

日志服务  
实践教学  
产品文档



腾讯云

**【版权声明】**

©2013-2024 腾讯云版权所有

本文档著作权归腾讯云单独所有，未经腾讯云事先书面许可，任何主体不得以任何形式复制、修改、抄袭、传播全部或部分本文档内容。

**【商标声明】**

及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。

**【服务声明】**

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或默示的承诺或保证。

## 文档目录

### 实践教程

#### 日志采集

采集 Windows 环境日志至 CLS

采集/查询主机文件日志

#### 检索分析

CLB 访问日志分析

Nginx 访问日志分析

CDN 访问日志分析

COS 访问日志分析

Flowlog网络流日志分析

TKE 事件日志分析

TKE 审计日志分析

#### 仪表盘

把 Grafana 的 ES 数据源迁移为 CLS 数据源

#### 监控告警

按时间段分别设置告警触发条件

使用同环比作为告警触发条件

#### 投递和消费

使用 Flink 消费 CLS 日志

使用 DLC (Hive) 分析 CLS 日志

#### 成本优化

节省产品使用成本

# 实践教程

## 日志采集

### 采集 Windows 环境日志至 CLS

最近更新时间：2024-01-20 17:28:40

## 操作场景

本文指导您使用 Winlogbeat 或者 Filebeat 采集 Windows 环境日志上传至 CLS。

## 前提条件

已开通日志服务（Cloud Log Service, CLS），并创建对应资源（如日志集和日志主题）。

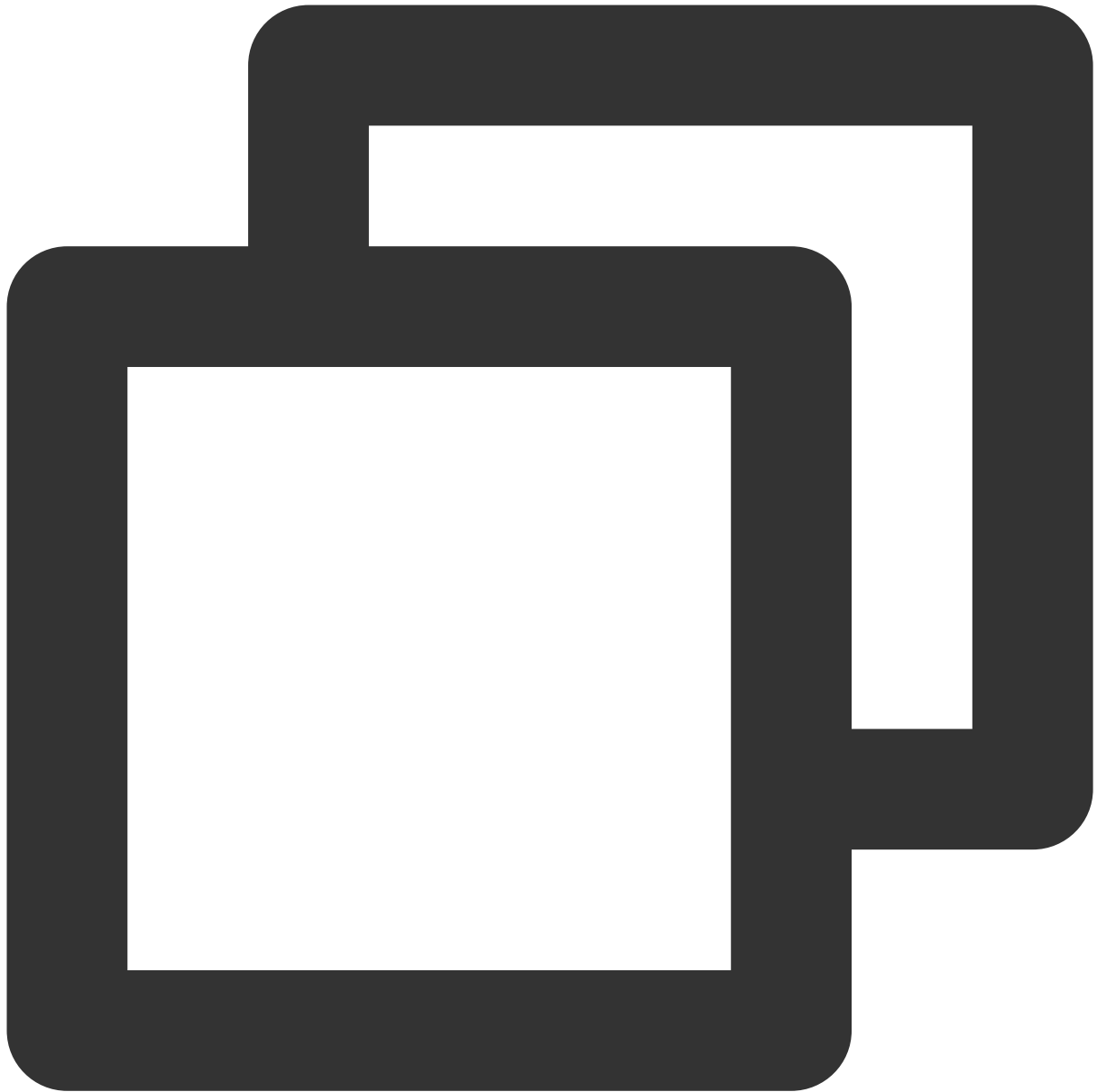
已在 [腾讯云控制台](#) 获取到 SecretId 和 SecretKey。

## 操作步骤

### 使用 Winlogbeat 采集 Windows 事件日志上传 CLS

#### 安装 Winlogbeat

1. 前往 Winlogbeat 官网下载，您需要下载的版本。  
本文以 Winlogbeat 7.6.2 版本为例，[点此前往下载](#)。
2. 将已下载的压缩包解压缩至 C: 盘下。  
建议在 Program Files 目录下新建一个 winlogbeat 文件夹，并解压缩至该文件夹下。
3. 以管理员身份打开 powershell，并执行如下命令。



```
cd C:\\Program Files
cd .\\winlogbeat-7.6.2-windows-x86_64
.\\install-service-winlogbeat.ps1
```

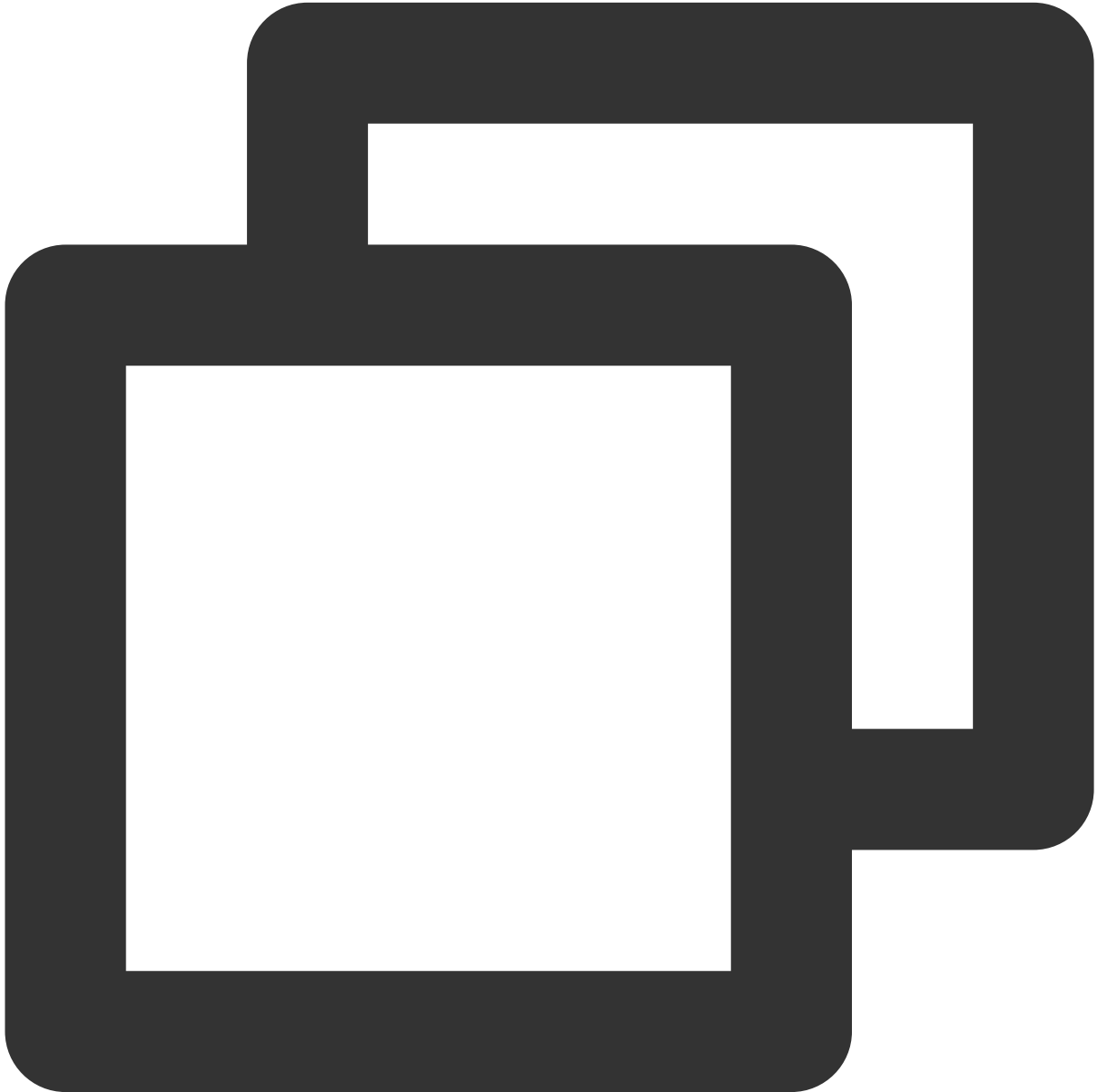
执行过程中，如果报错，请输入 `Set-ExecutionPolicy -ExecutionPolicy RemoteSigned` 命令并选择 `y`，然后重新输入以上命令。

执行命令后，如果返回下图结果则表示成功。

```
PS C:\Program Files\winlogbeat\winlogbeat-7.6.2-windows-x86_64> .\install-service-winlogbeat.ps1

Status   Name           DisplayName
-----   -
Stopped  winlogbeat     winlogbeat
```

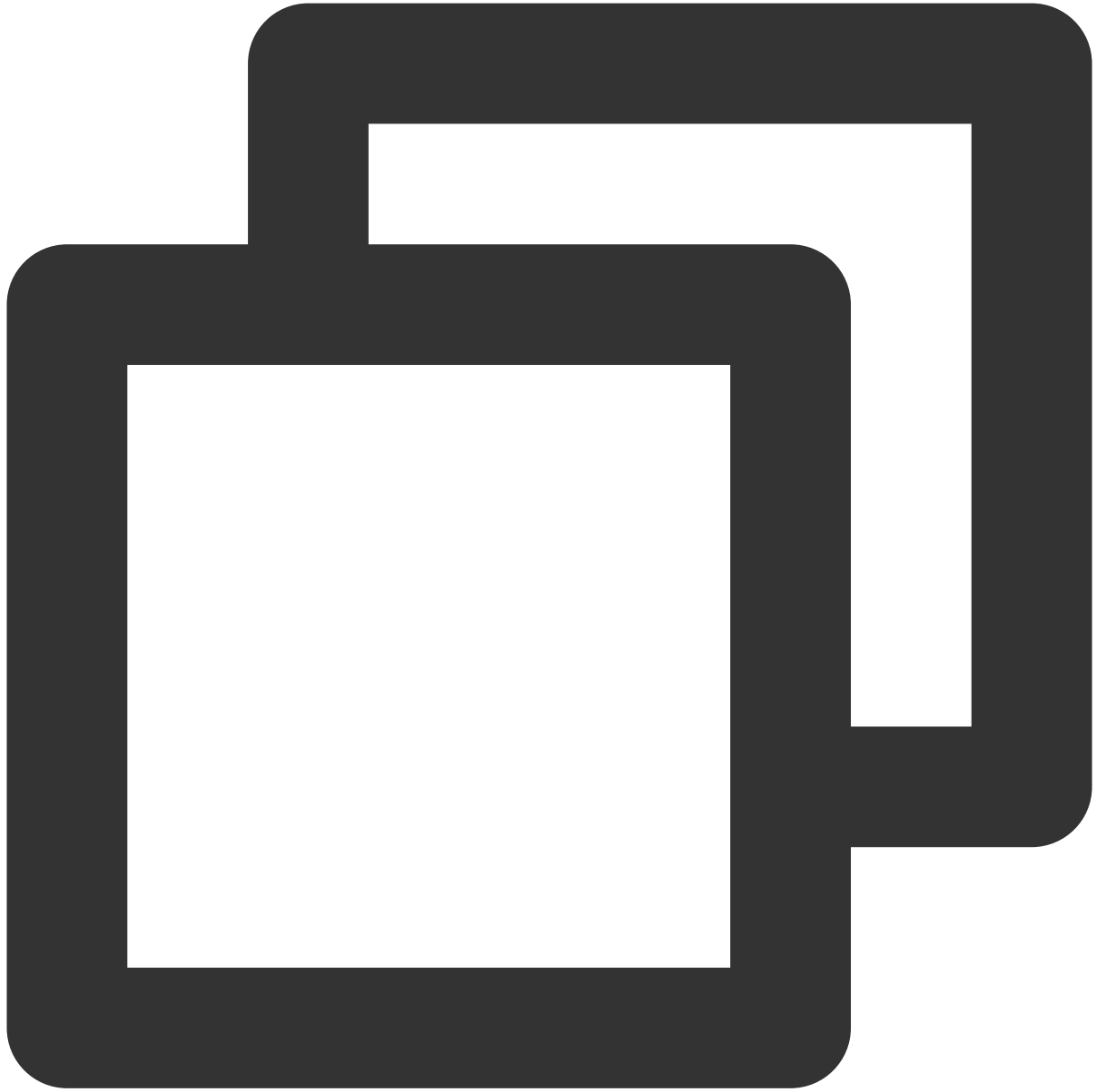
4. 执行如下命令，测试环境是否正常。



```
.\winlogbeat.exe test config -c .\winlogbeat.yml -e
```

