基于Consul的多Beats接入管控与多ES搜索编排

梁成
腾讯云, barryliang@tencent.com

拥抱开源、释放云原生的力量

- 背景与挑战
- 多Beats/Logstash接入管控
- 多ES搜索编排系统
- 日志AIOps探索

背景与挑战

如何降低日志接入门槛

如何保证日志实时上报

如何保障日志采集不影响业务

如何做配置标准化

如何帮助业务快速排障

如何提供方便便捷的性能分析调优能力

...

100+ 产品数量

1000 人员规模

+

10000 主机规模

+

多Beats/Logstash接入管控

提供多产品接入管理，多beats标准化、界面化、自动化的日志接入方案

elastic

案例:1000+业务10000+台主机如何快速实现日志接入？

业务规模

1000+业务、 10000+业务主机、每天百T日志增量

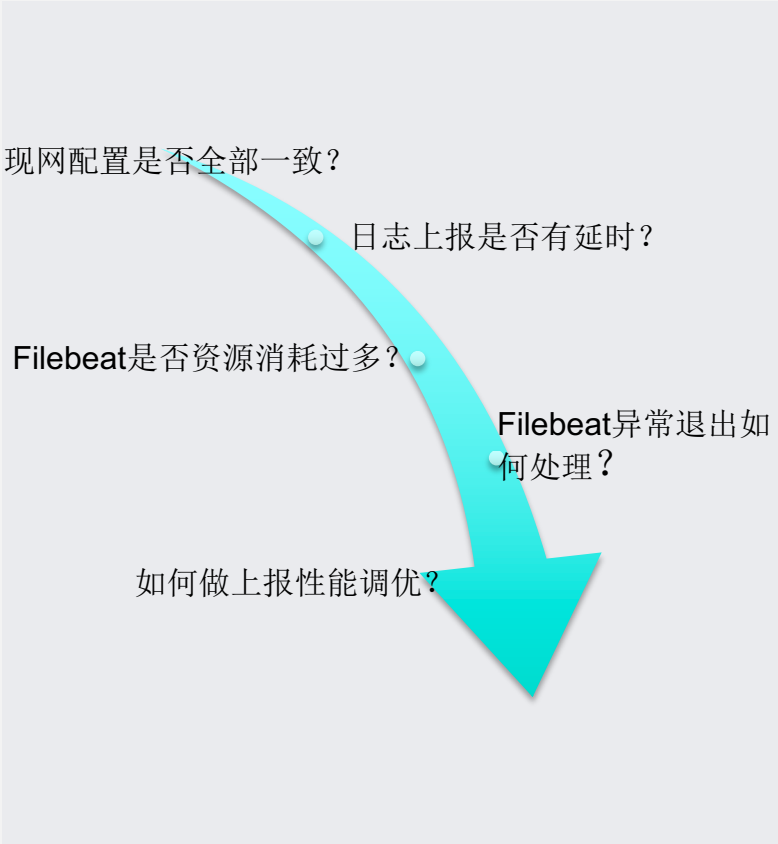
日志需求

收集业务日志文件用于故障分析与告警监控
收集主机性能数据做容量分析
日志热数据保存七天
历史数据冷备一个月

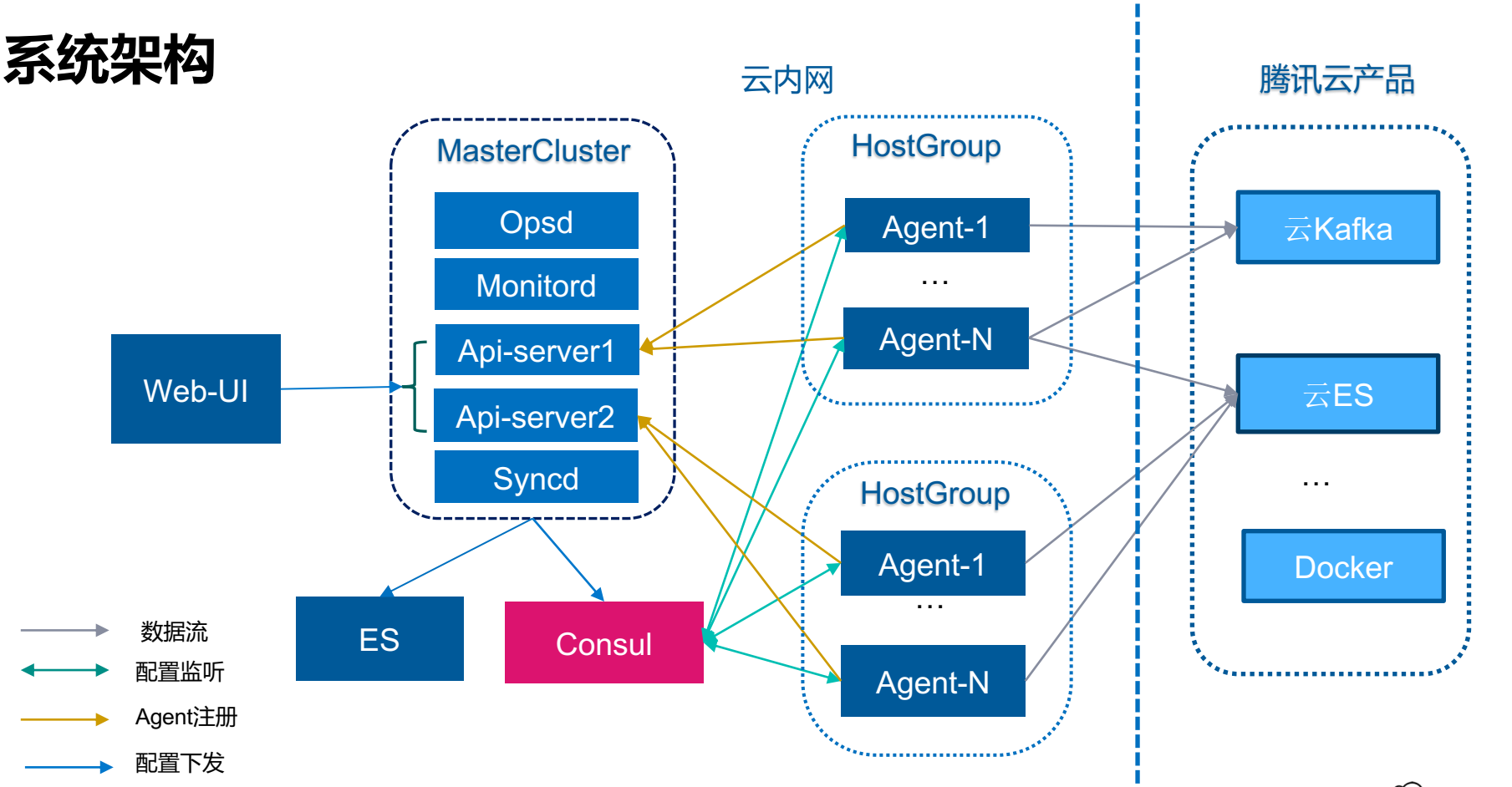
其他诉求

日志上报不能影响核心业务
数据上报延时可感知

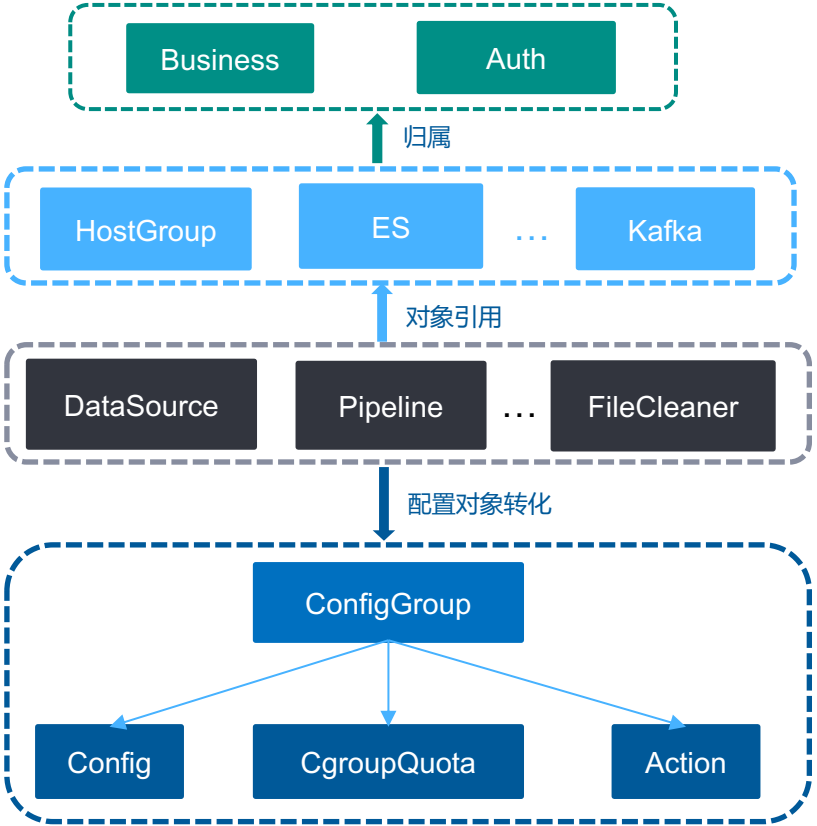
