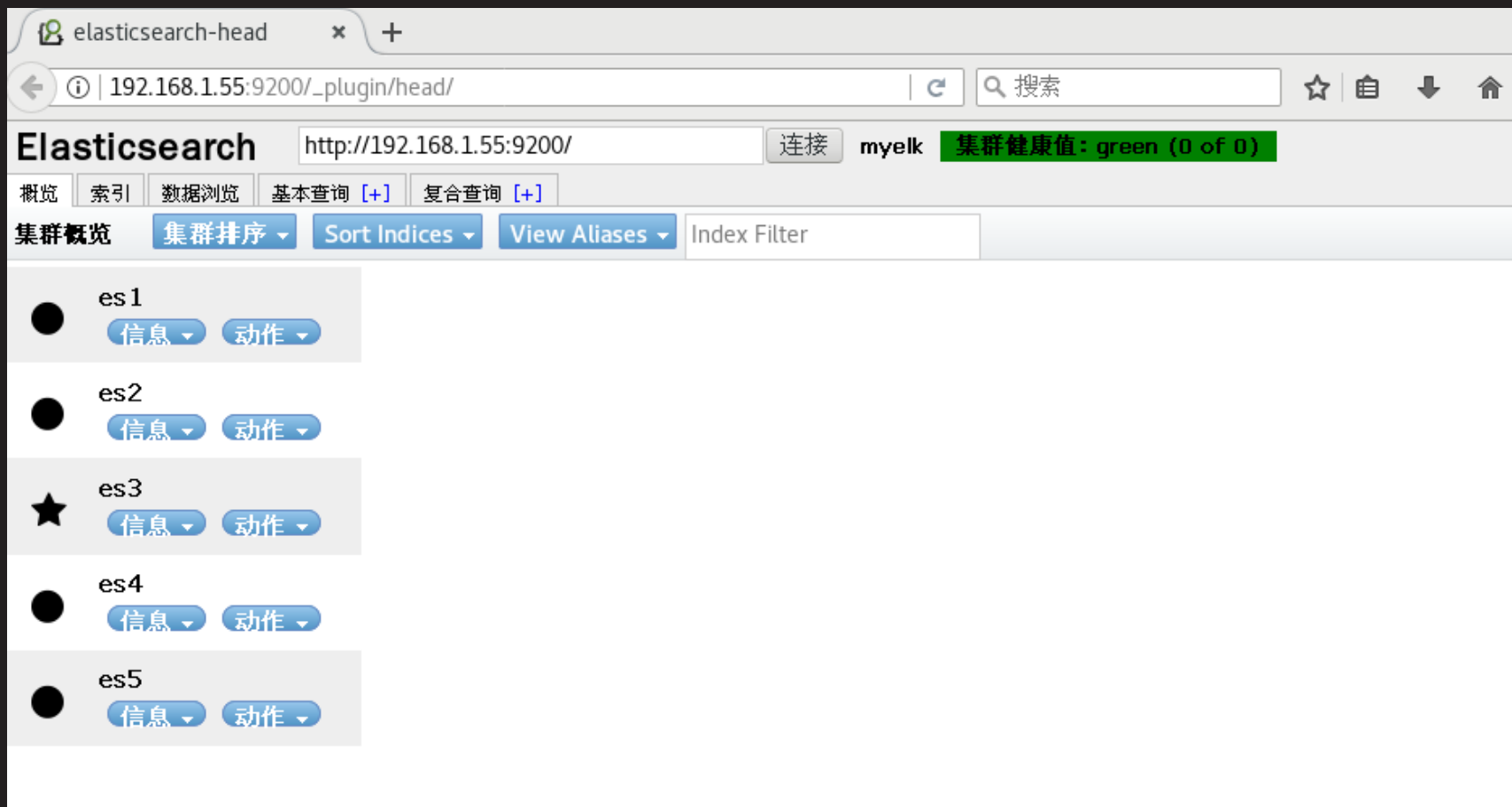
```
/usr/share/elasticsearch/bin/plugin install
```

```
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，删除使用 remove 指令



# ES HEAD插件



# ES KOPF插件

The screenshot shows the ES KOPF web interface in a browser. The address bar displays the URL `192.168.1.55:9200/_plugin/kopf/#!/cluster`. The interface includes a navigation bar with tabs for `cluster`, `nodes`, `rest`, and `more`. Below the navigation bar, there is a search bar and a table of nodes.

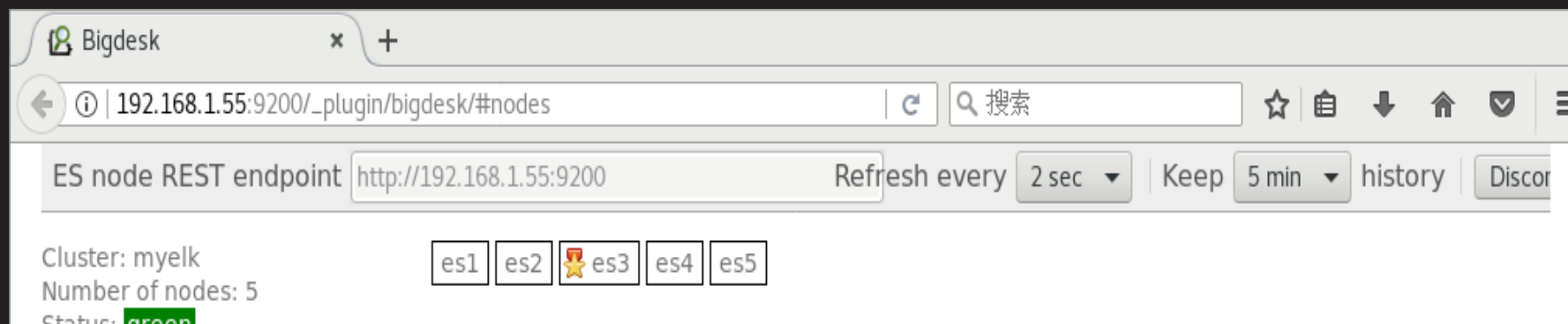
The table displays the following data:

Node Name	IP Address	heap	disk	cpu	load
es1	192.168.1.51				
es2	192.168.1.52				
es3	192.168.1.53				
es4	192.168.1.54				
es5	192.168.1.55				

At the bottom of the interface, there is a `show log` button.



# ES bigdesk 插件



## 案例4：练习插件

1. 在其中一台机器上部署插件
2. 使用bigdesk查看集群状态
3. 使用head创建index
4. 使用kopf查看数据



# 扩展插件

---



# RESTful API





# HTTP 协议简介

- 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和协议版本，格式如下：  
Method Request-URI HTTP-Version CRLF



# HTTP 协议简介（续1）

- http请求方法
  - 常用方法 GET , POST , HEAD
  - 其他方法 OPTIONS , PUT , DELETE , TRACE和 CONNECT
- 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，可以说是一款很强大的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



# RESTful API 调用

- Elasticsearch提供了一系列RESTful的API
  - 检查集群、节点、索引的健康度、状态和统计
  - 管理集群、节点、索引的数据及元数据
  - 对索引进行CRUD操作及查询操作
  - 执行其他高级操作如分页、排序、过滤等
- POST或PUT数据使用json格式



# RESTful API 调用 ( 续1 )

- JSON
  - JSON ( JavaScript Object Notation ) , 意思是 JavaScript 对象表示法 , 它是一种基于文本独立于语言的轻量级数据交换格式。
  - JSON 格式
  - 数组 [ "aa", "bb", "cc" ]
  - 键值对 { key: value }



# RESTful API 调用（续2）

- RESTful API的简单使用
  - \_cat API查询集群状态，节点信息
  - v参数显示详细信息  
[http://192.168.1.51:9200/\\_cat/health?v](http://192.168.1.51:9200/_cat/health?v)
  - help显示帮助信息  
[http://192.168.1.51:9200/\\_cat/health?help](http://192.168.1.51:9200/_cat/health?help)



# RESTful API 调用 ( 续3 )

- Rest API 的简单使用
  - nodes 查询节点状态信息  
[http://192.168.1.51:9200/\\_cat/nodes?v](http://192.168.1.51:9200/_cat/nodes?v)
  - 索引信息  
[http://192.168.1.51:9200/\\_cat/indices?v](http://192.168.1.51:9200/_cat/indices?v)



# RESTful API 调用 ( 续4 )

- RESTful API增加
  - 创建一个索引，并设置分片数量与副本数量