如果返回 `config OK` 则表示成功。

5. 执行如下命令，启动程序。



```
Start-Service winlogbeat
```

### 上传日志至 CLS

在 C:\\Program Files\\Winlogbeat\\winlogbeat.yml 文件中，将 output.kafka 修改为如下内容，将日志发送到 CLS。



```
output.kafka:
  enabled: true
  hosts: ["${region}-producer.cls.tencentyun.com:9095"] # TODO 服务地址；外网端口9096,
  topic: "${topicID}" # TODO topicID
  version: "0.11.0.2"
  compression: "${compress}" # 配置压缩方式，支持gzip, snappy, lz4；例如"lz4"
  username: "${logsetID}"
  password: "${SecurityId}#${SecurityKey}"
```

参数	说明



链接类型	当前支持 SASL_PLAINTEXT。
hosts	初始连接的集群地址，详细请参见 <a href="#">服务入口</a> 。
topic	配置为日志主题 ID，例如 76c63473-c496-466b-XXXX-XXXXXXXXXXXX。
username	配置为日志集 ID，例如 0f8e4b82-8adb-47b1-XXXX-XXXXXXXXXXXX。
password	格式为 \${SecurityId}#\${SecurityKey}，例如 XXXXXXXXXXXXXXX#YYYYYYYYY。

## 使用 Filebeat 采集 Windows 文件日志

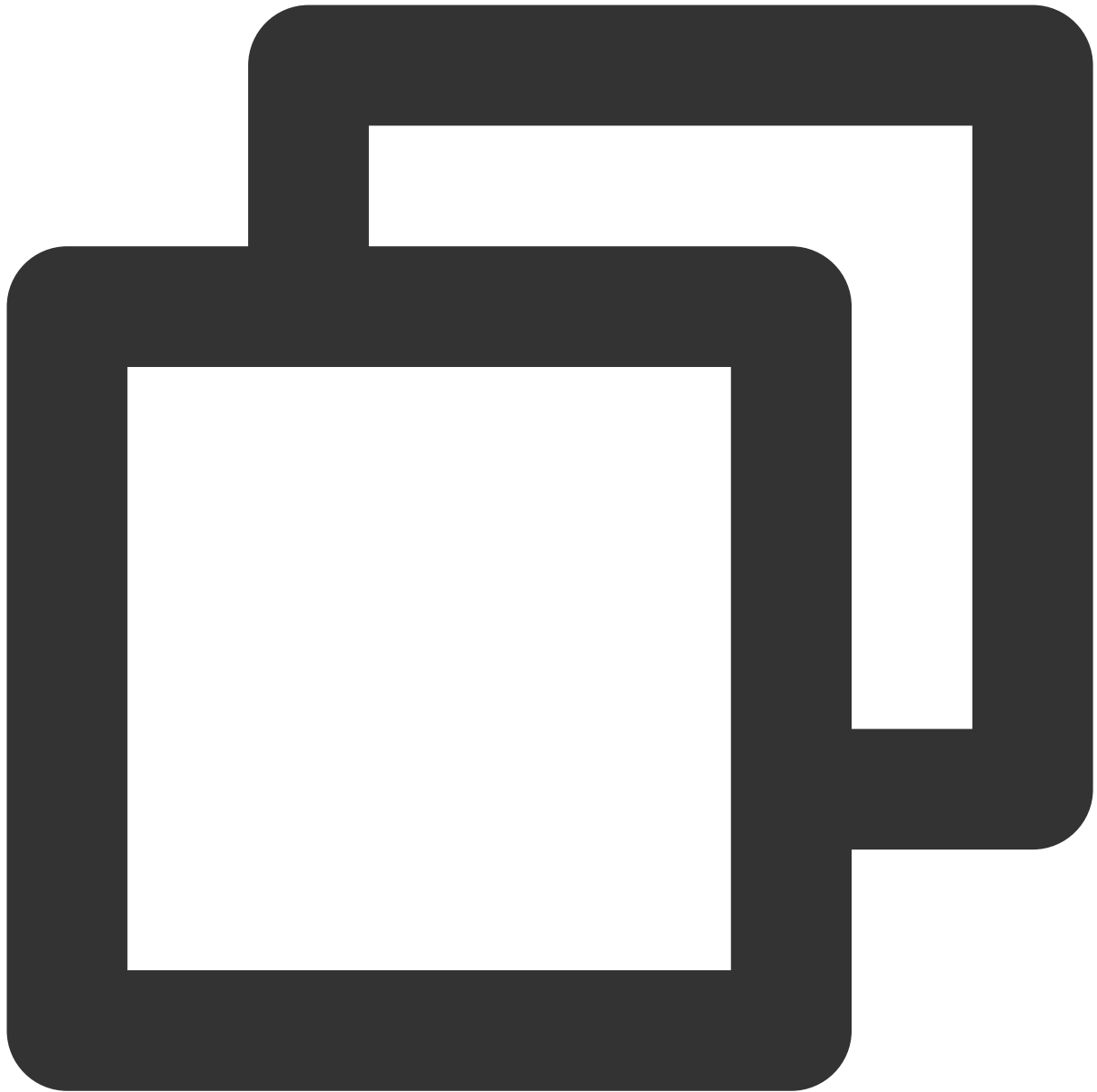
### 安装 Filebeat

1. 前往 [官网](#)，选择对应版本，下载安装包。
2. 将安装包上传至目标 Windows 主机某个盘的根目录，并解压缩。
3. 编辑 filebeat.yml 文件。

#### 说明：

路径符号，要用“/”而不是“\”。

4. 找到目标日志路径，编辑模块配置文件（这里以 mssql 为例）。



```
# Module: mssql
# Docs: https://www.elastic.co/guide/en/beats/filebeat/7.3/filebeat-module-mssql.ht
- module: mssql
# Fileset for native deployment
log:
enabled: true
  # Set custom paths for the log files. If left empty,
  # Filebeat will choose the paths depending on your OS.
var.paths: ["D:/Program Files/Microsoft SQL Server/MSSQL10_50.MSSQLSERVER/MSSQL/Log
```

5. 以管理员身份打开 powershell，并执行如下命令。



```
#进入 filebeat 的路径,具体看你自己放在哪里
```

```
cd c:/filebeat
```

```
#执行安装脚本, 安装 filebeat 服务
```

```
.\install-service-filebeat.ps1
```

```
#启动 mssql 模块
```

```
.\filebeat.exe modules enable mssql
```

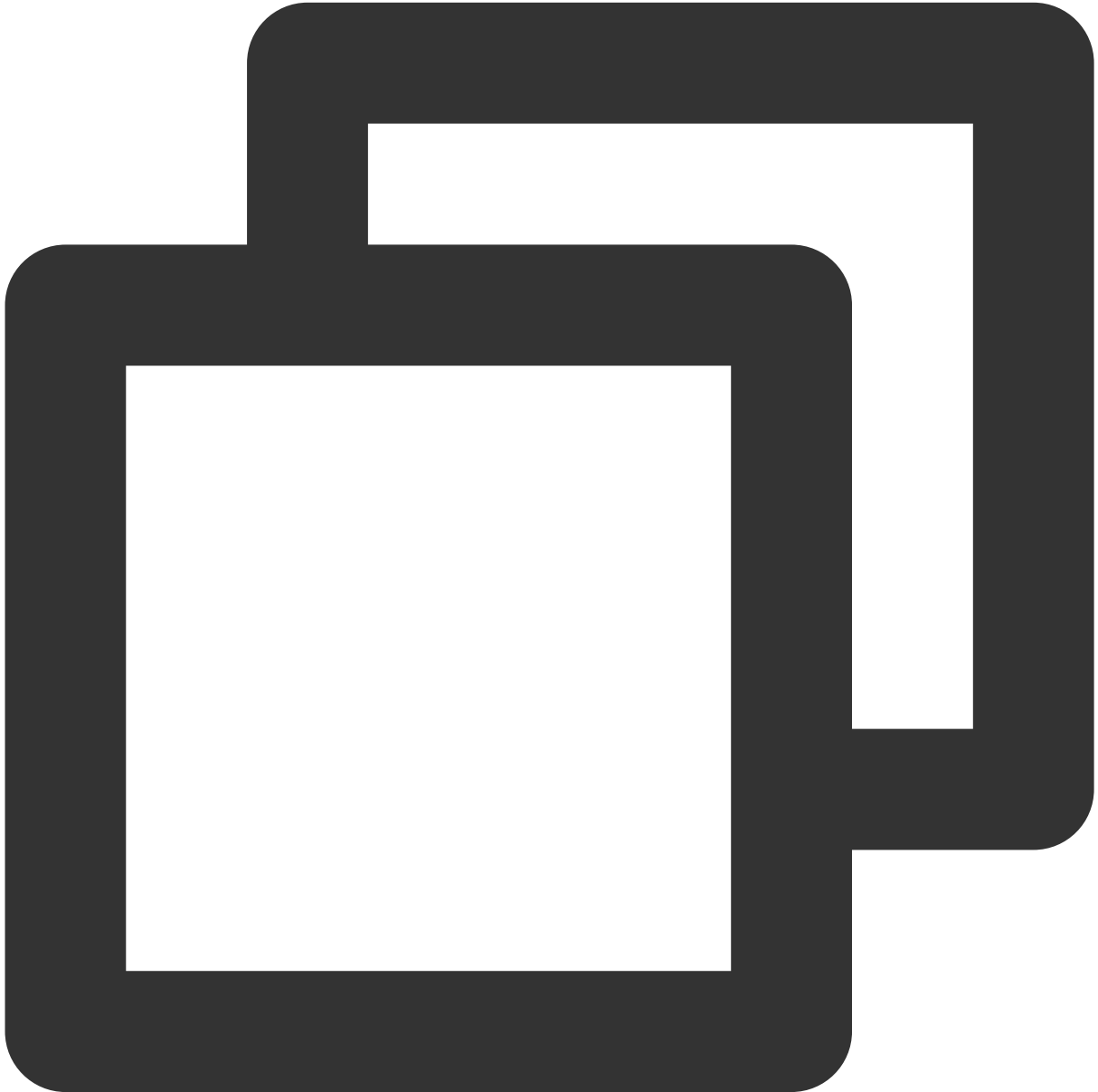
```
#安装模板文件
```

```
.\filebeat.exe setup -e
```

```
#启动 filebeat 服务
start-service filebeat
```

## 上传日志至 CLS

在 filebeat.yml 文件中，将 output.kafka 修改为如下内容，将日志发送到 CLS。



```
output.kafka:
  enabled: true
  hosts: ["${region}-producer.cls.tencentyun.com:9095"] # TODO 服务地址；外网端口9096,
```

```
topic: "${topicID}" # TODO topicID
version: "0.11.0.2"
compression: "${compress}" # 配置压缩方式, 支持gzip, snappy, lz4; 例如"lz4"
username: "${logsetID}"
password: "${SecurityId}#${SecurityKey}"
```

参数	说明
链接类型	当前支持 SASL_PLAINTEXT。
hosts	初始连接的集群地址, 详细请参见 <a href="#">服务入口</a> 。
topic	配置为日志主题 ID, 例如 76c63473-c496-466b-XXXX-XXXXXXXXXXXX。
username	配置为日志集 ID, 例如 0f8e4b82-8adb-47b1-XXXX-XXXXXXXXXXXX。
password	格式为 \${SecurityId}#\${SecurityKey}, 例如 XXXXXXXXXXXXXXXX#YYYYYYYYY。

## 服务入口

地域	网络类型	端口号	服务入口
广州	内网	9095	gz-producer.cls.tencentyun.com:9095
	外网	9096	gz-producer.cls.tencentcs.com:9096

### 注意：

本文档以广州地域为例, 内外网域名需用不同端口标识, 其他地域请替换地址前缀。详情请参考 [可用域名-Kafka上传日志](#)。

# 采集/查询主机文件日志

最近更新时间：2024-01-20 17:28:40

在简单的运维场景下，日志通常先直接输出到服务器本地文件中，再使用 Linux 系统下常用的 `grep` 命令查找符合要求的日志。此方式在业务系统较为复杂时，会由于日志分散在不同的服务器、命令行操作不直观、服务器权限管理限制等原因导致日志查找困难，严重影响运维效率。或者当用户需要基于日志做一些统计分析或监报告警，更是难上加难。

本文将介绍如何快速将通过 `grep` 命令查找的本地日志迁移至日志服务（Cloud Log Service, CLS），以获取如下优势：

数据集中存储及检索，无需登录多台服务器分别进行查询，在负载均衡、微服务等架构下尤为关键。

简单单击即可快速检索日志，告别命令行及繁琐的服务器权限管理。

基于日志进行统计分析，获取关键业务指标，例如 PV、接口响应时间、接口错误率等。

实时检测异常日志，通过短信、邮件和微信等多种方式获取通知。

## 说明：

如果您的日志已经采集到了 CLS，可跳过日志采集和配置索引步骤，直接执行 [步骤3：检索日志](#)。

## 步骤1：日志采集

针对服务器本地日志，可使用 LogListener 将原始日志采集至 CLS，LogListener 安装详情请参见 [LogListener 安装指南](#)。

如果您的服务器为腾讯云云服务器（Cloud Virtual Machine, CVM），还可以通过控制台自动安装 LogListener，详情请参见 [CVM 批量部署 LogListener](#)。

与服务器本地日志不同，为了后续对日志进行更方便的检索和统计分析，在采集时可将非格式化的原始日志转换为格式化的数据。例如原始日志为：



```
10.20.20.10 ::: [Tue Jan 22 14:49:45 CST 2019 +0800] ::: GET /online/sample HTTP/1.
```

可使用分割符 `:::` 切分为八个字段，并为每个字段定义名称：



```
IP: 10.20.20.10
bytes: 35
host: 127.0.0.1
length: 647
referer: http://127.0.0.1/
request: GET /online/sample HTTP/1.1
status: 200
time: [Tue Jan 22 14:49:45 CST 2019 +0800]
```



---

具体操作请参见 [分隔符格式](#)。除了使用分隔符切分日志，CLS 还支持正则、JSON、全文等多种日志切分方式，详情请参见 [采集文本日志](#)。

## 步骤2：配置索引

配置索引的目的在于定义哪些字段需要检索，字段的类型是什么，以便于后续检索日志。对于大多数使用场景，可使用自动配置索引功能，一键完成配置，详情请参见 [配置索引](#)。