传统Beats接入流程

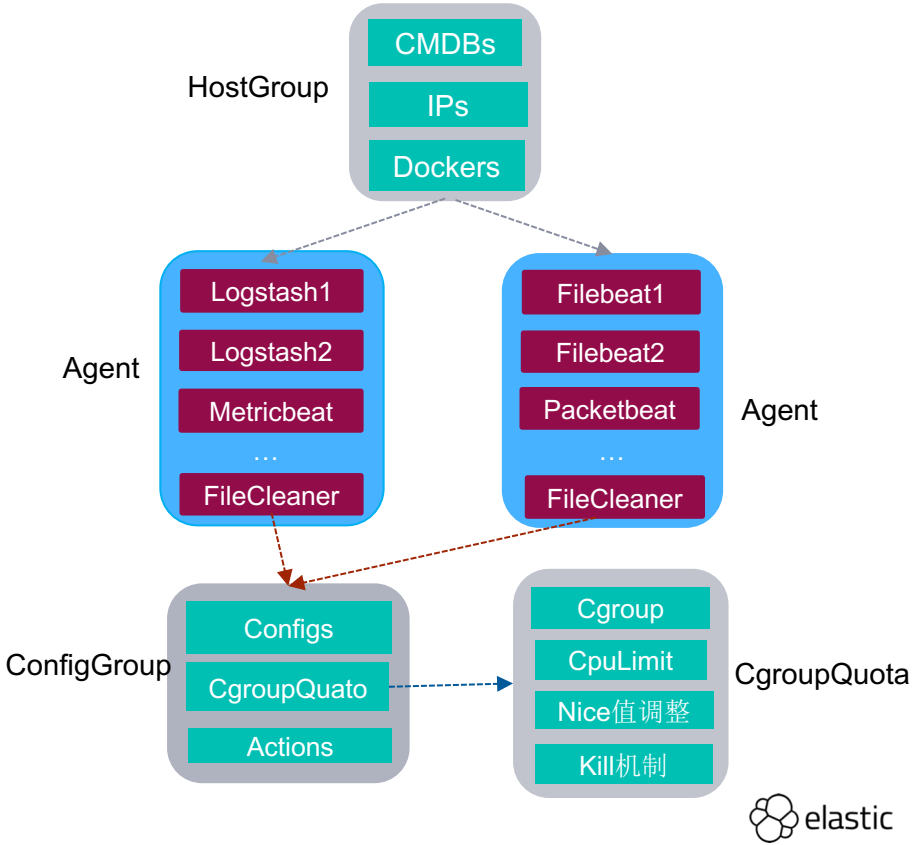


系统架构


云内网
腾讯云产品
MasterCluster
Opsd
Monitord
Api-server1
Api-server2
Syncd
Web-UI
ES
Consul
HostGroup
Agent-1
…
Agent-N
HostGroup
Agent-1
…
Agent-N
云Kafka
云ES
…
Docker
数据流
配置监听
Agent注册
配置下发

数据模型


Business
Auth
归属
HostGroup
ES
…
Kafka
对象引用
DataSource
Pipeline
…
FileCleaner
配置对象转化
ConfigGroup
Config
CgroupQuota
Action


HostGroup
CMDBs
IPs
Dockers
Agent
Logstash1
Logstash2
Metricbeat
…
FileCleaner
Filebeat1
Filebeat2
Packetbeat
…
FileCleaner
Agent
ConfigGroup
Configs
CgroupQuato
Actions
Cgroup
CpuLimit
Nice值调整
Kill机制
CgroupQuota

