

# 大型架构

**NSD ELK**

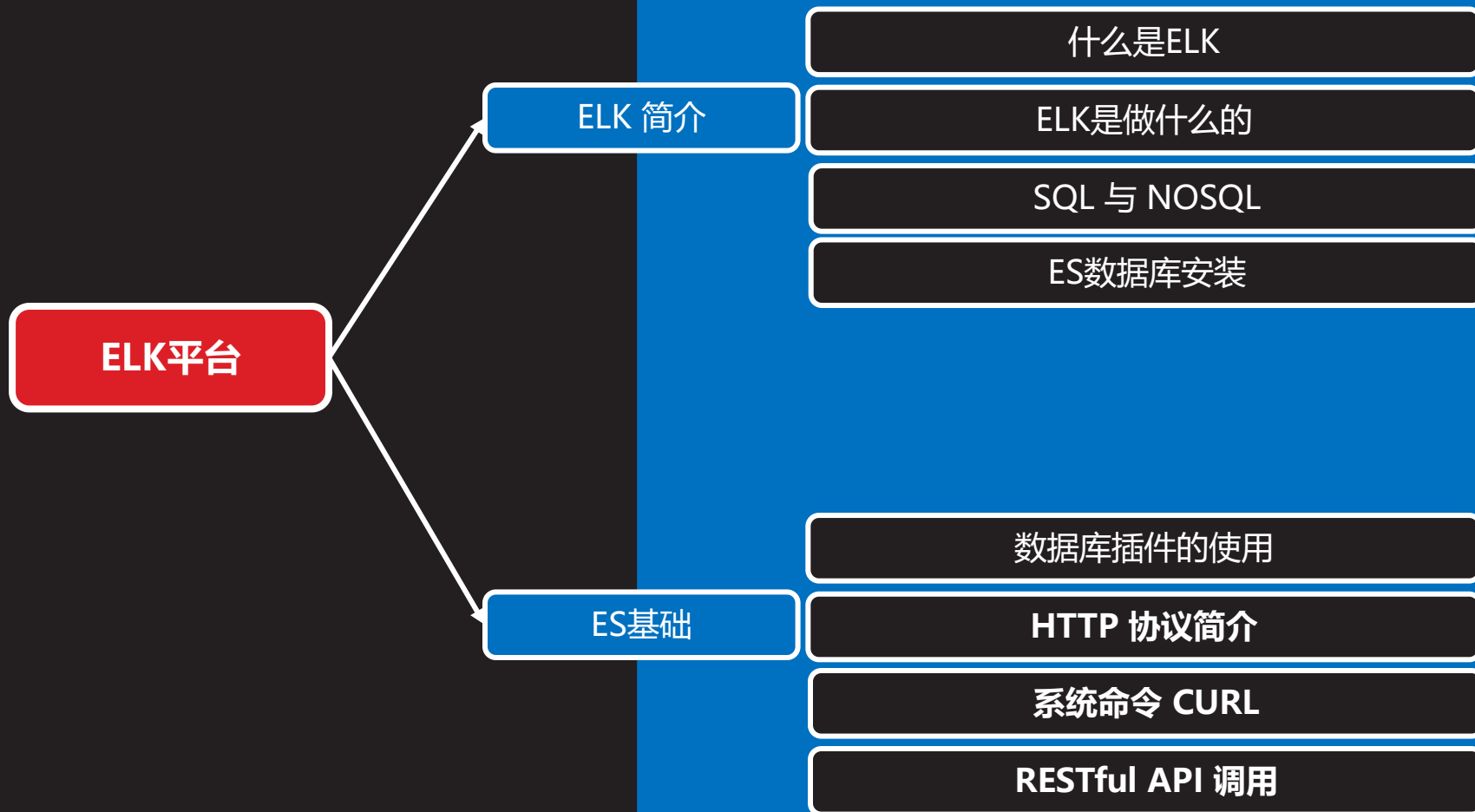
**DAY01**

# 内容

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



# 分布式ELK平台



# 基础知识

---

# ELK 是什么？

- Sina、饿了么、携程、华为、美团、freewheel、畅捷通、新浪微博、大讲台、魅族、IBM..... 这些公司都在使用 ELK！ELK！ELK！
- ELK竟然重复了三遍，是个什么鬼？



# ELK 是什么？

- 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
  - 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