## 步骤3：检索日志

本文以常用的 `grep` 命令为例，介绍如何通过 CLS 实现类似的日志检索效果。

示例原始日志为：



```
10.20.20.10 ::: [Tue Jan 22 14:49:45 CST 2019 +0800] ::: GET /online/sample HTTP/1.
```

在 CLS 对应的格式化后日志为：

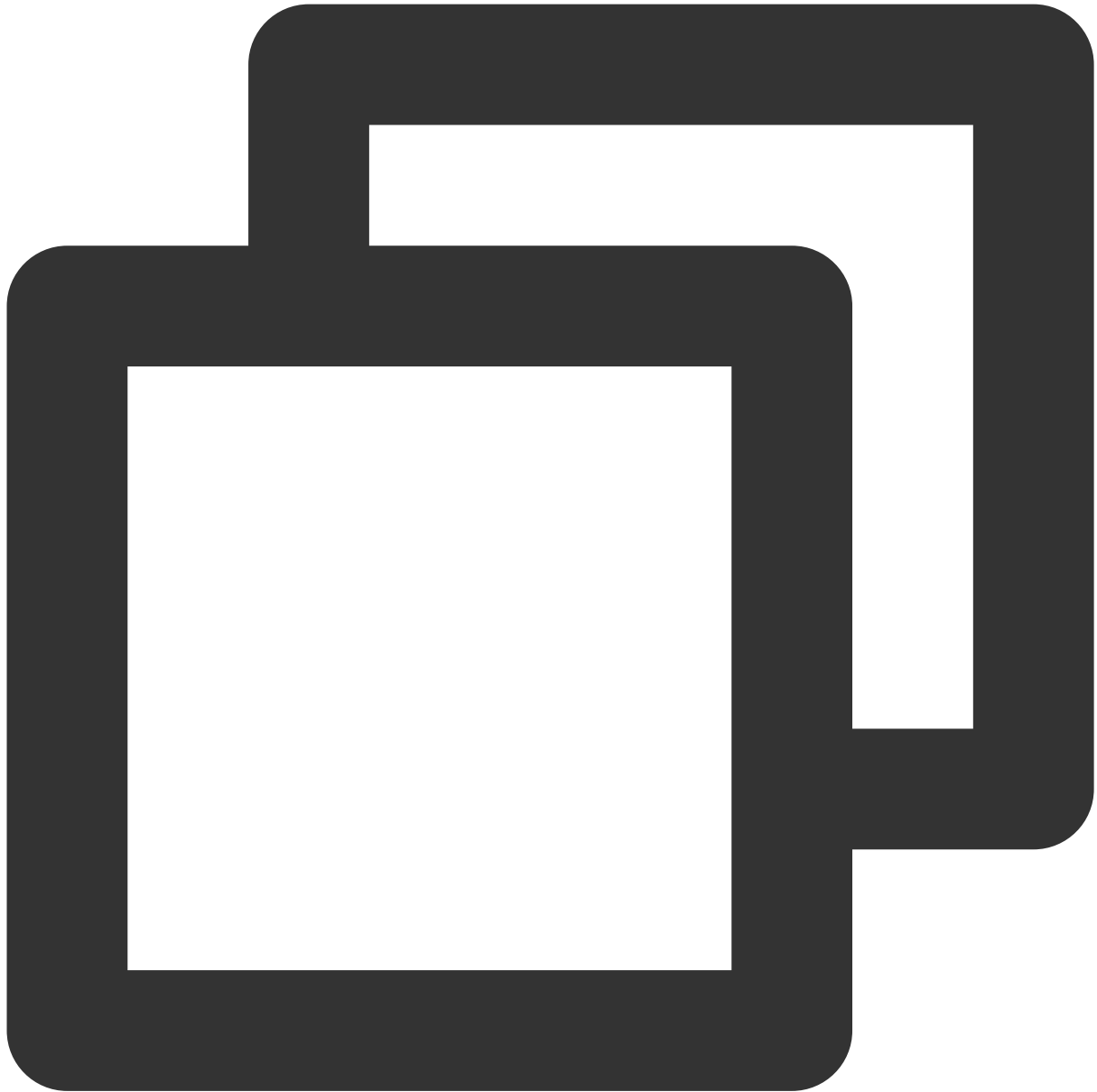


```
IP: 10.20.20.10
bytes: 35
host: 127.0.0.1
length: 647
referer: http://127.0.0.1/
request: GET /online/sample HTTP/1.1
status: 200
time: [Tue Jan 22 14:49:45 CST 2019 +0800]
```

### 案例1

检索 request为/online/sample 的日志。

使用 grep 命令：



```
# grep "/online/sample" test.log
```

使用 CLS 检索方式：



```
request: "/online/sample"
```

## 案例2

检索状态码 `status` 不为200的日志。

使用 `grep` 命令：



```
# grep -v "200" test.log
```

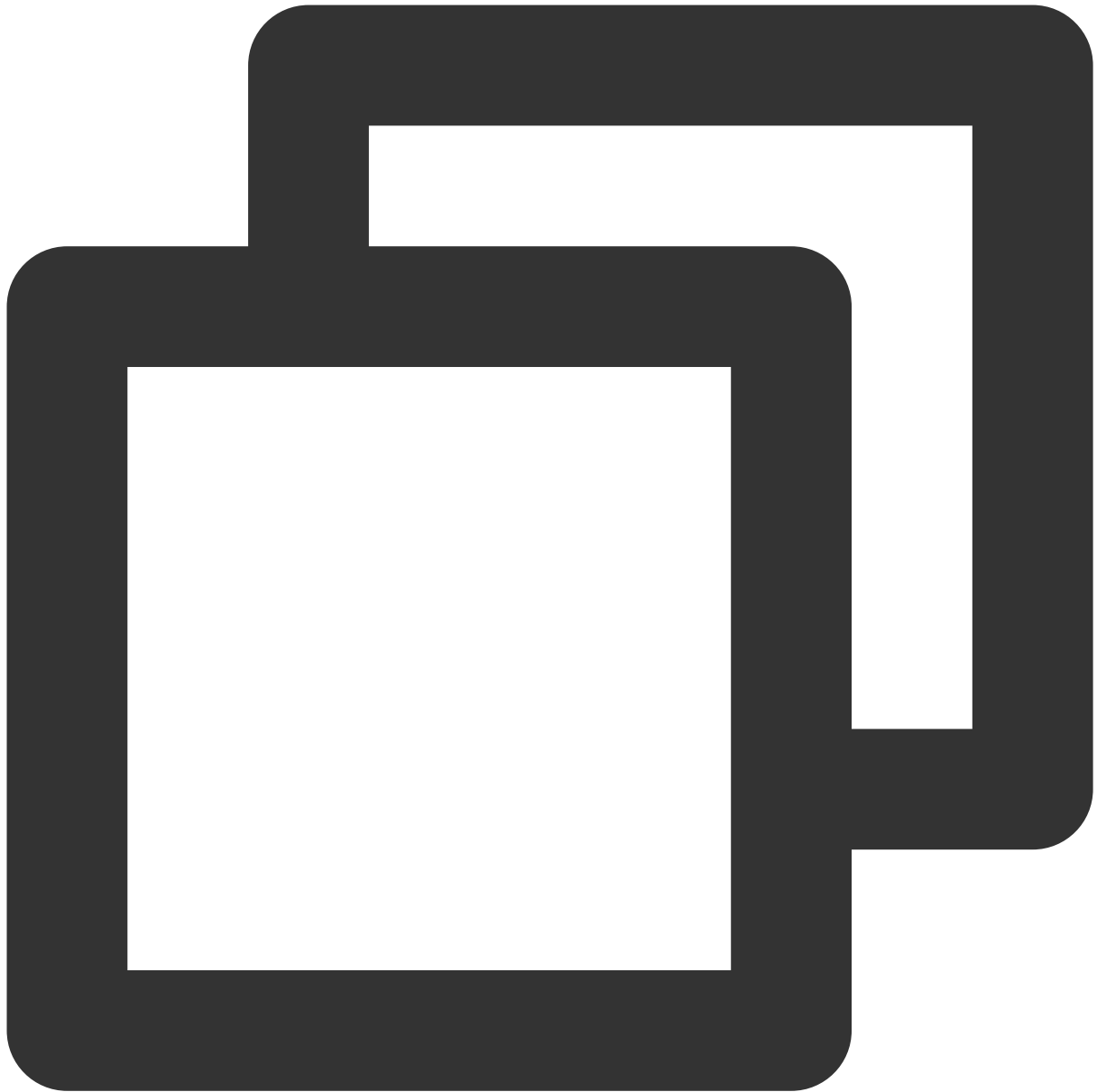
实际上，此方式可能会把一些日志也排除掉（出现了200，但 `status` 不是200的字段）。如需准确检索，则需要编写正则表达式。

使用 CLS 检索方式：



```
NOT status:200
```

CLS 还支持更加灵活的检索方式，例如检索状态码 `status` 大于等于500的日志。



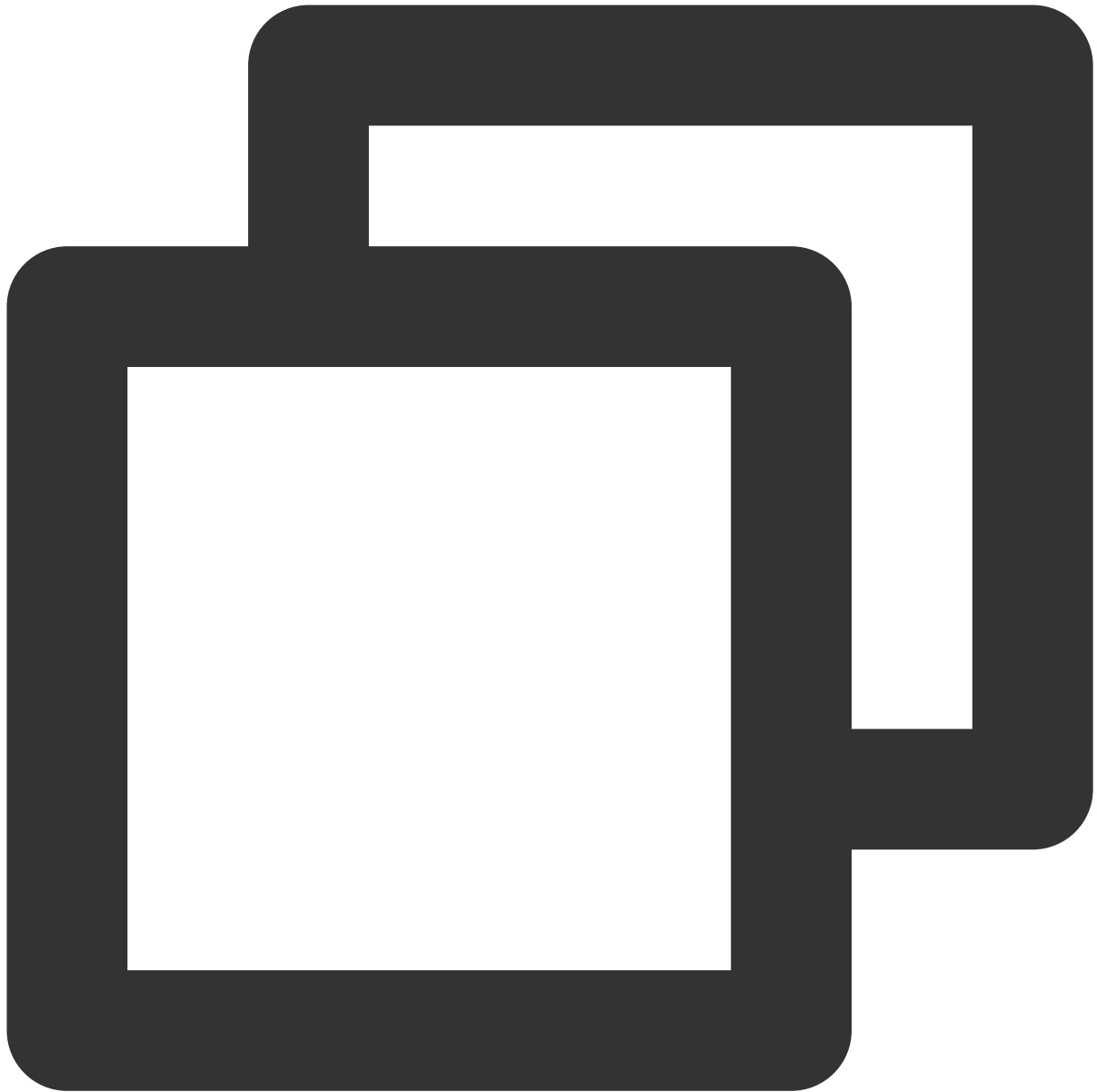
```
status:>=500
```

### 案例3

统计 status 不为200的日志条数。

使用 grep 命令：





```
# grep -c -v "200" test.log
```

使用 CLS 检索方式：



```
NOT status:200 | select count(*) as errorLogCounts
```

#### 案例4

检索状态码 `status` 为200，且 `request` 为 `/online/sample` 的日志。

使用 `grep` 命令：



```
# grep "200" test.log | grep "/online/sample"
```

使用 CLS 检索方式：



```
status:200 AND request:"/online/sample"
```

### 案例5

检索 request 为 /online/sample 或 /offline/sample 的日志。

使用 grep 命令：



```
# grep -E "/online/sample|/offline/sample" test.log
```

使用 CLS 检索方式：



```
request:"/online/sample" OR request:"/offline/sample"
```

### 案例6

检索 request 为 /online/sample，但日志文件不是 test.log 的日志。

使用 grep 命令：



```
# grep -rn "/online/sample" --exclude=test.log
```

使用 CLS 检索方式：



```
request:"/online/sample" AND NOT __FILENAME__:"test.log"
```

### 案例7

检索 time 为 [Tue Jan 22 14:49:45 CST 2019 +0800] 的日志的前10行日志。

使用 grep 命令：





```
# grep "[Tue Jan 22 14:49:45 CST 2019 +0800]" -B 10 test.log
```