Agent管理

时序图

Agent注册

Agent启动首先向Consul获取Master服务列表，并向Master发起Agent注册逻辑，获取agent id

配置获取

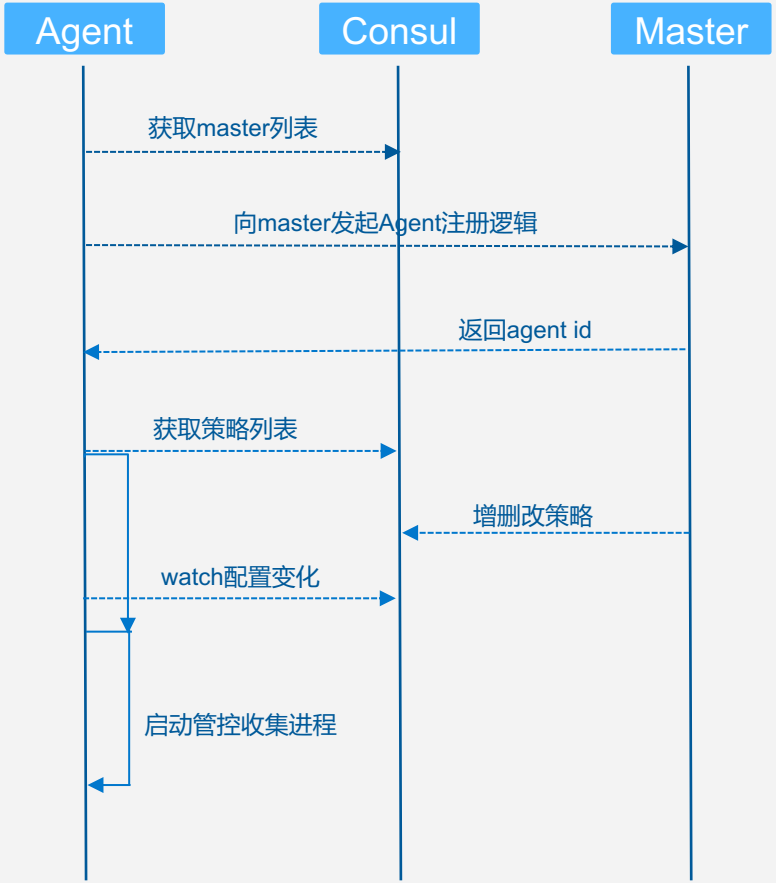
从Consul中获取当前agent的配置组列表，并启动多个采集进程

配置变更感知

watch到Consul对应的agent id路径，实时感知配置变化，并对启动的进程列表做重启清理等工作

管理多Beats/logstash

Beats等以agent子进程启动其管理这些进程的cpu/内存等资源



```
[root@ ~]# cat /data/log_ops_agent/qcloud_fbeat
/3909bc /config/qcloud_fbeat.yml
fields:
  name:
  region: ${LOG_OPS_REGION:none}
  report_ip: ${LOG_OPS_IP:none}
fields_under_root: true
filebeat.inputs:
- type: log
  enable: true
  paths:
  - /*.log
  tail_files: true
  ignore_older: 24h
  clean_inactive: 25h
  harvester_limit: 100
  fields_under_root: true
  scan_frequency: 2s
  harvester_buffer_size: 1638400
  max_bytes: 2485760
- type: log
  enable: true
  paths:
  - /*.log
  tail_files: true
  ignore_older: 24h
  clean_inactive: 25h
  harvester_limit: 100
  fields_under_root: true
  scan_frequency: 2s
  harvester_buffer_size: 1638400
  max_bytes: 2485760
- type: log
  enable: true
  paths:
  - /*.log
  tail_files: true
  ignore_older: 24h
  clean_inactive: 25h
  harvester_limit: 100
  fields_under_root: true
  scan_frequency: 2s
```


```
{
  "type": "",
  "id": "3909bd45b00a5b1",
  "version": 3,
  "name": "price_业务日志",
  "description": "jf_price_log",
  "version_info": "",
  "create_at": "2019-02-21T11:02:21+08:00",
  "modified_at": "2019-09-03T10:54:51.604607675+08:00",
  "auth": {
   "creator": "admin",
   "owners": [
    "admin"
   ],
   "readers": null
  },
  "configs": [
   {
    "json_values": "{}",
    "template": "fields:\n  name: jf_price_log\n  region: ${LOG_OPS_REGION:none}\n  report_ip:
    "meta": {
     "file": "/data/log_ops_agent/qcloud_fbeat/3909bd45b00a5b1/config/qcloud_fbeat.yml",
    },
    "content": "fields:\n  name: jf_price_log\n  region: ${LOG_OPS_REGION:none}\n  report_ip:
   },
   {
   }
  ],
  "pre_actions": [
   {
    "exe": "mkdir",
    "args": [
     "-p",
     "/data/log_ops_agent/qcloud_fbeat/3909bd45b00a5b1/monitor"
    ]
   }
  ],
  "actions": [
   {
    "exe": "qcloud_fbeat",
    "args": [
     "-c",
     "/data/log_ops_agent/qcloud_fbeat/3909bd45b00a5b1/config/qcloud_fbeat.yml",
    ]
   },
  ],
  "cgroup_quota": {
   "cpu_percent": 10,
   "mem_quota_type": "percent"
```


配置组 id 与版本

配置文件列表

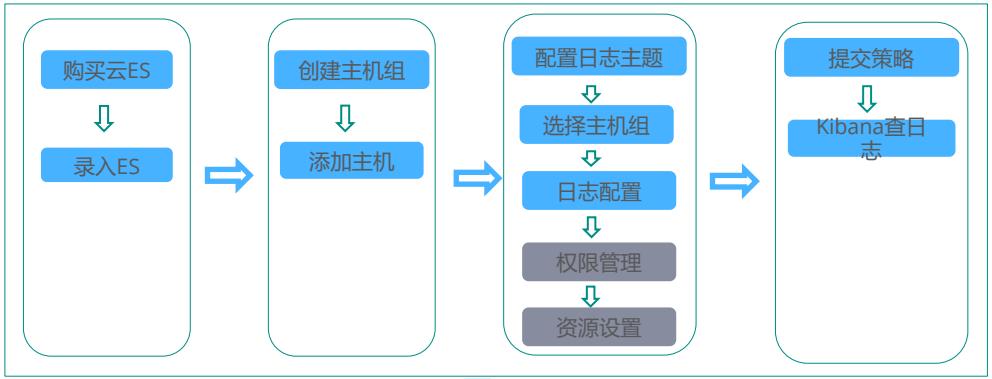
生成配置前的预处理命令

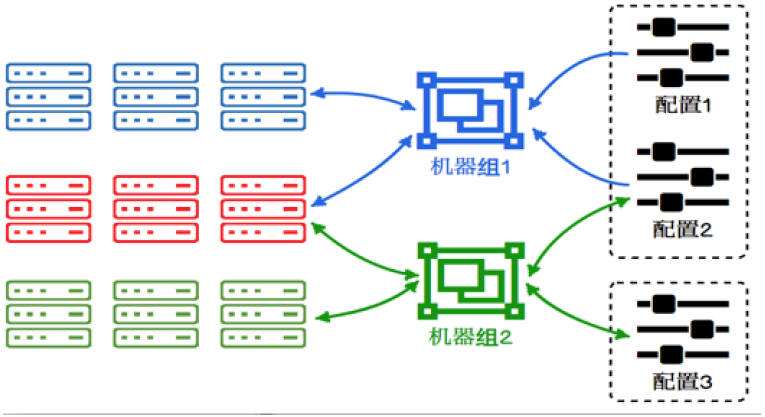
程序启动命令列表

当前配置组对应的资源配额

stic

日志接入


购买云ES
录入ES
创建主机组
添加主机
配置日志主题
选择主机组
日志配置
权限管理
资源设置
提交策略
Kibana查日志


机器组1
机器组2
配置1
配置2
配置3


源信息
接入配置
* 接入方式：
日志文件
日志文件
网络抓包
指标获取
路径(glob查找)：
/data/test/
查看帮助
配置详情
设置标签：
+ 新增
开启Json分析：
排除关键字：
+ 新增
支持正则,点击查看配置说明
包括关键字：
+ 新增
支持正则,点击查看配置说明
开启多行解析：
高级配置
排除文件设置：
+ 新增
支持正则,点击查看配置说明
日志文件忽略时长设置：
24
小时
Cancel
提交