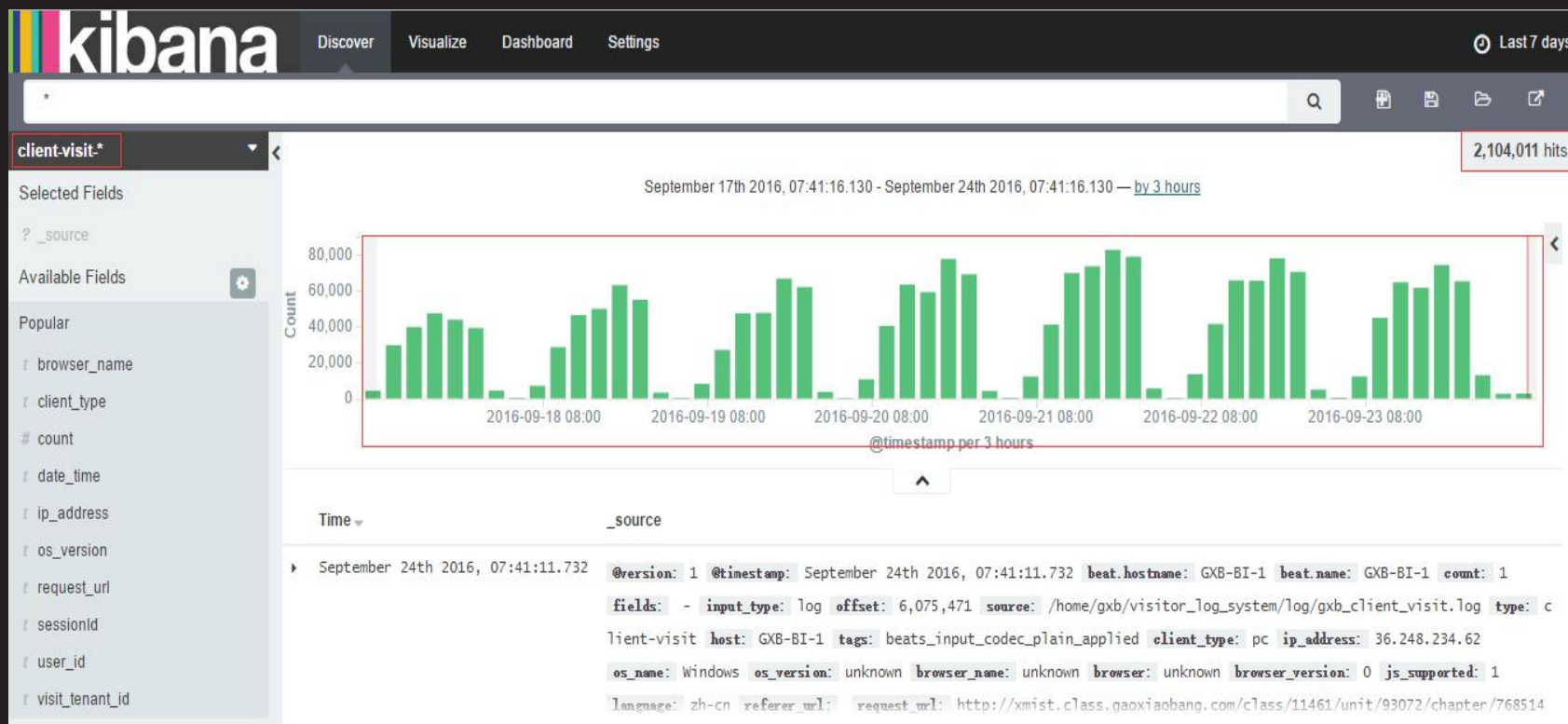


# ELK 能做什么？

- 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



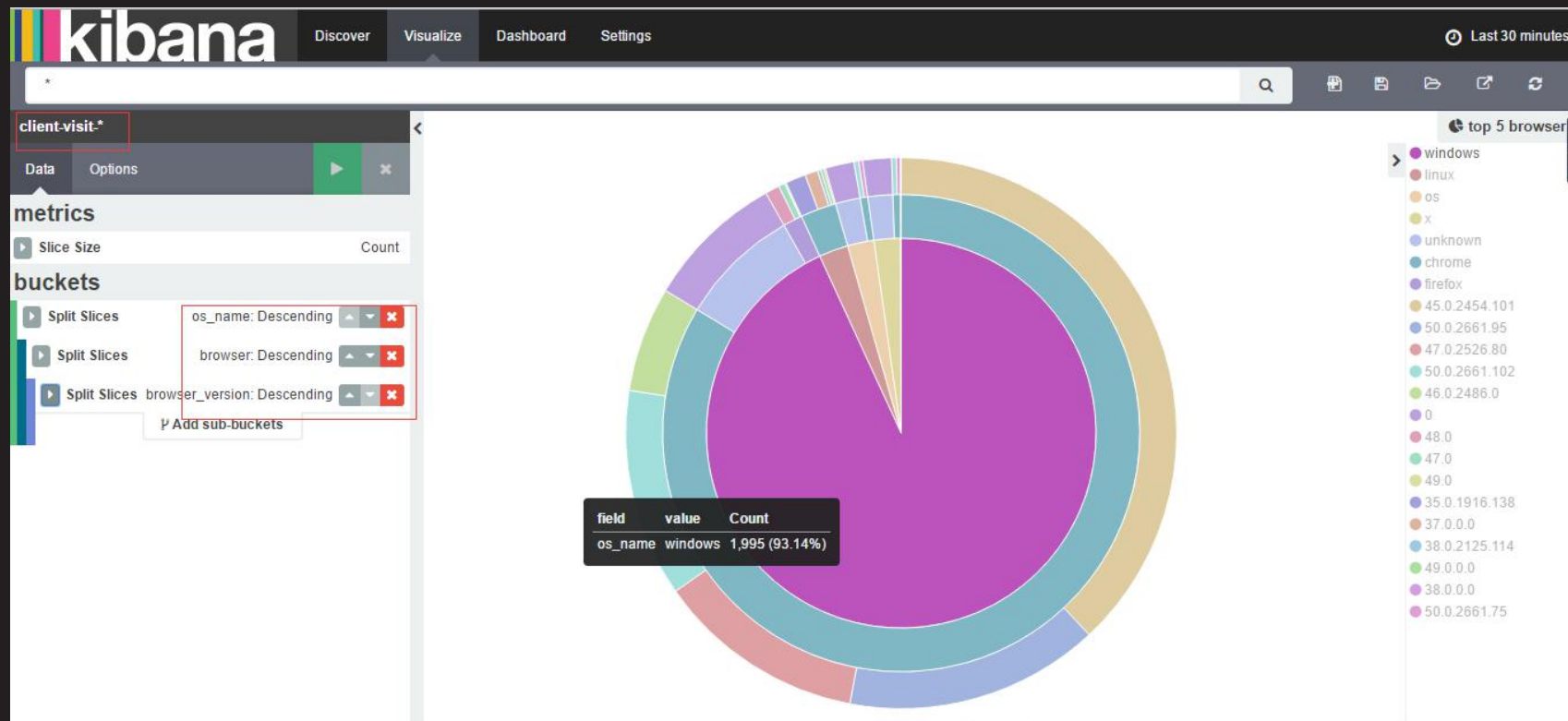
# ELK 是什么样子的？





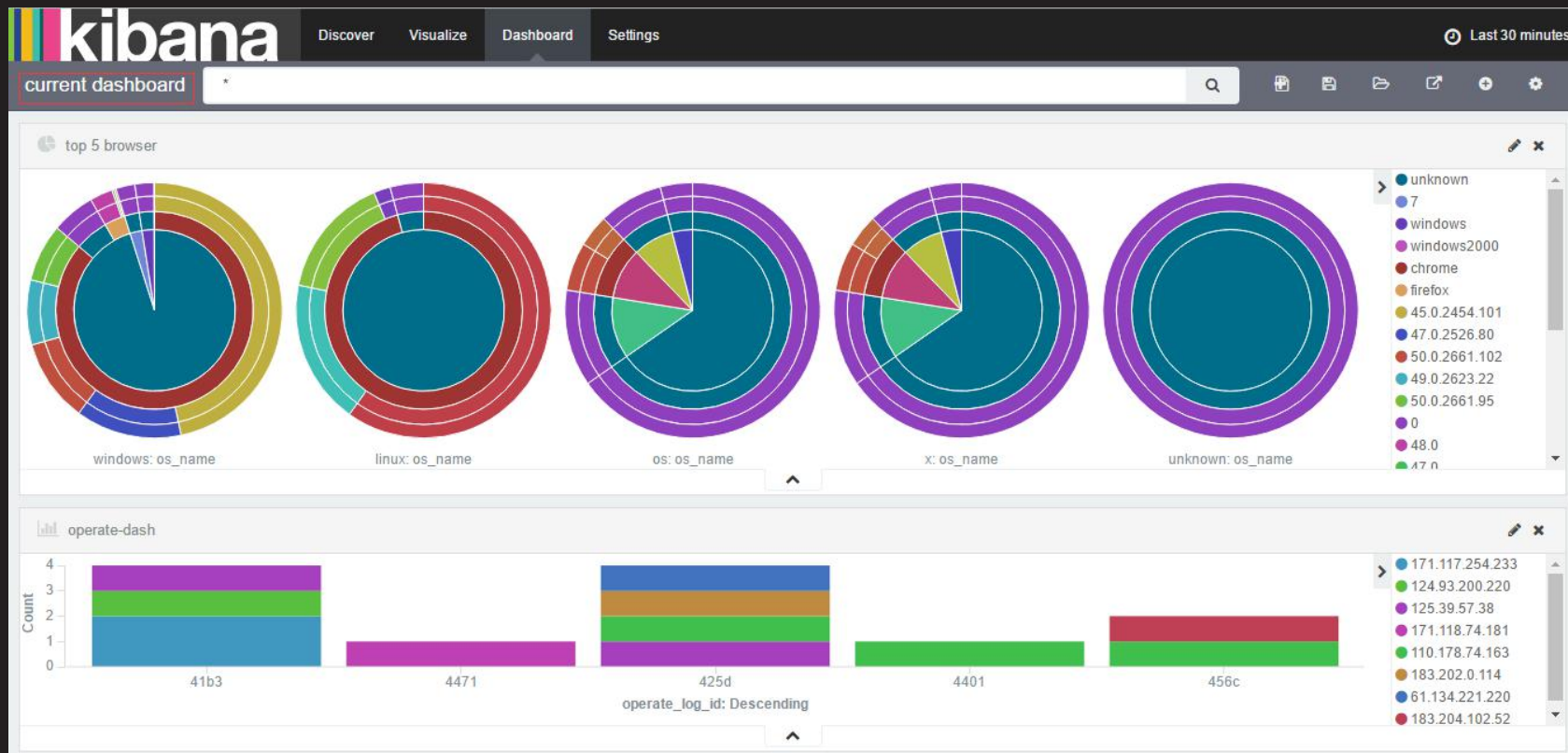
# ELK 是什么样子的？

课堂练习



# ELK 是什么样子的？

课堂练习



# 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

- 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

- 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

- 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



# Elasticsearch部分

- 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 )