使用 CLS 检索方式：



```
time:"[Tue Jan 22 14:49:45 CST 2019 +0800]"
```

检索到匹配的日志后，在控制台单击**上下文检索**，即可查看该条日志附近的日志。

# 检索分析

## CLB 访问日志分析

最近更新时间：2024-01-20 17:28:40

**负载均衡（Cloud Load Balancer, CLB）** 作为千亿 QPS 的网关产品，精细化运营十分重要，而日志服务（Cloud Log Service, CLS）访问日志则是其中的利器。通过 **CLB > 访问日志**，我们可以挖掘海量的数据价值，不仅可以从访问日志中监控客户端请求、辅助排查问题、也可以分析梳理用户行为，为运营角色提供数据支持。本文主要介绍如何使用 CLS 分析 CLB 访问日志。

### 运维监控场景示例

小秦负责某互联网业务广告平台的运维，遇到广告合作方的反馈：用户在平台点击广告时，打开很慢。由于广告合作方对时效性和稳定性要求比较高，提出如果服务出现异常，需要1min内告警，5min内解决。此时，可利用 CLB 日志达到以下能力：

对客户端的访问时延，异常请求监控，高于一定阈值告警。

出现告警，有额外信息帮助判断故障原因。

延时高于阈值的请求都是访问哪些网站，哪些 LB 实例和后端 RS 服务器。

LB 实例和后端 RS 服务器的延时情况统计。

### 运维监控解决方案

基于 CLS 的1min实时告警及多维分析能力，用户可以快速针对 CLB 访问日志进行运维监控，快速定位异常问题修复故障。

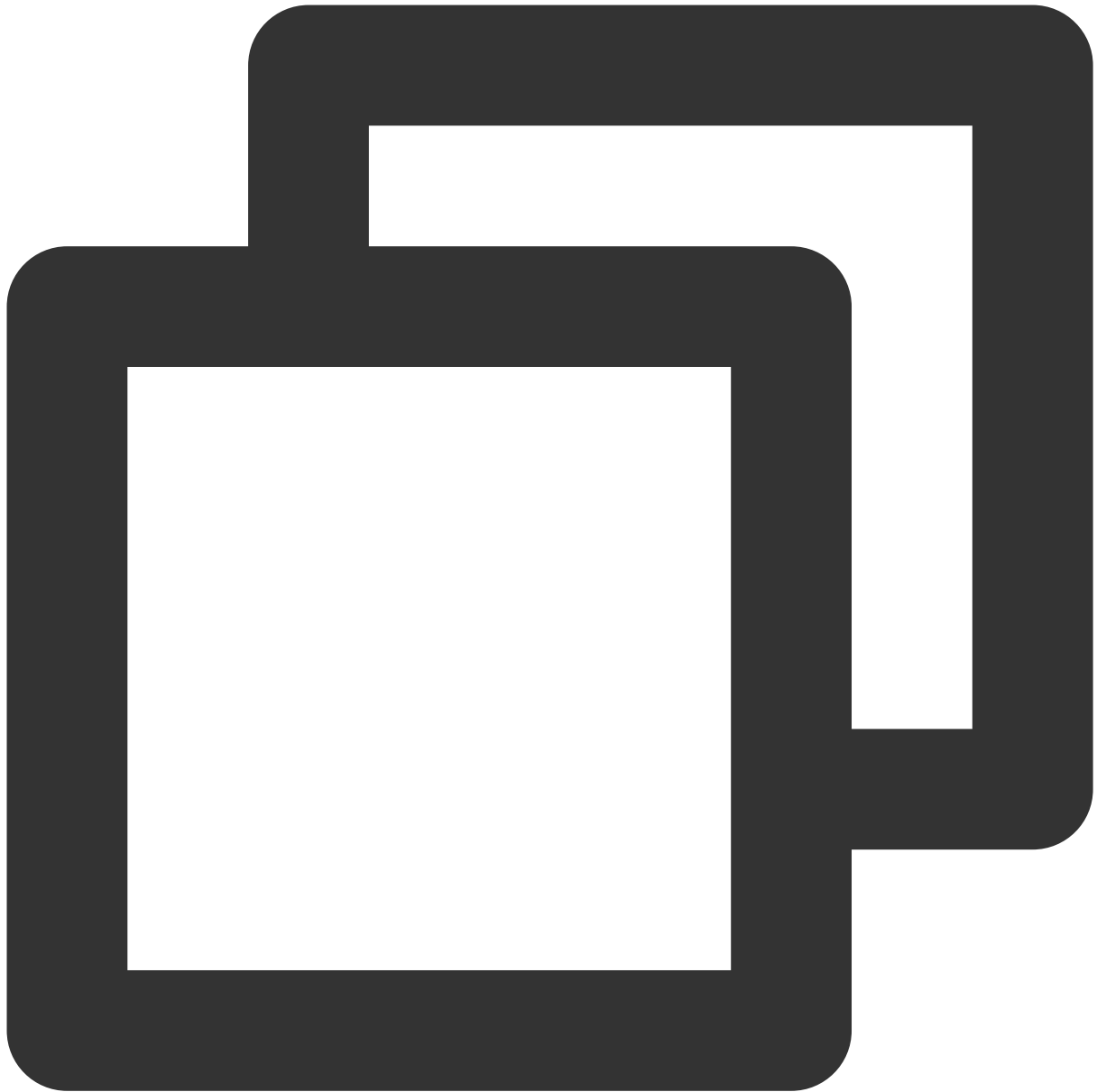
#### 步骤1：开启 CLB 访问日志投递 CLS

1. 登录 [负载均衡控制台](#)。
2. 在左侧导航栏中，选择**实例管理**，进入实例管理页面。
3. 单击**负载均衡 ID/名称**，进入该负载均衡管理页面。
4. 找到并开启**日志服务CLS**。如下图所示：

详情请参考 [配置访问日志](#)。

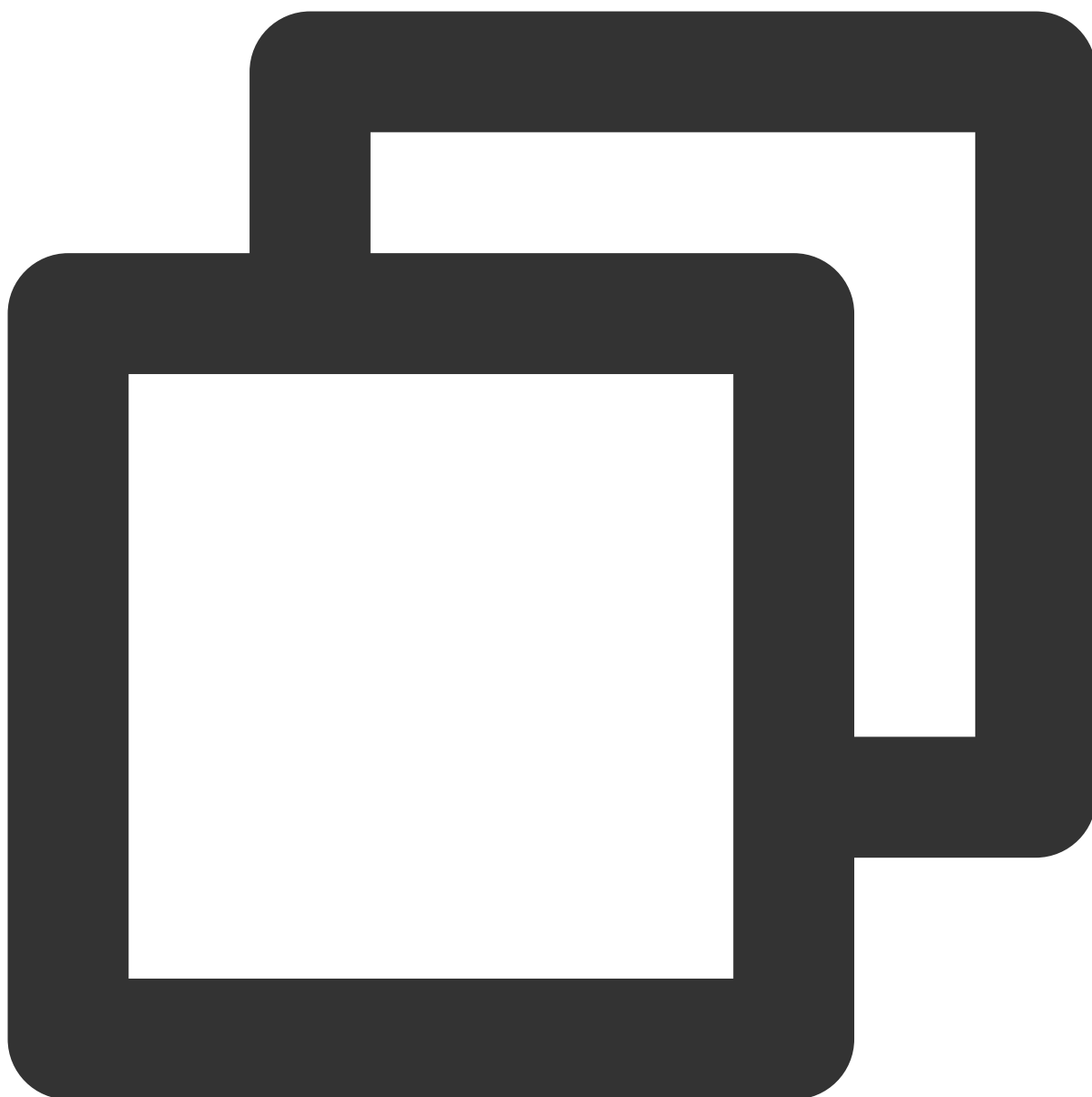
#### 步骤2：配置客户端访问时延及告警策略

客户端访问时延统计：



```
* | select time_series(__TIMESTAMP__, '1m', '%Y-%m-%d %H:%i:%s', '0') as time, rou
```

异常请求统计：



```
status:>200 | select time_series(__TIMESTAMP__, '1m', '%Y-%m-%d %H:%i:%s', '0') as
```

配置告警策略，检测每分钟平均延时，高于阈值告警。

告警策略中配置多维分析，出现告警时，附带额外信息。

LB 实例和后端 RS 服务器的延时情况统计。

延时高于阈值的请求都是访问哪些网站，哪些 LB 实例和后端 RS 服务器。

配置通知渠道，可支持如下渠道：

邮件

短信

微信  
企业微信  
电话  
自定义接口回调

### 步骤3：接收告警和快速定位

一旦触发告警，微信、企业微信、短信、电话等多端接收告警信息及详情内容：

告警详情中，可以看到受影响的 RS 实例，LB 实例等信息。

由此可得知，LB 实例平均延时较高，受影响的 LB 实例主要是 `9.*****.1`。小秦即可快速定位出异常 LB 实例并修复，整体耗时仅用1分钟。

## 运营统计场景示例

某科技内容 App 希望下个月策划一次线下沙龙会，一方面增加存量用户的粘性，另一方面借此机会宣传产品，拉动新用户。由于准备时间较短，经费有限，为了完成 KPI 目标，运营小婷列了如下需要获知的信息。

线下沙龙在哪里举办：需要了解访问客户地理来源，了解重点客户群体地理位置。

沙龙主题是什么：统计热点网站 TOP 排序，了解用户关注较多的内容版块是哪些。

当前用户主要用哪些客户端访问：针对当前客户端分布，重点设计落地页。

宣传落地页投放在哪些渠道：统计当前网站请求来源，寻找流量高的导流入口重点投放广告。

## 运营统计解决方案

使用 CLB 访问日志，用户可以轻松解决运营统计问题。

### 步骤1：了解访问客户地理来源

利用 CLS 提供 IP 函数，将客户端 IP 转换为对应的省份或国家。

中国分布：



```
* | select count(1) as c, ip_to_province(remote_addr) as address group by address l
```

全球分布：



```
* | select count(1) as c, ip_to_country(remote_addr) as address group by address li
```

## 步骤2：统计热点网站 TOP 排序

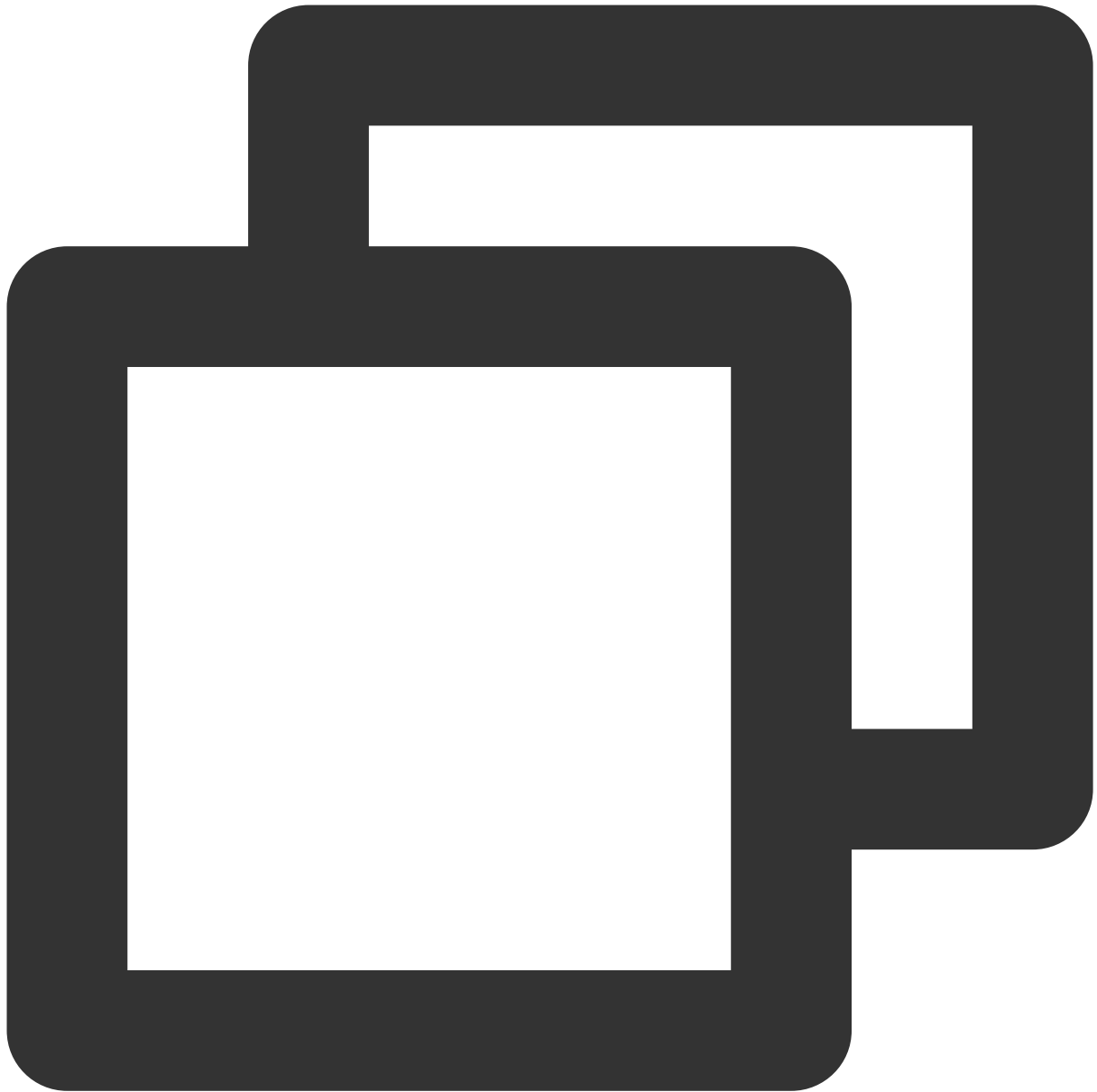
http\_host 记录了访问的请求域名，通过统计请求域名的 pv, uv, 可以统计 top host 排序。





```
* | select http_host, count(*) as pv, approx_distinct(remote_addr) as uv group by h
```