当前收益

快

- 快速接入(5min)
- 配置UI化标准化
- 配置变更实时感知
- 部署全自动化
- 多Beats支持

稳

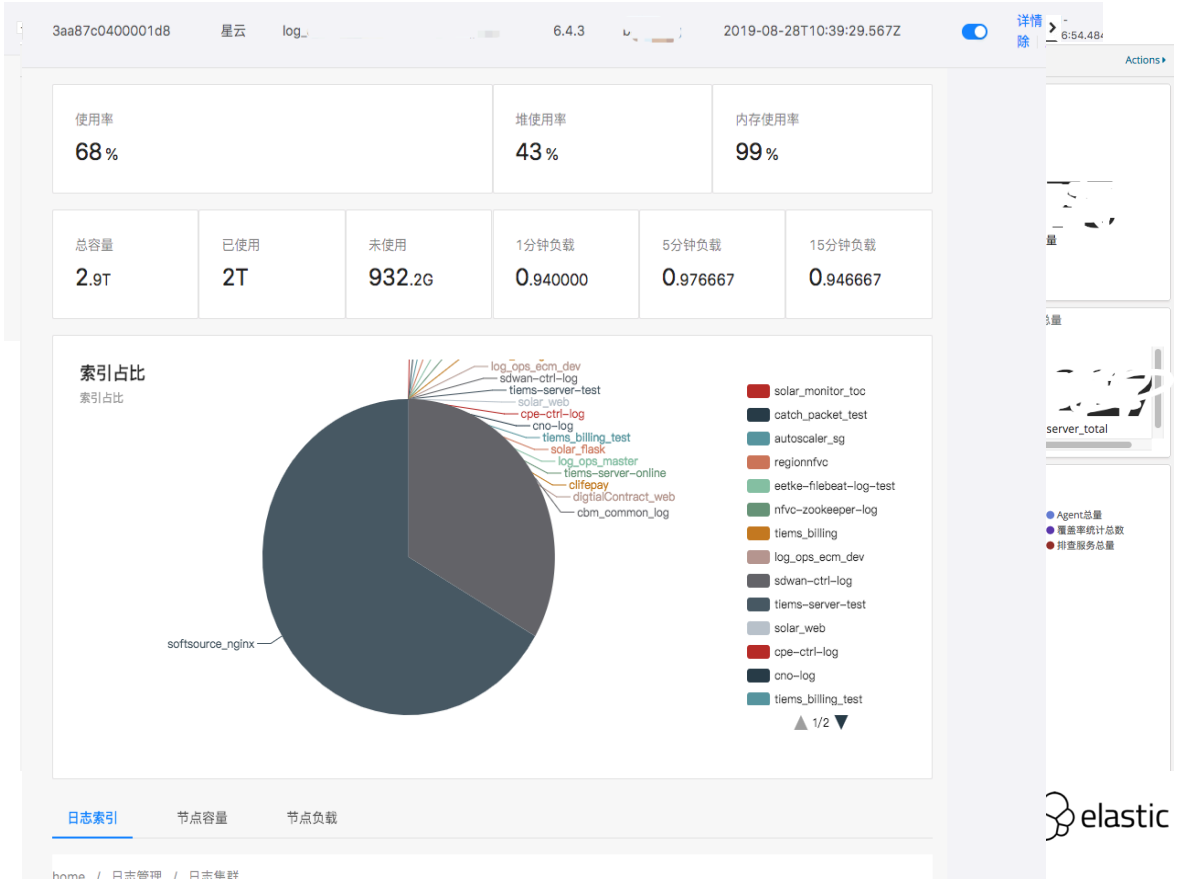
- Beats运行时cpu/mem可控
- Agent监控视图
- 离线/容量/延时监控
- 分布式集群管理
- 异常快速定位

准

- 关联公司CMDB
- 资源权限管理
- 配置灰度控制发布
- 配置一致性检测
- 日志覆盖率

案例:如何管控整个日志数据流相关资源性能与容量？

- Agent运行时监控
- Beats cpu/mem管控
- 日志延时分析
- 日志覆盖率分析
- ES/kafka容量管理



案例:高并发写入场景下Beats与ES性能优化

对比	自建	日志平台接入
上报延时分析	无法实时感知日志积压情况	日志延时dashboard快速定位
Filebeat性能分析	依赖白金版monitor机制	不依赖es版本、结合cpu/mem限额配置与实时指标采集分析
Filebeat性能调优	需要修改配置文件不断尝试	界面提交核心参数并结合延时图对比分析
Filebeat性能管控	日志量太大Cpu飙升影响业务	精准控制资源消耗防止异常减少抖动
Es写入性能调优	修改配置文件不断观察数据情况	基于ES压测报告给出专家级es参数优化建议
参数优化体验	修改配置文件、参数调优相对麻烦	全UI化、一站式处理