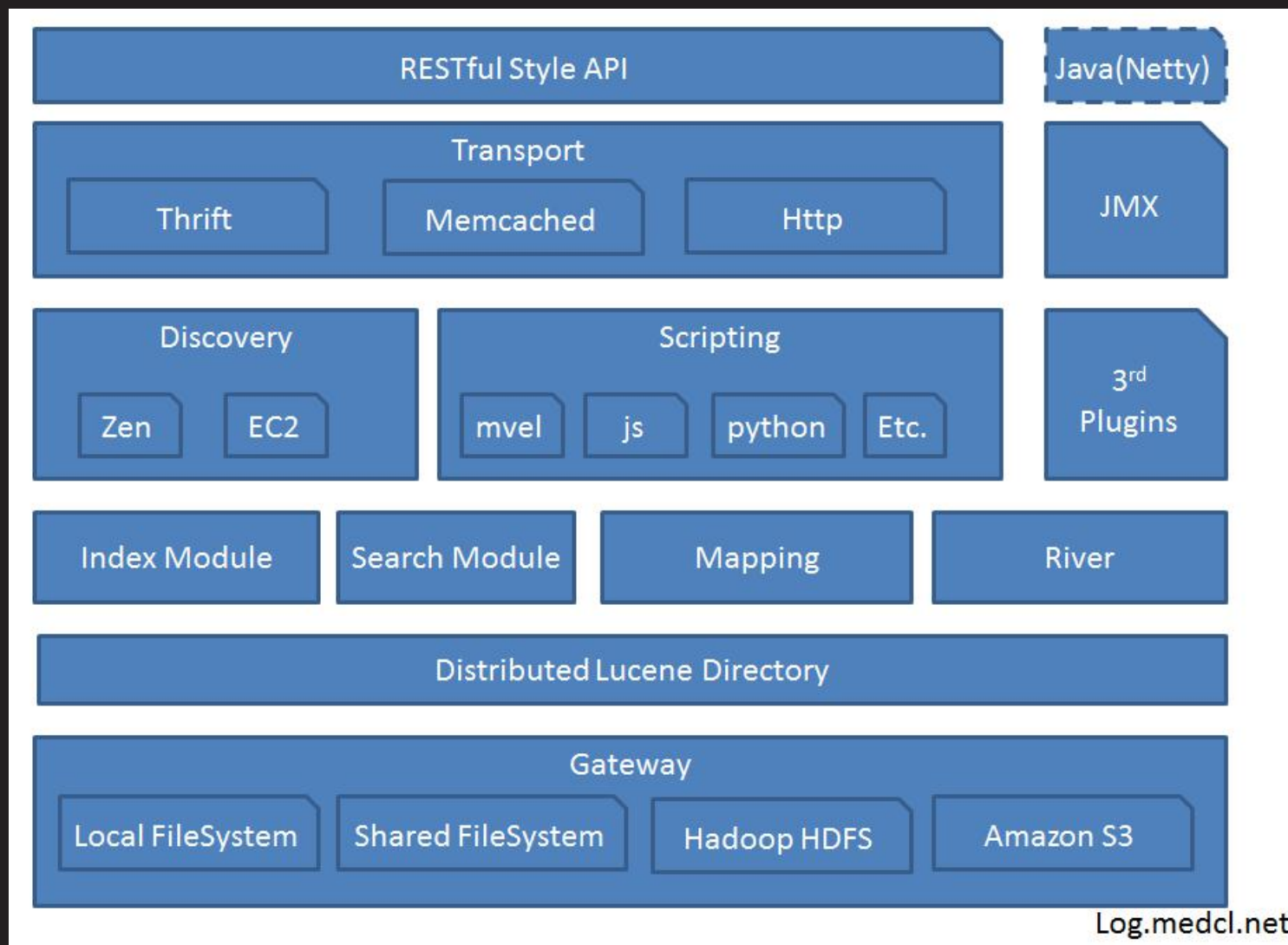
# Elasticsearch部分

- 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



# Elasticsearch部分

- 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



# Elasticsearch部分

- 步骤 1，设置 ip 与主机名称对应关系

- 配置 /etc/hosts

- 192.168.4.11 node1

- 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



# Elasticsearch部分

- 步骤 3

- 安装 ES

- ```
rpm -ivh elasticsearch-2.3.4-1.noarch
```

- 步骤 4

- 修改配置文件

- elasticsearch.yml

- ```
network.host: 0.0.0.0
```



# Elasticsearch部分

- 步骤 5
  - 启动服务，设置自启动  
`systemctl enable elasticsearch`  
`systemctl start elasticsearch`
  - 验证：  
`netstat -ltunp`
  - 能够看到 9200，9300 被监听



# Elasticsearch部分

- 通过浏览器或 curl 访问 9200 端口

curl <http://192.168.4.11:9200/>

```
{
  "name" : "node1",
  "cluster_name" : "my-es",
  "version" : {
    "number" : "2.3.4",
    .....
    "build_snapshot" : false,
    "lucene_version" : "5.5.0"
  },
  "tagline" : "You Know, for Search "
}
```





# Elasticsearch部分

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



# Elasticsearch部分

- 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: my-es  
node.name: node1  
network.host: 0.0.0.0  
discovery.zen.ping.unicast.hosts: ["node1", "node2",  
"node3"]
```



# Elasticsearch部分

- 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（有可能会有一定的延时，需要等待几十秒）



# Elasticsearch部分

- ES 集群配置

- 验证集群，使用 ES 内置字段 `_cluster/health`