### 步骤3：统计客户端分布情况



```
* | select http_user_agent, count(*) group by http_user_agent
```

#### 步骤4：统计当前网站请求来源

http\_referer 字段记录了网站的请求来源。



```
* | select http_referer, count(*) as count group by http_referer order by count des
```

## 总结

CLB 的访问日志挖掘出很多价值，例如 pv、uv 趋势统计，客户端报文流量统计，状态码分布，P99，P95 访问延时等。为了帮助用户快速分析 CLB 访问日志，CLS 和 CLB 联合打造了开箱即用的可视化分析方案，用户只需开启 CLB 访问日志投递 CLS，即可马上享用。

# Nginx 访问日志分析

最近更新时间：2024-01-20 17:28:40

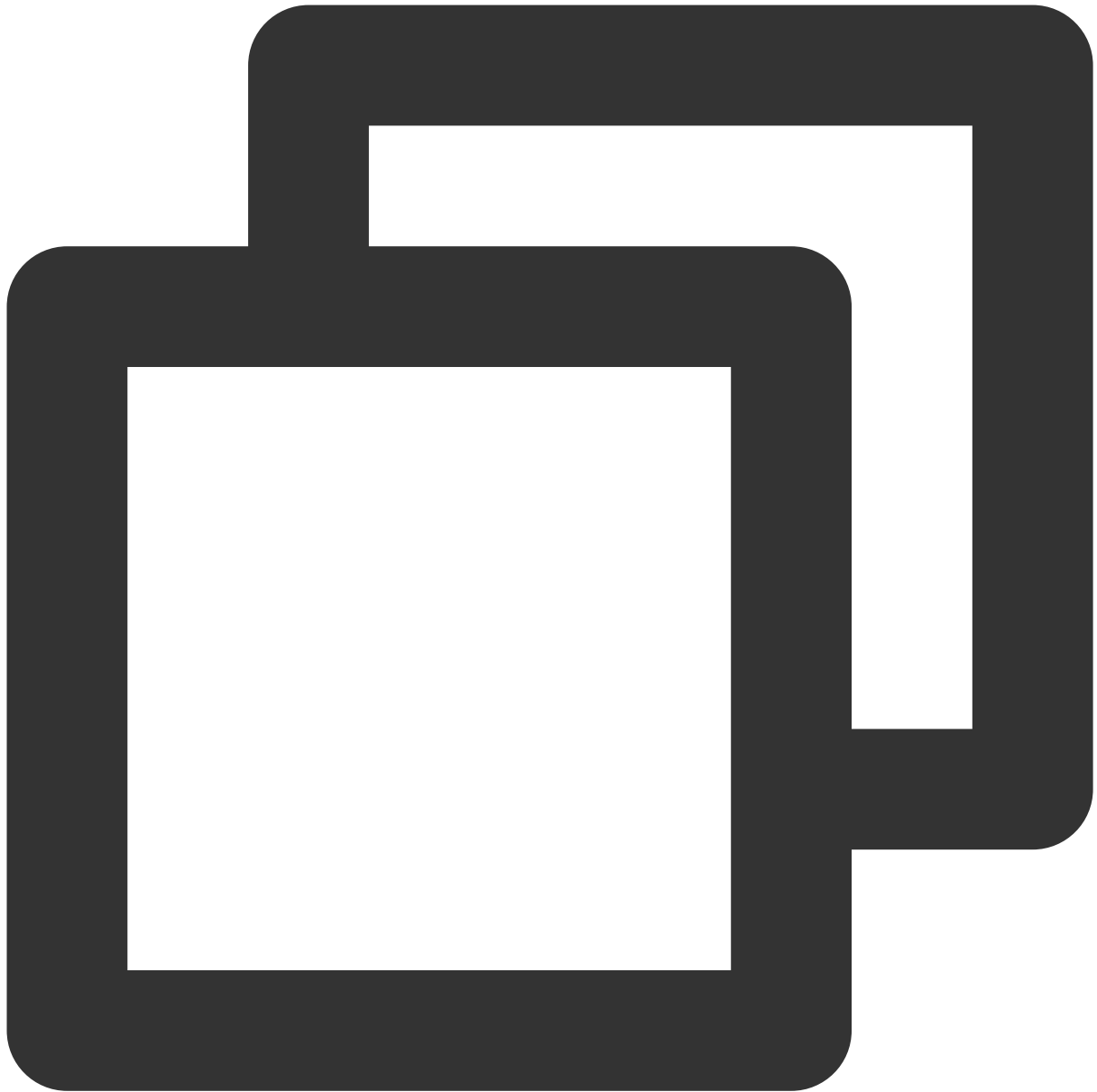
## 概述

Nginx 是一个高性能的 HTTP 和反向代理 Web 服务器，透过 Nginx 日志可以挖掘非常大的价值，例如诊断调优网站，监控网站稳定性，运营数据统计等。本文介绍如何通过日志服务（Cloud Log Service, CLS）对 Nginx 进行全方位日志数据挖掘。

## 前提条件

已将 Nginx 日志采集至 CLS，详见 [操作指南](#)。

本文采取标准 Nginx 日志配置：



```
log_format main '$server_name $remote_addr - $remote_user [$time_local] "$request
                '$status $upstream_status $body_bytes_sent "$http_referer" '
                '$http_user_agent' "$http_x_forwarded_for" ';
```

## 场景示例

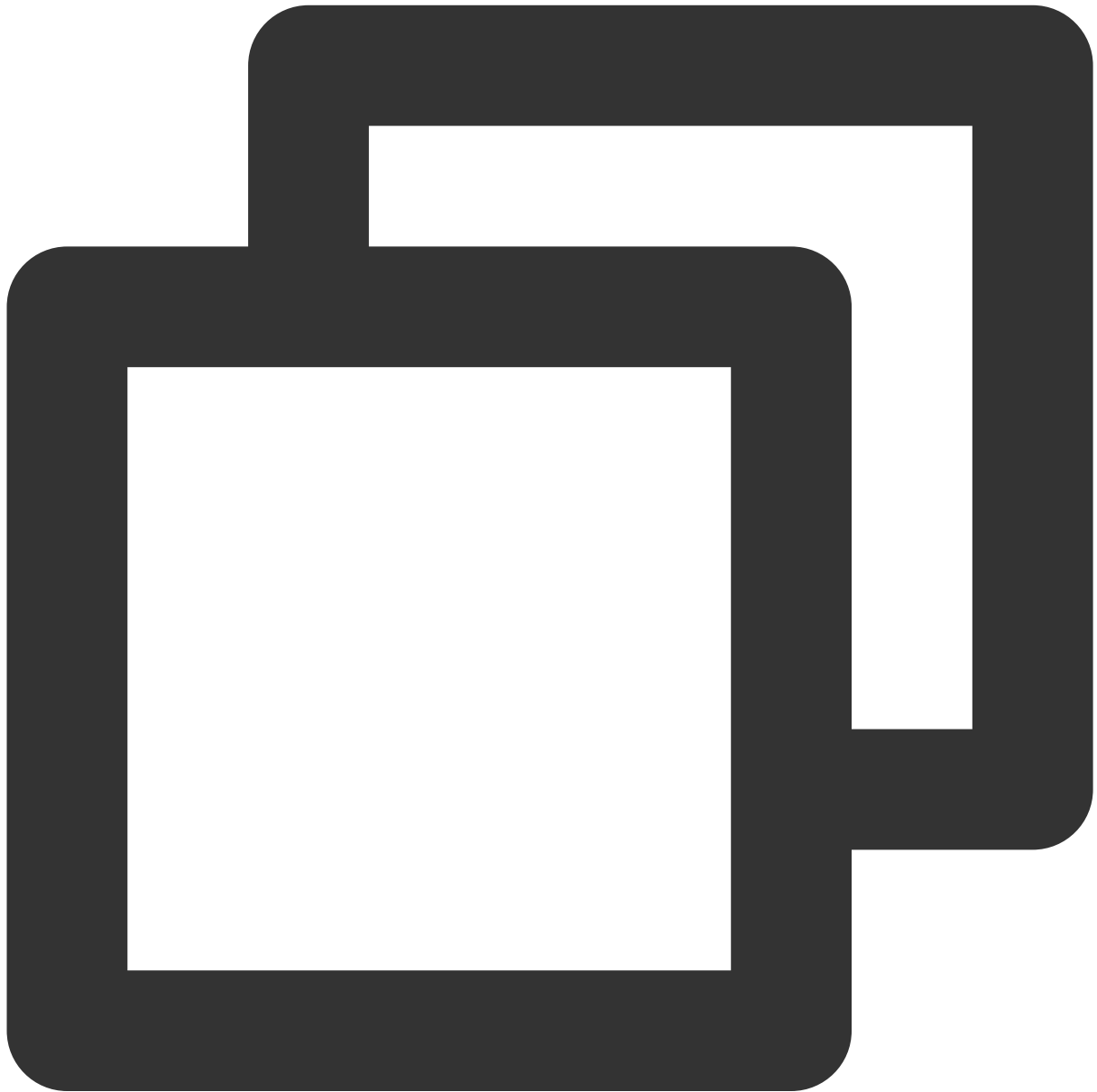
### 诊断调优

## 需求场景

针对访问延时大的页面进行调优，优化用户体验。

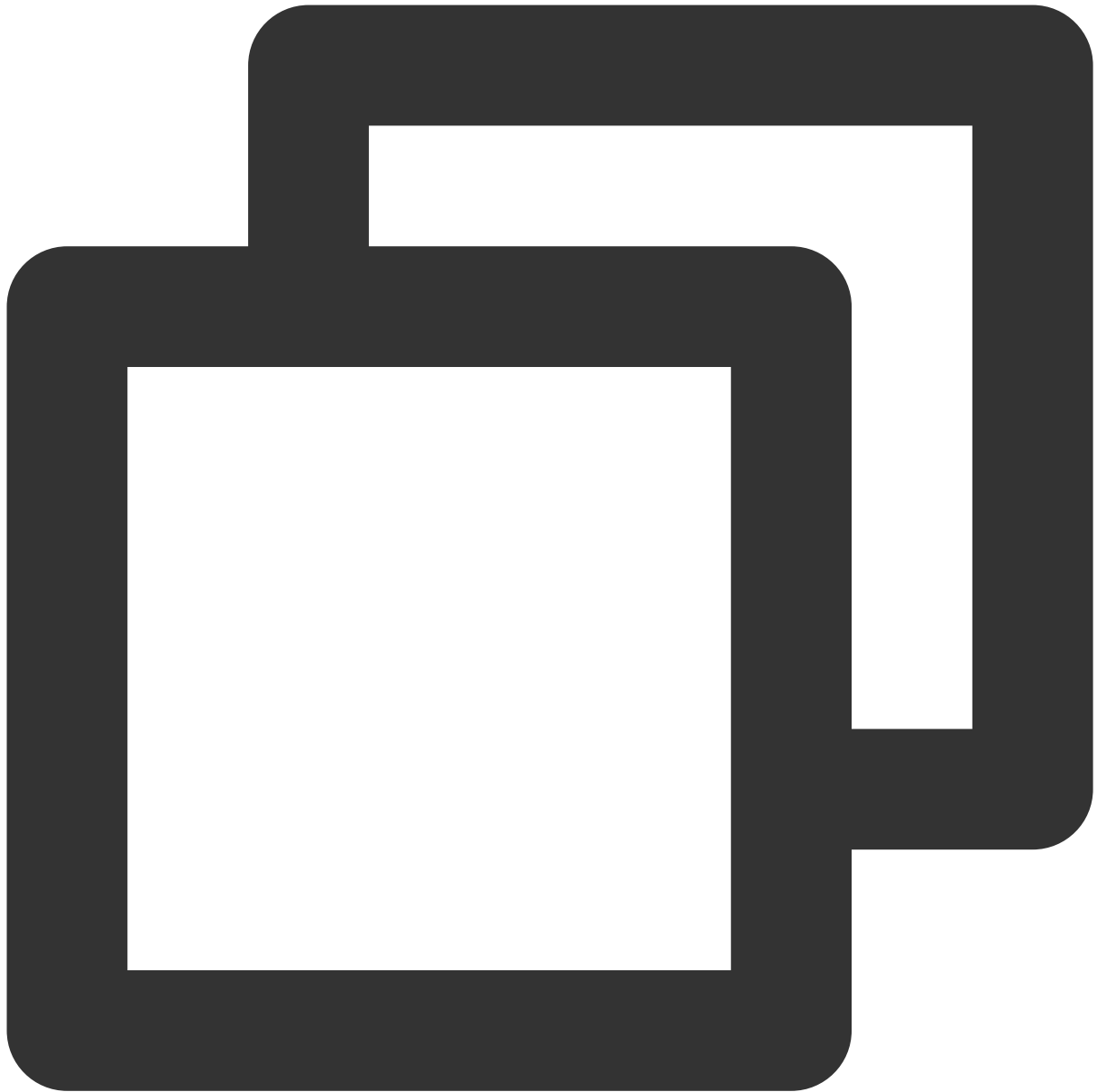
## 解决方案

计算每5分钟请求的平均延时和最大延时，从整体了解延时情况。



```
* | select time_series(__TIMESTAMP__, '5m', '%Y-%m-%d %H:%i:%s', '0') as time, avg
```