配置UI化

配置UI化开发思路

嵌套式表单

大表单套小表单，所有表单都是以angular组件形式开发，保证代码的可复用性与质量

配置分级展现

把复杂配置独立成高级选择，并设置默认值，并在复杂项给出有效帮助

自定义组件

编写大量自定义小组件，比如cmdb设置，时间设置等组件提高用户体验，尽量减少直接填文本

前后端强类型

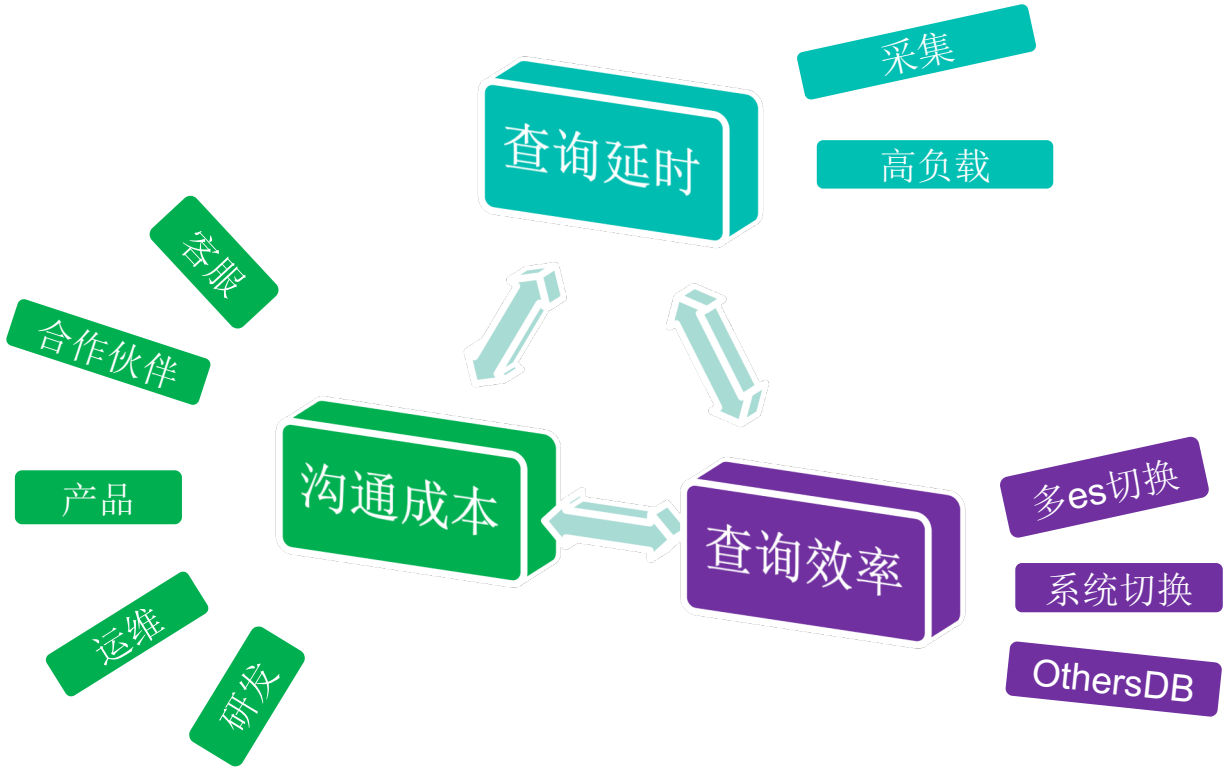
前端采用基于ts 的angular开发所有数据定义与后端golang数据保持强一致，保证数据一致性