```
curl http://192.168.4.11:9200/_cluster/health?pretty
```

```
{
  "cluster_name" : "my-es",
  "status" : "green",
  .....
  "number_of_nodes" : 5,
  "number_of_data_nodes" : 5,
  .....
  "task_max_waiting_in_queue_millis" : 0,
  "active_shards_percent_as_number" : 100.0
}
```



# Elasticsearch部分

- 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
  - }



# Elasticsearch部分

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



# ES 插件的安装与使用



# Elasticsearch部分

- 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





# Elasticsearch部分

- 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



# Elasticsearch部分

- 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` 删除使用 `remove` 指令



# ES head 插件

http://192.168.4.15:9200/ 连接 my-es 集群健康值: green (10 of 10)

**Elasticsearch** 概览 索引 数据浏览 基本查询 [+] 复合查询 [+] 信息 ▾


**集群概览** 集群排序 ▾ Sort Indices ▾ View Aliases ▾ Index Filter 刷新 ▾


**weblog**  
size: 46.0Mi (91.2Mi)  
docs: 14,005 (28,010)  
信息 ▾ 动作 ▾


● node1	<span>信息 ▾</span> <span>动作 ▾</span>	3	4
★ node2	<span>信息 ▾</span> <span>动作 ▾</span>	1	3
● node3	<span>信息 ▾</span> <span>动作 ▾</span>	0	2
● node4	<span>信息 ▾</span> <span>动作 ▾</span>	2	4
● node5	<span>信息 ▾</span> <span>动作 ▾</span>	0	1


# ES Kopf 插件

课堂练习

 cluster

 nodes

 rest

 more

my-es @ node5

5 nodes

2 indices

12 shards

14,006 docs

91.23MB

filter nodes by name

☒ ☆ master

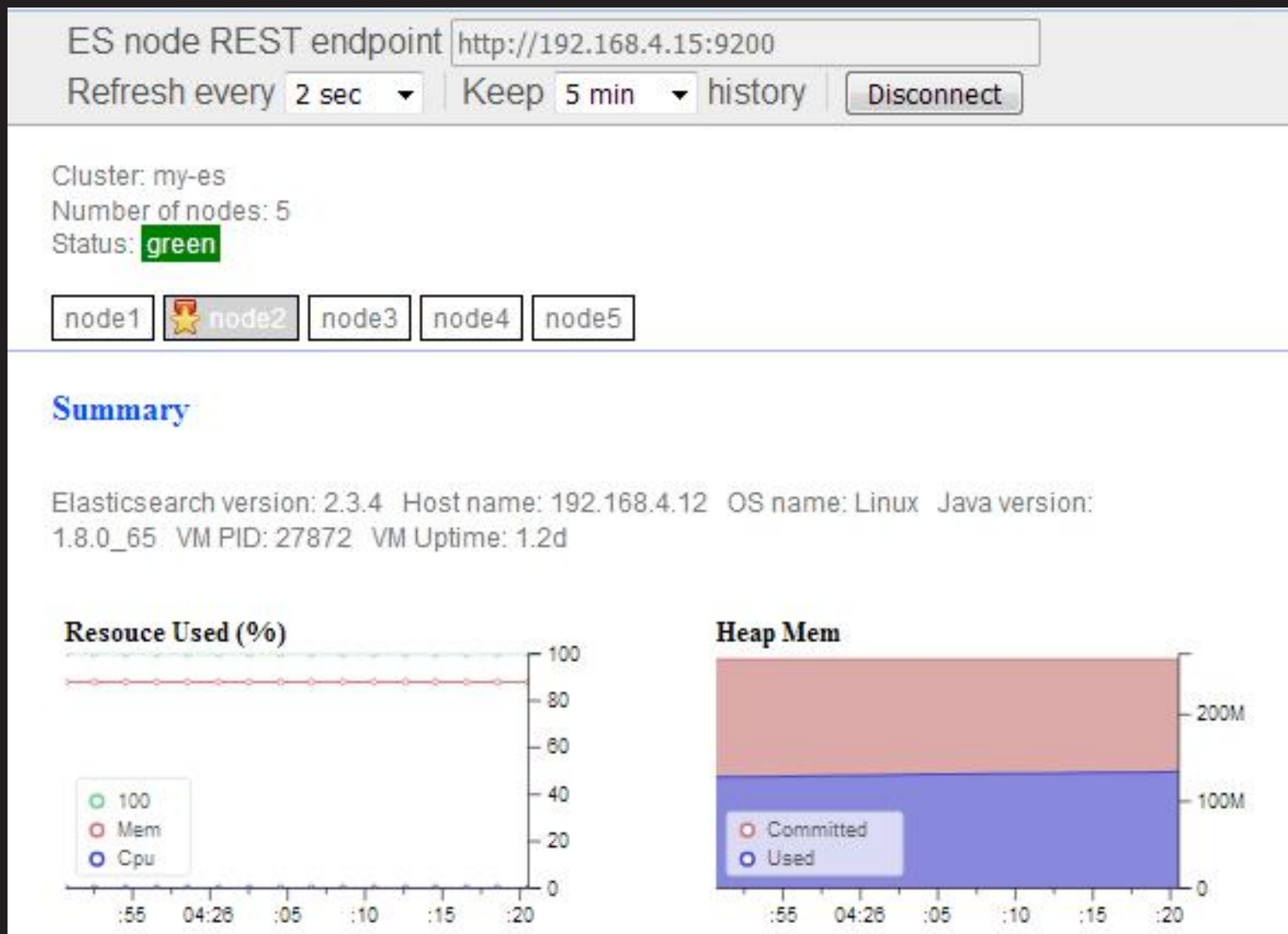
☒ 🗄 data

☒ 🔍 client

name ^	load average	cpu %	heap usage %	disk usage %	uptime
☆ node1 🗄 192.168.4.11 192.168.4.11:9300 JVM: 1.8.0_65 ES: 2.3.4	N/A	0.0	16.0 used: 170.30MB max: 1007.38MB	9.0 free: 15.94GB total: 17.46GB	2h.
★ node2 🗄 192.168.4.12 192.168.4.12:9300 JVM: 1.8.0_65 ES: 2.3.4	0.0	0.0	10.0 used: 109.82MB max: 1007.38MB	8.0 free: 16.06GB total: 17.46GB	1d.