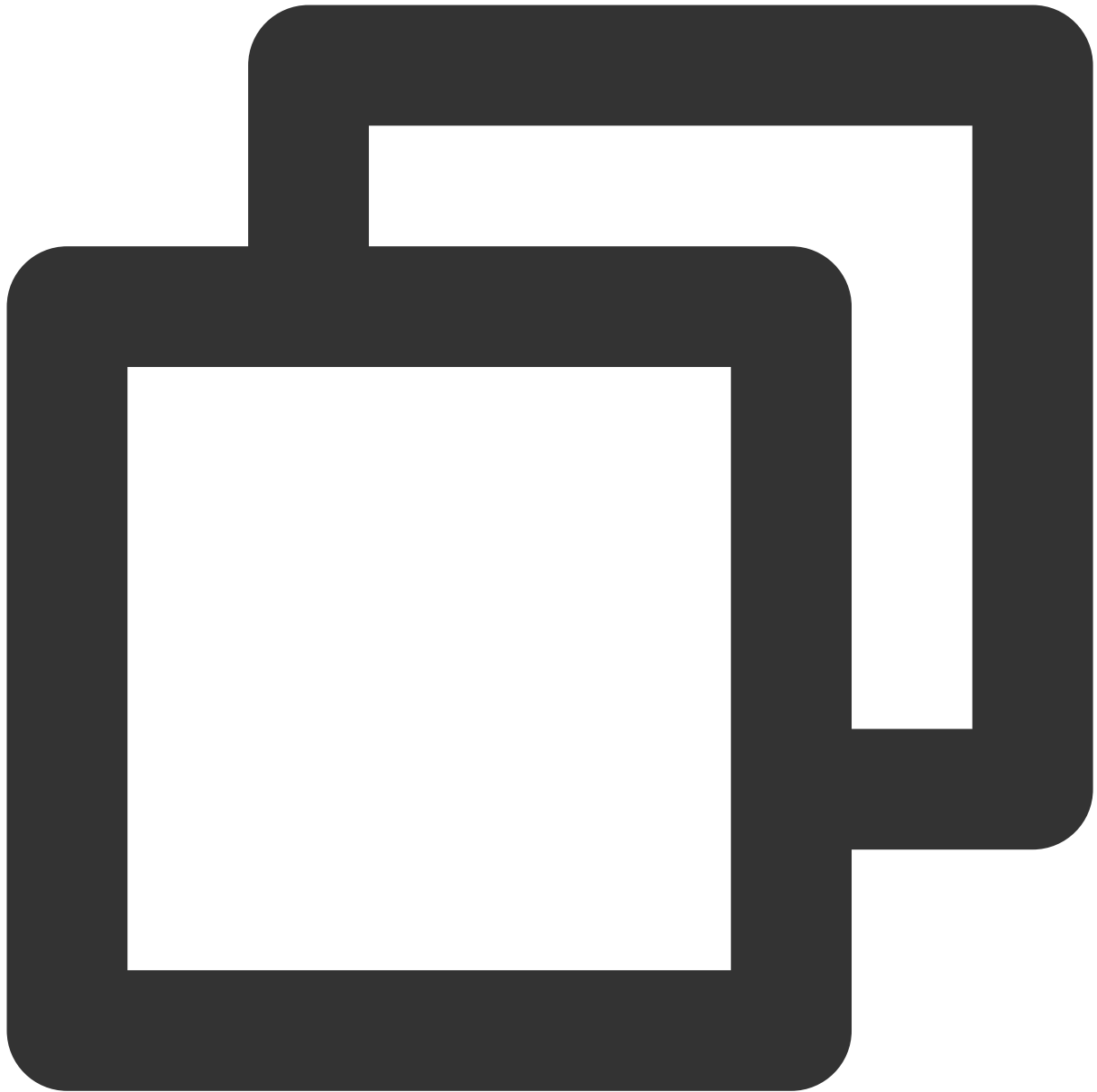
统计最大延时对应的请求页面，进一步优化页面响应。



```
* | select time_series(__TIMESTAMP__, '5m', '%Y-%m-%d %H:%i:%s', '0') as time, max
```

对延时最大的页面进行调优。

例如 `/4nm8c.html` 页面的访问延时最大，需要对 `/4nm8c.html` 页面进行调优，则需计算 `/4nm8c.html` 页面的访问 PV、UV、各种请求方法次数、各种请求状态次数、各种浏览器次数、平均延时和最大延时。



```
request_uri:"/4nm8c.html*" | select count(1) as pv,  
    approx_distinct(remote_addr) as uv,  
    histogram(method) as method_pv,  
    histogram(status) as status_pv,  
    histogram(user_agent) as user_agent_pv,  
    avg(request_time) as avg_latency,  
    max(request_time) as max_latency
```

## 监控网站稳定性问题

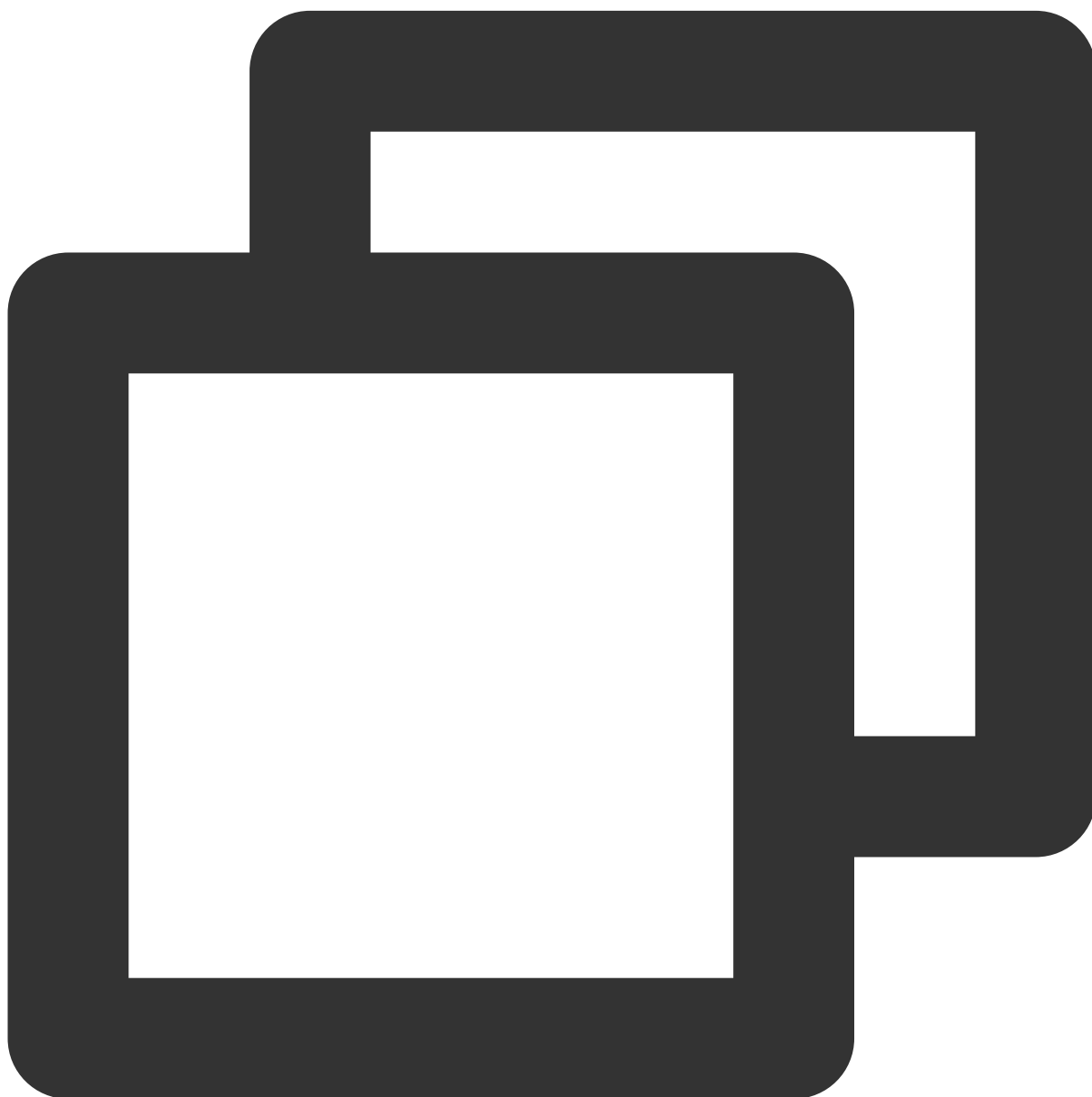


## 需求场景

针对性能问题、网站错误、流量急跌或暴涨等情况，根据日志监控阈值，一旦触发阈值告警，先于用户发现问题。

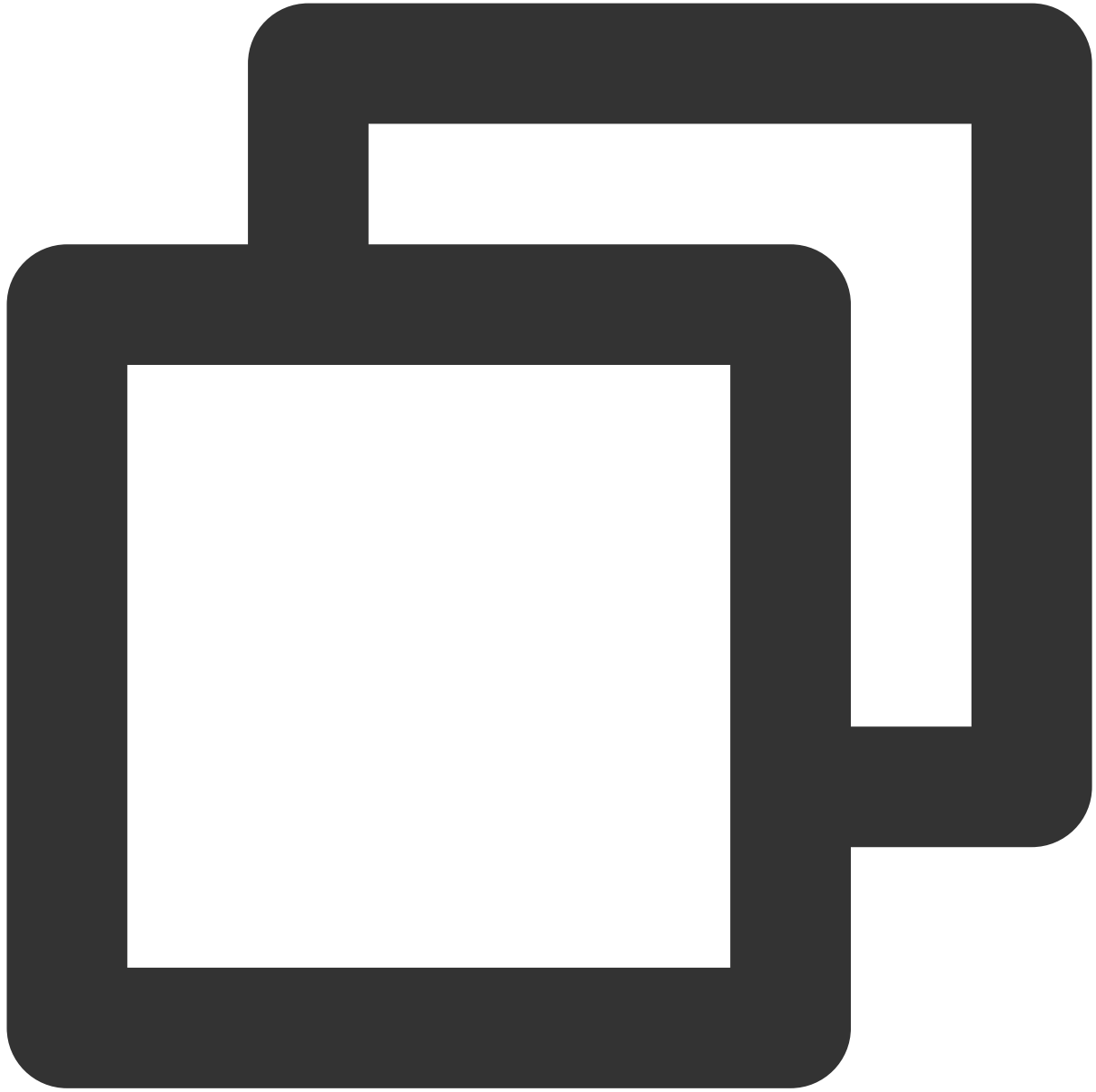
## 解决方案

使用数学统计中的百分数（例如99%最大延时）来作为告警触发条件较为准确，使用平均值，个体值触发告警会造成一些个体请求延时被平均，无法反映真实情况。例如使用如下查询分析语句计算一天窗口（1440分钟）内各分钟的平均延时大小、50%分位的延时大小和90%分位的延时大小。



```
* | select avg(request_time) as l, approx_percentile(request_time, 0.5) as p50, app
```