多ES搜索编排系统

提供多ES多索引搜索编排功能，帮助业务快速定位异常

elastic

故障定位遇到的困扰



elastic

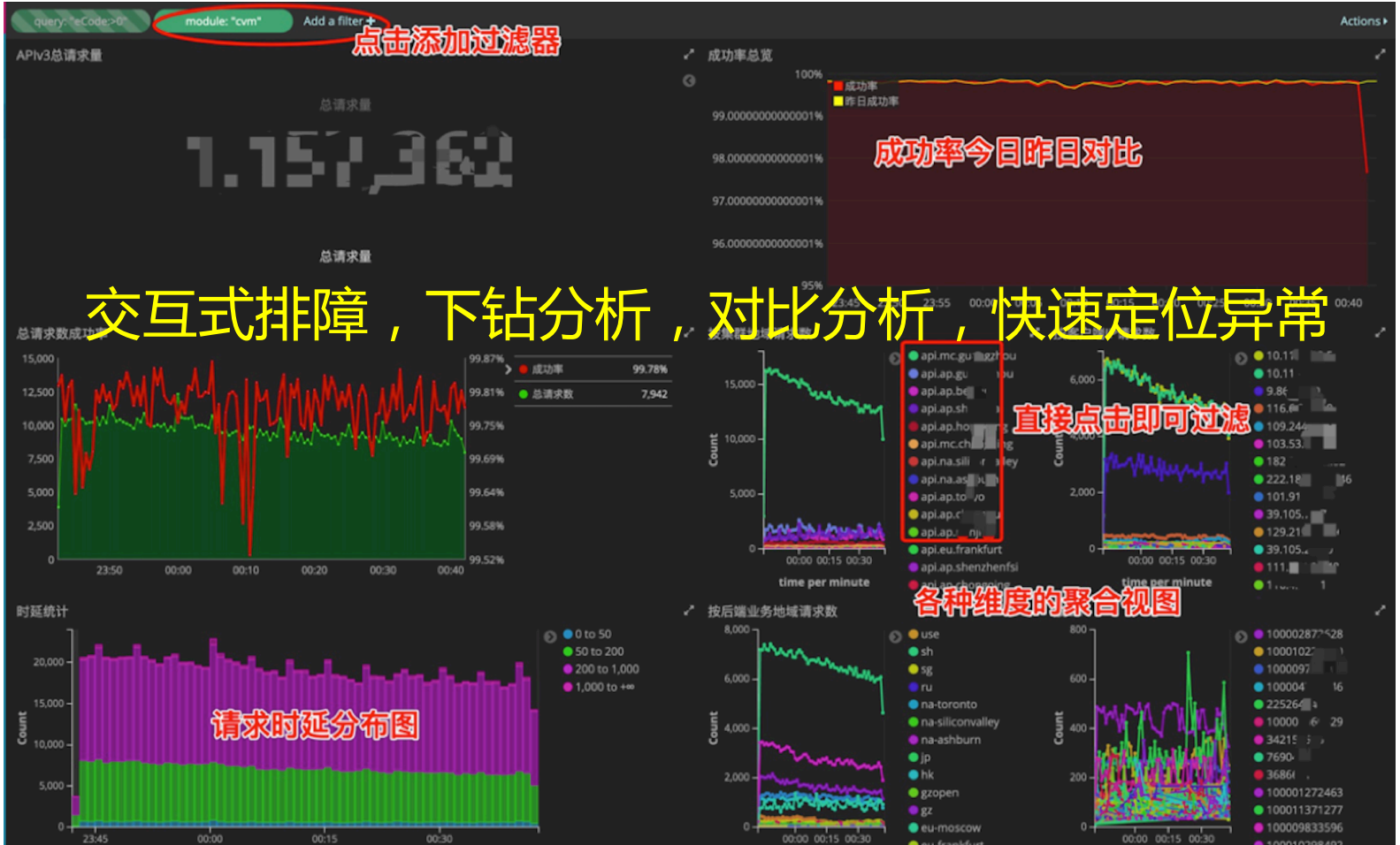
案例:非APM场景下多组件日志搜索探索

日志AIOps探索

日志最佳实践、拨测系统、智能排障、AIOps探索

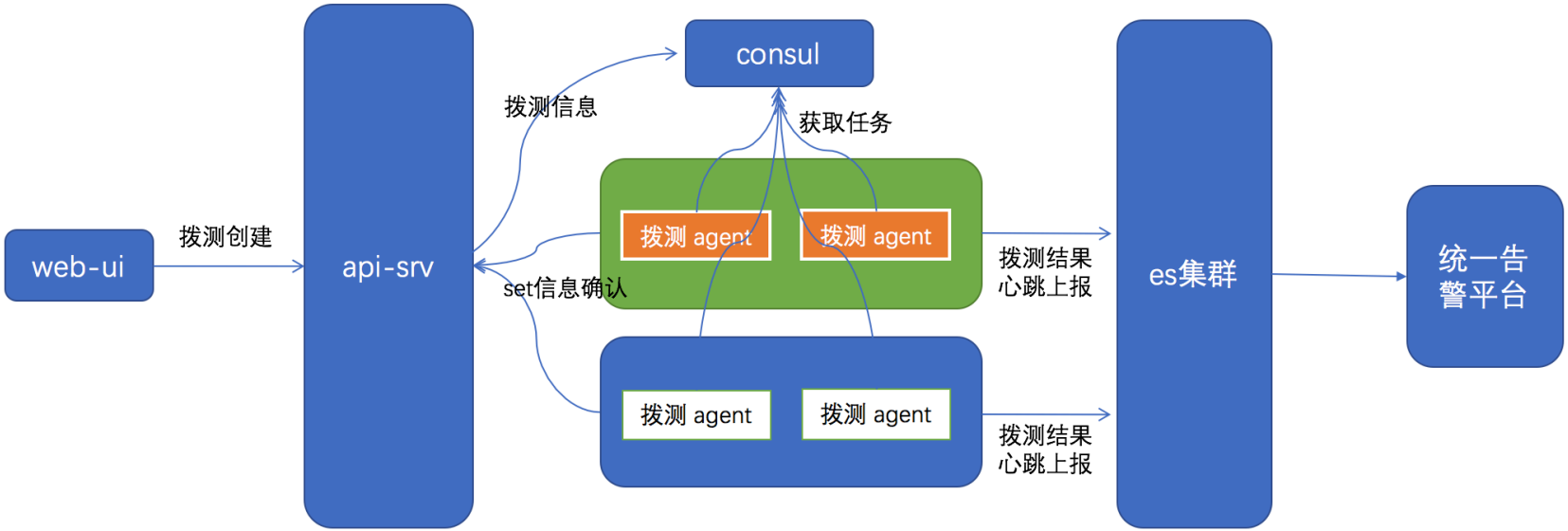