☆ node3 🗄 192.168.4.13 192.168.4.13:9300 JVM: 1.8.0_65 ES: 2.3.4	N/A	0.0	20.0 used: 206.30MB max: 1007.38MB	8.0 free: 16.06GB total: 17.46GB	1d.
☆ node4 🗄 192.168.4.14 192.168.4.14:9300 JVM: 1.8.0_65 ES: 2.3.4	N/A	0.0	21.0 used: 217.80MB max: 1007.38MB	8.0 free: 16.06GB total: 17.46GB	1d.
☆ node5 🗄 192.168.4.15 192.168.4.15:9300 JVM: 1.8.0_65 ES: 2.3.4	N/A	0.0	11.0 used: 116.56MB max: 1007.38MB	8.0 free: 16.04GB total: 17.46GB	1d.



# ES bigdesk 插件



# HTTP 与 RESTful API



# ES RESTful API 部分

- 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



# ES RESTful API 部分

- json
  - JSON的全称是“ 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 传输的就是一个字符串
  - python 中对应的 字符串, 列表, 字典都可以转换成对应的 json 格式





# ES RESTful API 部分

- Rest 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://192.168.4.15:9200/\\_cat/health?v](http://192.168.4.15:9200/_cat/health?v)
  - help 显示帮助信息  
[http://192.168.4.15:9200/\\_cat/health?help](http://192.168.4.15:9200/_cat/health?help)



# ES RESTful API 部分

- Rest API 的简单使用

- nodes 查询节点状态信息

- [http://192.168.4.15:9200/\\_cat/nodes?v](http://192.168.4.15:9200/_cat/nodes?v)

- 索引信息

- [http://192.168.4.15:9200/\\_cat/indices?v](http://192.168.4.15:9200/_cat/indices?v)



# ES RESTful API 部分

- HTTP Methods 和 RESTful API 设计
  - HTTP Methods 也叫 HTTP Verbs, 它们是 HTTP 协议的一部分, 主要规定了 HTTP 如何请求和操作服务器上的资源, 常见的有GET, POST等
  - HTTP Methods 一共有九个, 分别是 GET, HEAD, POST, PUT, DELETE, TRACE, OPTIONS, CONNECT, PATCH



# ES RESTful API 部分

- HTTP Methods

- 在RESTful API 设计中，常用的有POST，GET，PUT，PATCH 和 DELETE。分别对应对资源的创建，获取，修改，部分修改和删除操作。
- 我们默认访问 ES API 的方法是 GET，如果要对数据库增加、删除、修改数据我们还要使用对应的方法
- GET      查询
- POST     增加
- PUT      更改
- DELETE 删除



# ES RESTful API 部分

- RESTful API 增加
  - 创建一个 school 的 (Index) 和一个 students (Type)
  - 并增加一条信息