针对99%分位的延时大于100ms告警，并且在告警信息中直接展示受影响的 url，用户，快速判断出错情况。



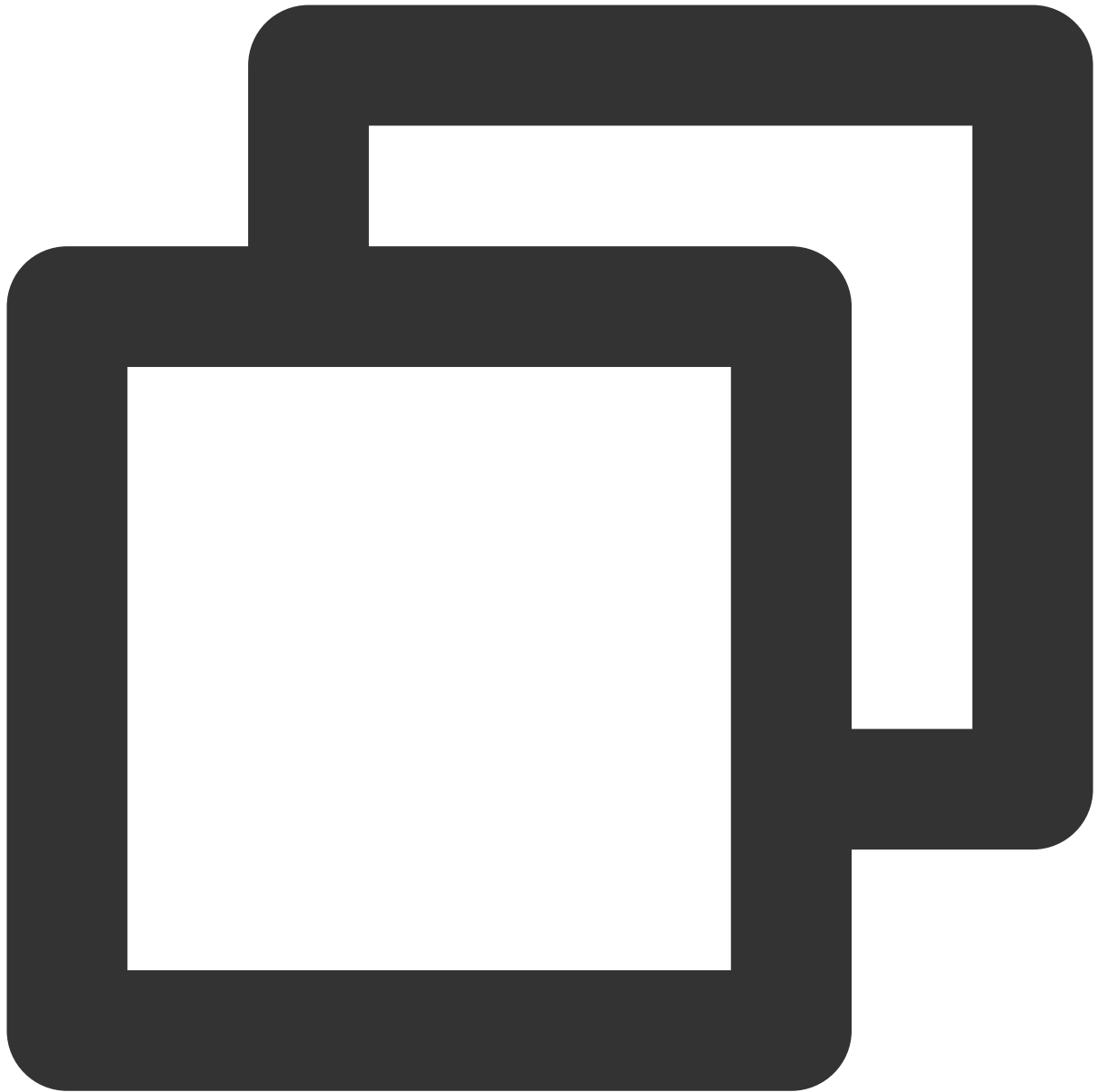
```
* | select approx_percentile(request_time, 0.99) as p99
```

接收告警信息，根据受影响的 top url 及用户信息，进行针对性的告警恢复决策。

## 分析网站访问情况

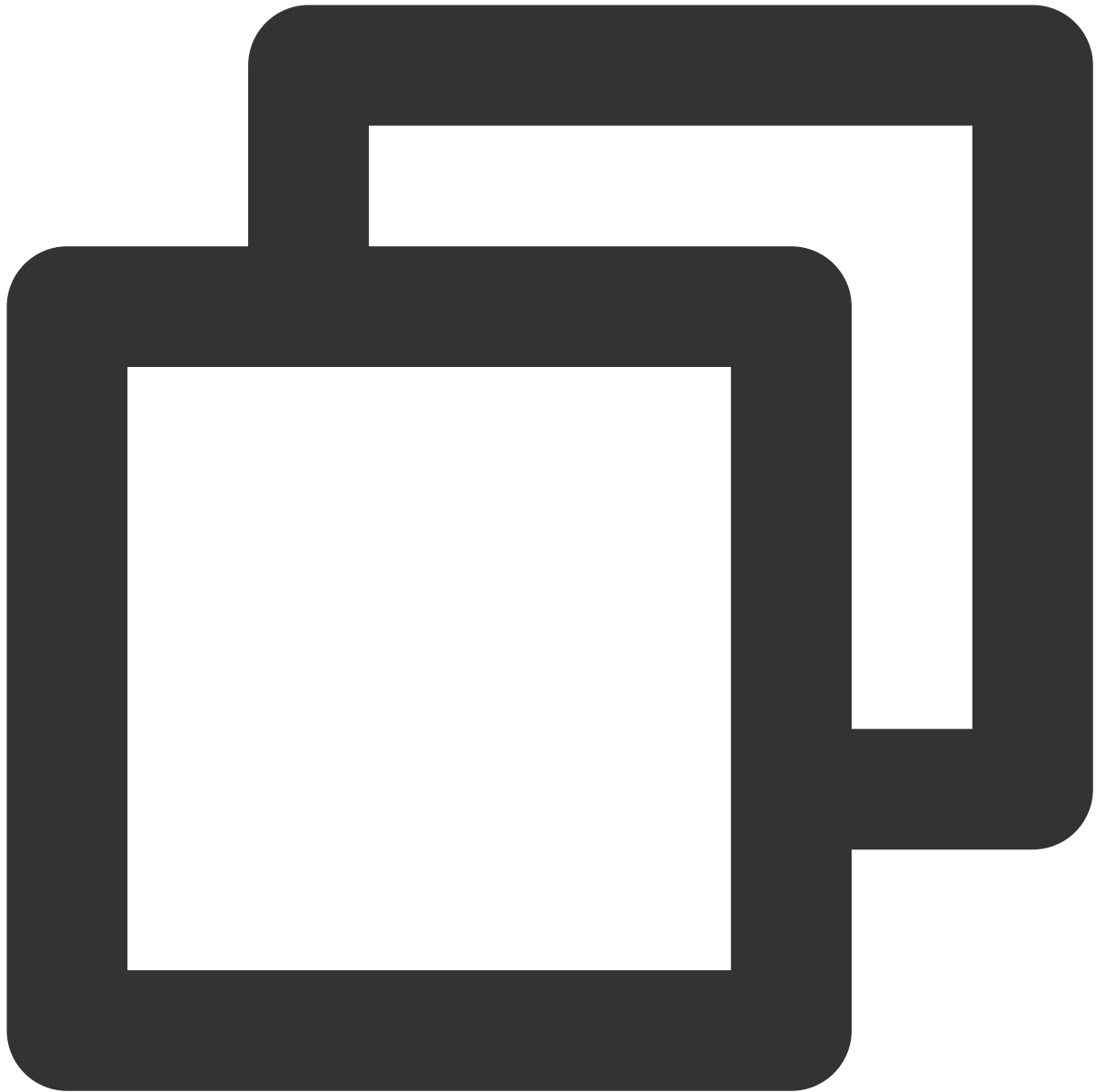
利用 CLS，用户可以搭建运营数据大盘，全方位展示网站访问情况。访问 PV/UV 统计、访问地理信息统计，前十访问来源、访问前十地址和等信息均可快速分析。

统计最近一天访问 IP 地址的来源情况。



```
* | select count(1) as c, ip_to_province(remote_addr) as address group by address l
```

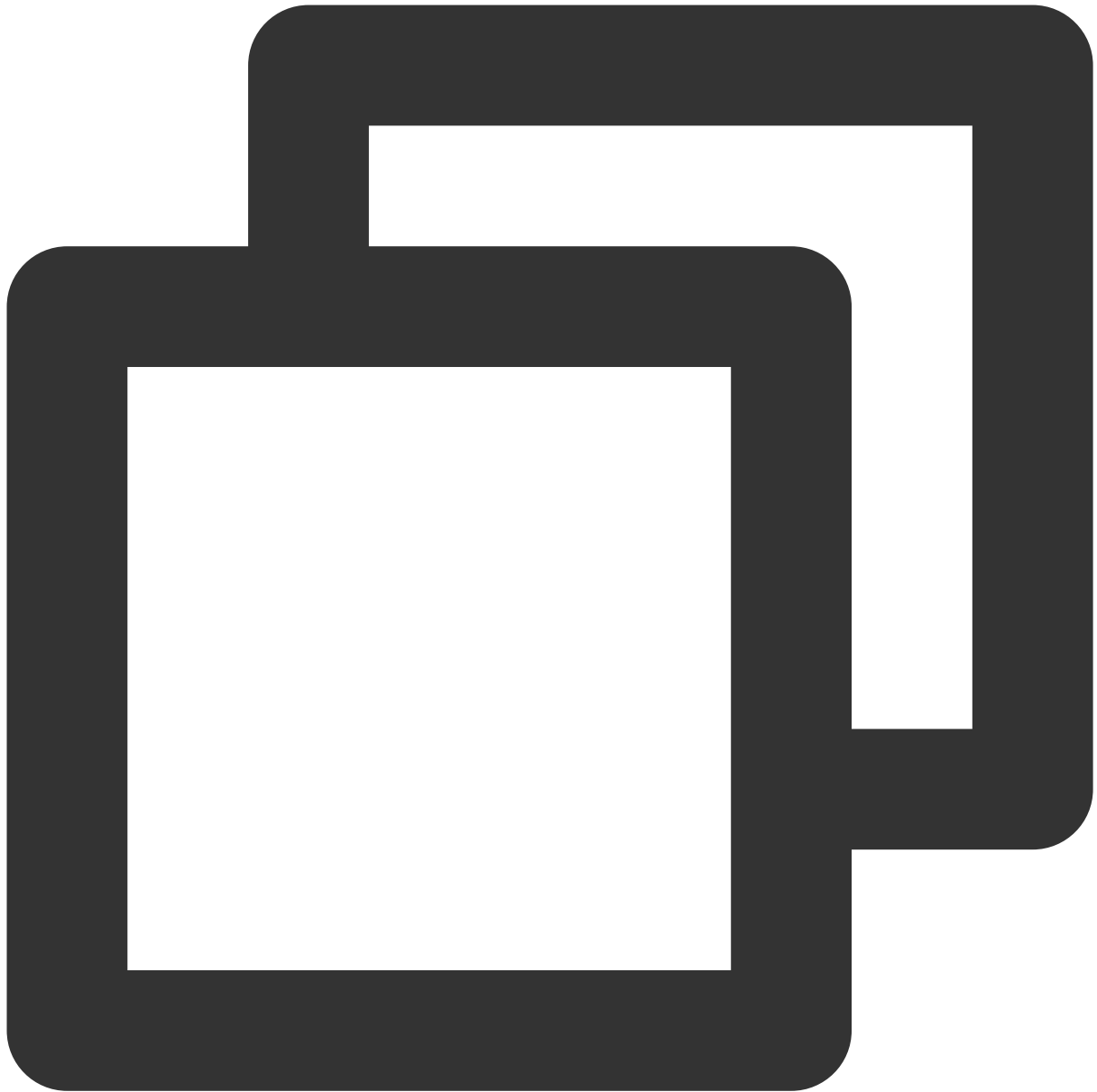
展示最近一天 PV 数最多的前十个访问来源页面，获取热门页面。



```
* | select count(1) as pv , http_referer group by http_referer order by pv desc li
```

pv ↑	http_referer ↓
13	http://01omc.com/v_19rrknzf8g.html?vfm=m
13	http://02adv.com/lc03_p/201509/29/17/19/
13	http://0e3cs.com/q595k1o5z1jn/article/14
13	http://02bal.com/ui/swf/player/v0929/mai
13	http://0kipu.com/ads/index.htm?v=3.63
19	http://jwf.cc:8085/wap/wifi/wait
32	http://jwf.cc:8085/wap/wifi/confirm
51	http://www.guilin.cm/

展示最近一天内的 PV 数和 UV 数。



```
*| select approx_distinct(remote_addr) as uv ,count(1) as pv , time_series(__TIMEST
```

# CDN 访问日志分析

最近更新时间：2024-01-20 17:28:40

## 概述

内容分发网络（Content Delivery Network, CDN）是非常重要的互联网基础设施，用户可以通过 CDN，快速的访问网络中各种图片，视频等资源。在访问过程中，CDN 会产生大量的日志数据，通过对 CDN 访问日志的分析，用户可以挖掘出大量有用的信息用于 CDN 质量和性能的分析，错误诊断，客户端分布，以及用户行为分析。

## 前提条件

已将 CDN 日志采集至日志服务（Cloud Log Service, CLS），详见 [操作详情](#)。

## 场景示例

### 传统 CDN 日志分析

当前，各 CDN 服务提供商，通常会实时提供基础的监控指标，例如请求次数，宽带等信息。但是，在许多特定的分析场景下，这些默认的实时指标可能并不能满足用户定制化的分析需求。因此，通常用户会进一步将 CDN 的原始日志下载下来，进行离线的深入分析与挖掘。

这种情况用户自行搭建离线分析集群，不仅需要大量的运维开发成本和人力成本；同时在一些基于 CDN 日志的告警，排障等分析场景下，离线日志的实时性难以保证。

### CDN to CLS 方案

腾讯云 CDN 与 CLS 实现打通，用户可以将 CDN 的数据实时投递至 CLS，并进一步使用 CLS 的检索和 SQL 分析能力，来满足不同场景下用户个性化的实时日志分析需求：