elastic

案例: 基于Kibana的交互式排障



elastic

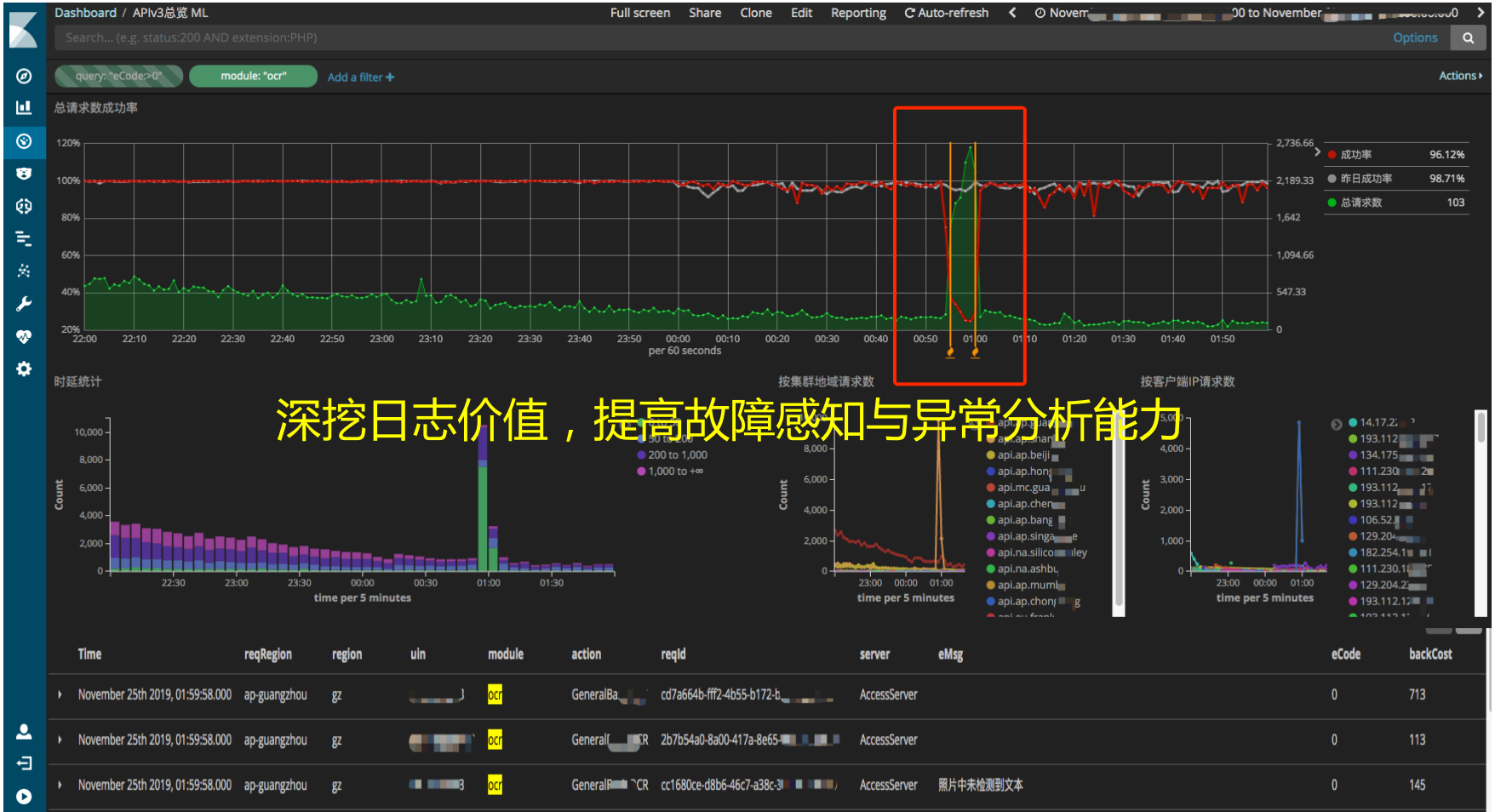
案例:如何基于ELK构建内网拨测系统?



基于Healthbeat与Metricbeat构建分布式、全region覆盖、协议支撑丰富的内网拨测平台

elastic

案例: 使用ES原生ML能力提供排障效率



技术栈

技术选型

前端