```
curl -XPUT 'http://192.168.4.11:9200/school/students/1' -d
'{
  "title": "devops",
  "name":{
    "first": "guzhang",
    "last": "wu"
  },
  "age": 25
}'
```



# ES RESTful API 部分

- RESTful API 更改
  - 修改 school 下面 students 的第一个文档中的 age 信息，从 25 修改为 30

```
curl -XPOST  
'http://192.168.4.11:9200/school/students/1/_update' -d '{  
  "doc":{  
    "age": 30  
  }  
'
```



# ES RESTful API 部分

- RESTful API 查询

- 查询刚刚创建的文档信息

- ```
curl -XGET 'http://192.168.4.11:9200/school/students/1'
```

- 只查询 name 和 age

- ```
curl -XGET  
'http://192.168.4.11:9200/school/students/1?_source=name,age'
```



# ES RESTful API 部分

- RESTful API 删除

- 删除刚才创建的文档

- ```
curl -XDELETE 'http://192.168.4.14:9200/school/students/1'
```

- 删除 school

- ```
curl -XDELETE 'http://192.168.4.14:9200/school'
```





# kibana 安装

---

# Kibana 安装

- kibana是什么
  - 数据可视化平台工具
- 特点：
  - 灵活的分析 and 可视化平台
  - 实时总结和流数据的图表
  - 为不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安装
  - kibana 的安装非常简单，我们使用 rpm 方式安装  
`rpm -ivh kibana-4.5.2-1.x86_64.rpm`
  - 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 安装

- kibana.yml 的配置
  - server.port: 5601
  - server.host: "0.0.0.0"
  - elasticsearch.url: "http://192.168.4.13:9200"
  - kibana.index: ".kibana"
  - kibana.defaultAppId: "discover"
  - elasticsearch.pingTimeout: 1500
  - elasticsearch.requestTimeout: 30000
  - elasticsearch.startupTimeout: 5000



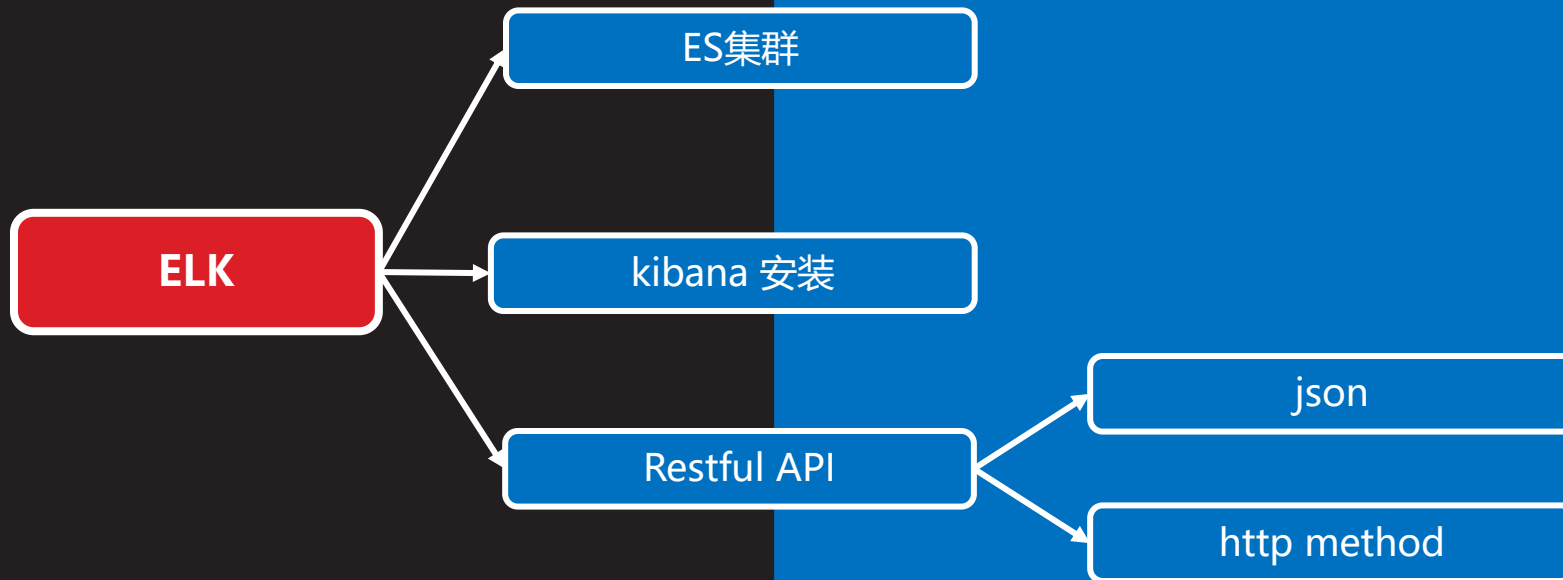
# Kibana 安装

- 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  
`http://192.168.4.20:5601/`



# 知识点总结

---



# 总结和答疑

---