```
curl -XPUT 'http://192.168.1.55:9200/tedu/' -d '{
  "settings":{
    "index":{
      "number_of_shards": 5,
      "number_of_replicas": 1
    }
  }
}'
```





# RESTful API 调用 ( 续5 )

- RESTful API插入数据

- ( 增 ) 加数据 , 使用 PUT 方法

- 调用方式 : 数据库地址/索引/类型/id值

```
curl -XPUT 'http://192.168.1.55:9200/tedu/teacher/1' -d '{  
  "职业": "诗人",  
  "名字": "李白",  
  "称号": "诗仙",  
  "年代": "唐"  
}'
```



# RESTful API 调用 ( 续6 )

- POST修改
    - 修 ( 改 ) 数据 , 使用 POST 方法
    - 在修改数据的时候必须调用 \_update 关键字
    - 调用方式 : 数据库地址/索引/类型/id值/\_update
- ```
curl -XPOST
'http://192.168.1.55:9200/tedu/teacher/3/_update' -d '{
    "doc":{
        "年代": "唐代"
    }
}'
```



# RESTful API 调用 ( 续6 )

- ( 查 ) 查询
  - 查询使用 GET 方法 , GET为默认方法
  - 查询显示结果时候可以用 pretty 规范显示格式
  - 多条查询需要使用 \_mget 关键字配合 json 调用

```
curl -XGET 'http://192.168.1.55:9200/tedu/teacher/1'
```
- ( 删 ) 除
  - 删除时候可以是文档 , 也可以是库 , 但不能是类型

```
curl -XDELETE 'http://192.168.1.55:9200/tedu/teacher/1'
```

```
curl -XDELETE 'http://192.168.1.55:9200/tedu'
```



## 案例5：插入，增加，删除查询数据

1. 使用curl命令连接使用ES数据库
2. 使用PUT方法增加数据
3. 使用POST修改数据
4. 使用GET查询数据
5. 使用DELETE删除数据



# 批量导入数据

- 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://192.168.1.51:9200/_bulk' --data-binary @logs.jsonl
```



# map 映射

- mapping :
  - 映射：创建索引的时候，可以预先定义字段的类型及相关属性
  - 作用：这样会让索引建立得更加的细致和完善
  - 分类：静态映射和动态映射
  - 动态映射：自动根据数据进行相应的映射
  - 静态映射：自定义字段映射数据类型



# 案例6：批量导入数据

## 1. 使用 curl 命名为集群批量导入数据



# 总结和答疑

---