TypeScript、Angular 8、 Ng-zorro

后台

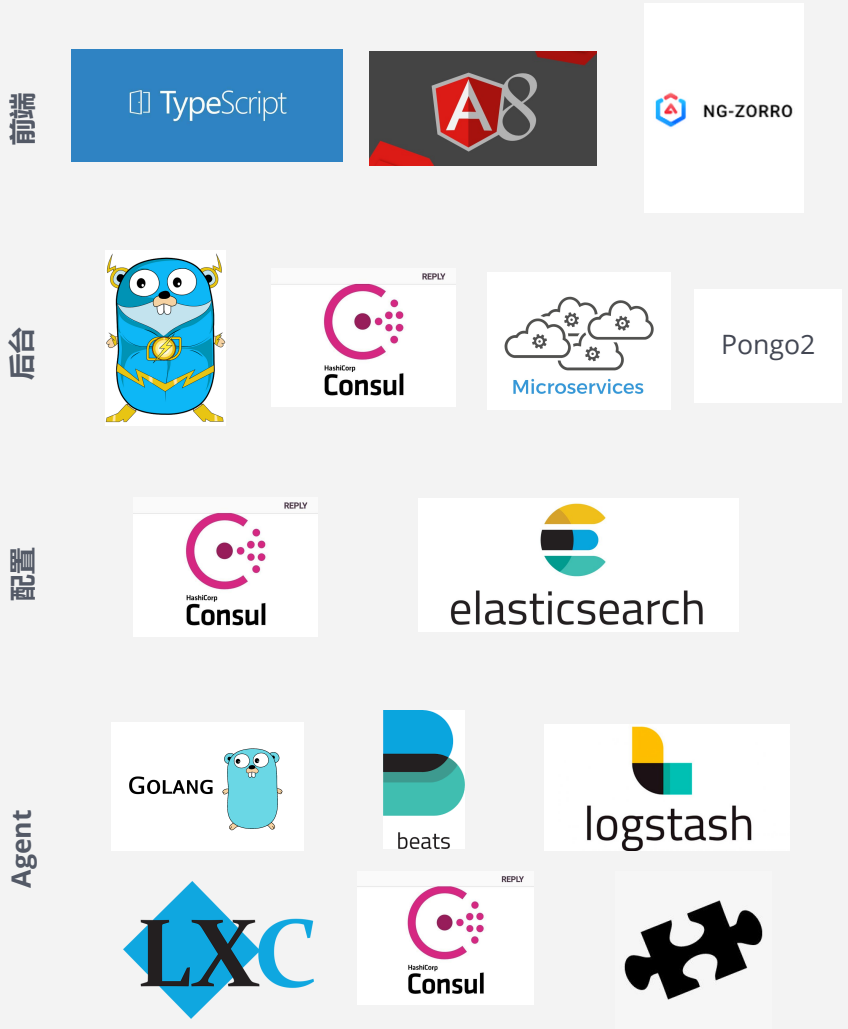
Golang、Iris、Consul、微服务化设计、Pongo2模板管理

配置

Consul、Elasticsearch

Agent

Golang、ELK beats、Logstash、LXC资源管控、
Consul、其他自定义插件



谢谢大家！

elastic