日志一键投递

百亿级日志，秒级分析

Dashboard 仪表盘实时日志可视化

一分钟实时告警

### CDN 日志介绍

CDN 日志字段说明：

字段名	原始日志类型	日志服务类型	说明

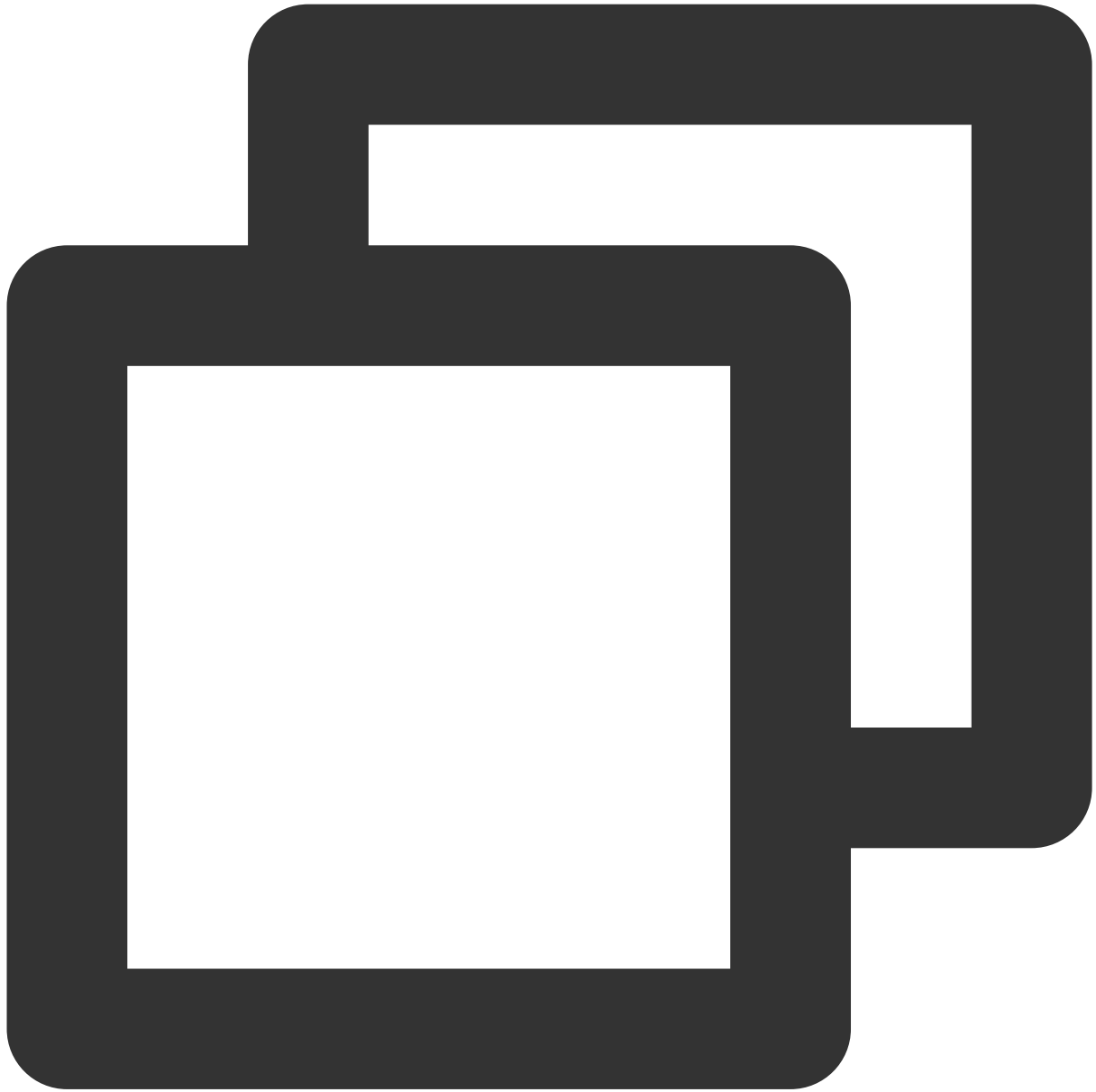
app_id	Integer	long	腾讯云账号 APPID
client_ip	String	text	客户端 IP
file_size	Integer	long	文件大小
hit	String	text	缓存 HIT / MISS, 在 CDN 边缘节点命中、父节点命中均标记为 HIT
host	String	text	域名
http_code	Integer	long	HTTP 状态码
isp	String	text	运营商
method	String	text	HTTP Method
param	String	text	URL 携带的参数
proto	String	text	HTTP 协议标识
prov	String	text	运营商省份
referer	String	text	Referer 信息, HTTP 来源地址
request_range	String	text	Range 参数, 请求范围
request_time	Integer	long	响应时间 (毫秒), 指节点从收到请求后响应所有回包再到客户端所花费的时间
request_port	String	long	客户端与 CDN 节点建立连接的端口。若无, 则为 -
rsp_size	Integer	long	返回字节数
time	Integer	long	请求时间, UNIX 时间戳, 单位为: 秒
ua	String	text	User-Agent 信息
url	String	text	请求路径
uuid	String	text	请求的唯一标识
version	Integer	long	CDN 实时日志版本

## CDN 质量监控

场景1：监控 CDN 访问延时高于一定阈值告警



使用数学统计中的百分数（例如99%最大延迟）来作为告警触发条件较为准确，使用平均值，个体值触发告警会造成一些个体请求延时被平均，无法反映真实情况。例如使用如下查询分析语句计算一天窗口（1440分钟）内各分钟的平均延时大小，50%分位的延时大小，和90%分位的延时大小。



```
* | select avg(request_time) as l, approx_percentile(request_time, 0.5) as p50, app
```

针对99%的延时大于100ms进行告警，并且在告警信息中展示受影响域名、url、client\_ip，以便快速判断错误情况。告警设置如下语句。



```
* | select approx_percentile(request_time, 0.99) as p99
```

通过配置多维度分析，在告警信息中展示受影响的域名，客户端 IP，url，帮助开发人员快速定位问题。一旦告警触发后，通过微信，企业微信，短信第一时间获取关键信息。

**场景2: 资源访问错误激增告警，当同比增数超过一定阈值时，告警通知用户**

当页面访问错误的数量出现激增时，可能说明 CDN 后端服务器出现故障，或者请求过载。我们可以通过设置告警来对一定时间范围内（eg.一分钟）请求错误数量的同比增数进行监控，当同比增数超过一定阈值时，告警通知用户。

**最近一分钟内的错误数量**



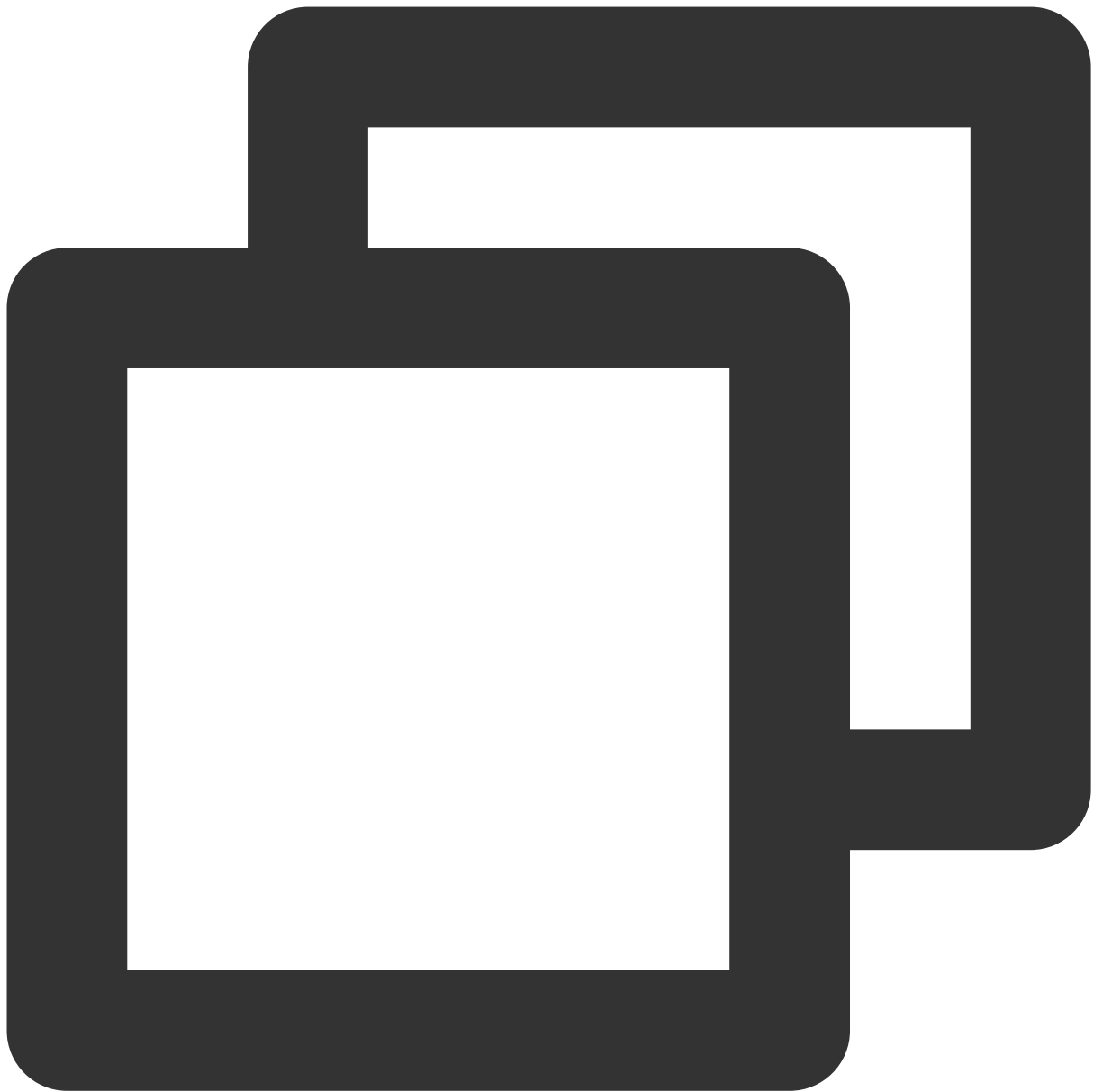
```
* | select * from (select * from (select * from (select date_trunc('minute', __TIME
```

上一分钟内的错误数量



```
* | select * from (select * from (select * from (select date_trunc('minute', __TIM
```

告警策略配置触发条件为【最近一分钟内的错误数量】 - 【上一分钟内的错误数量】 > 指定阈值



```
$2.errrct-$1.errrct >100
```

### CDN 质量和性能分析

CDN 提供日志中，包含了丰富的内容，我们可以从多个维度对 CDN 的整体质量和性能进行全方位的统计和分析。

健康度

缓存命中率

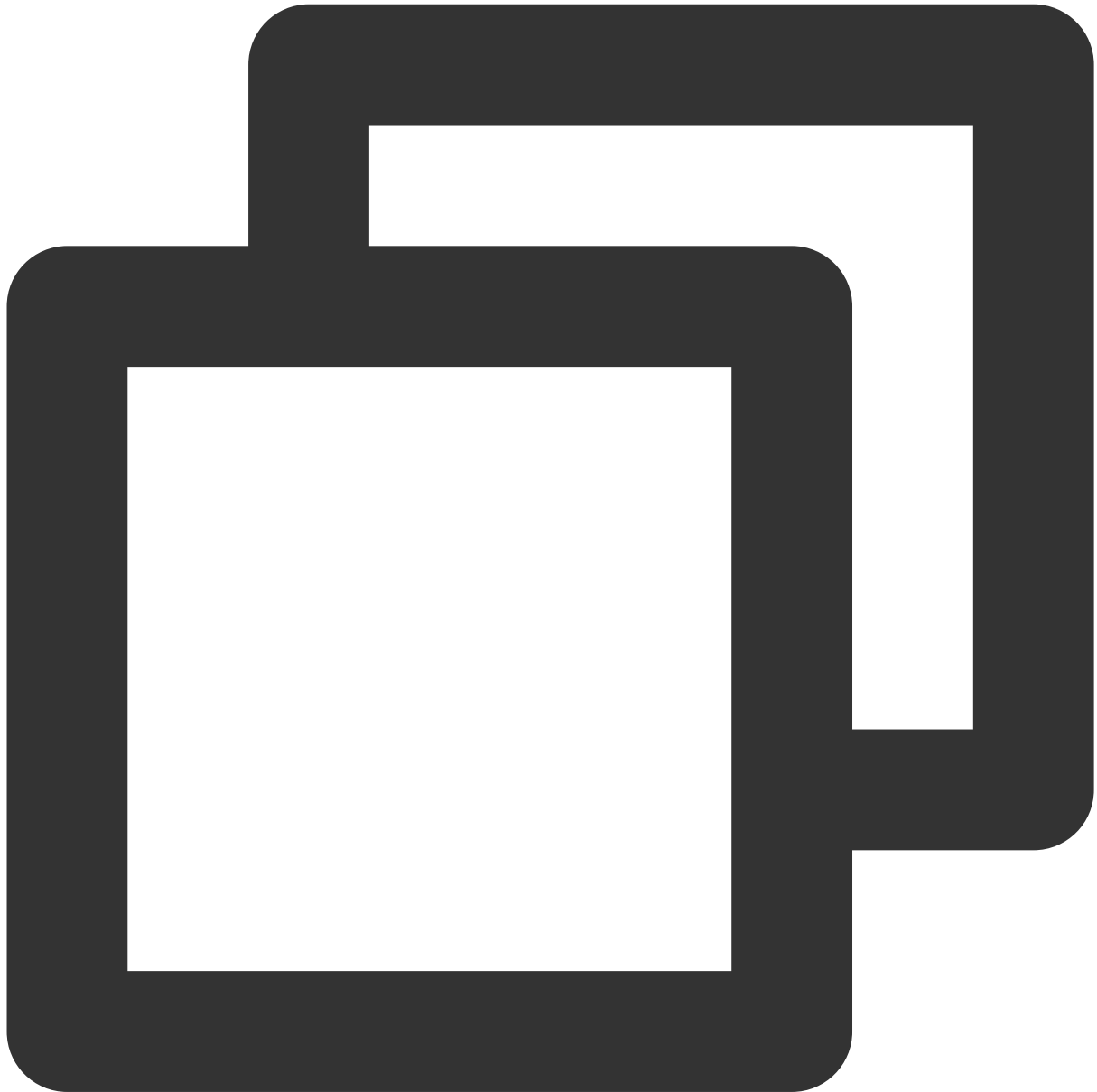
平均下载速度

运营商的下载次数、下载流量、速度

请求延时响应

### 健康度

统计 http\_code 小于500的请求占有所有请求的百分比。



```
* | select round(sum(case when http_code<500 then 1.00 else 0.00 end) / cast(count (
```

### 缓存命中率

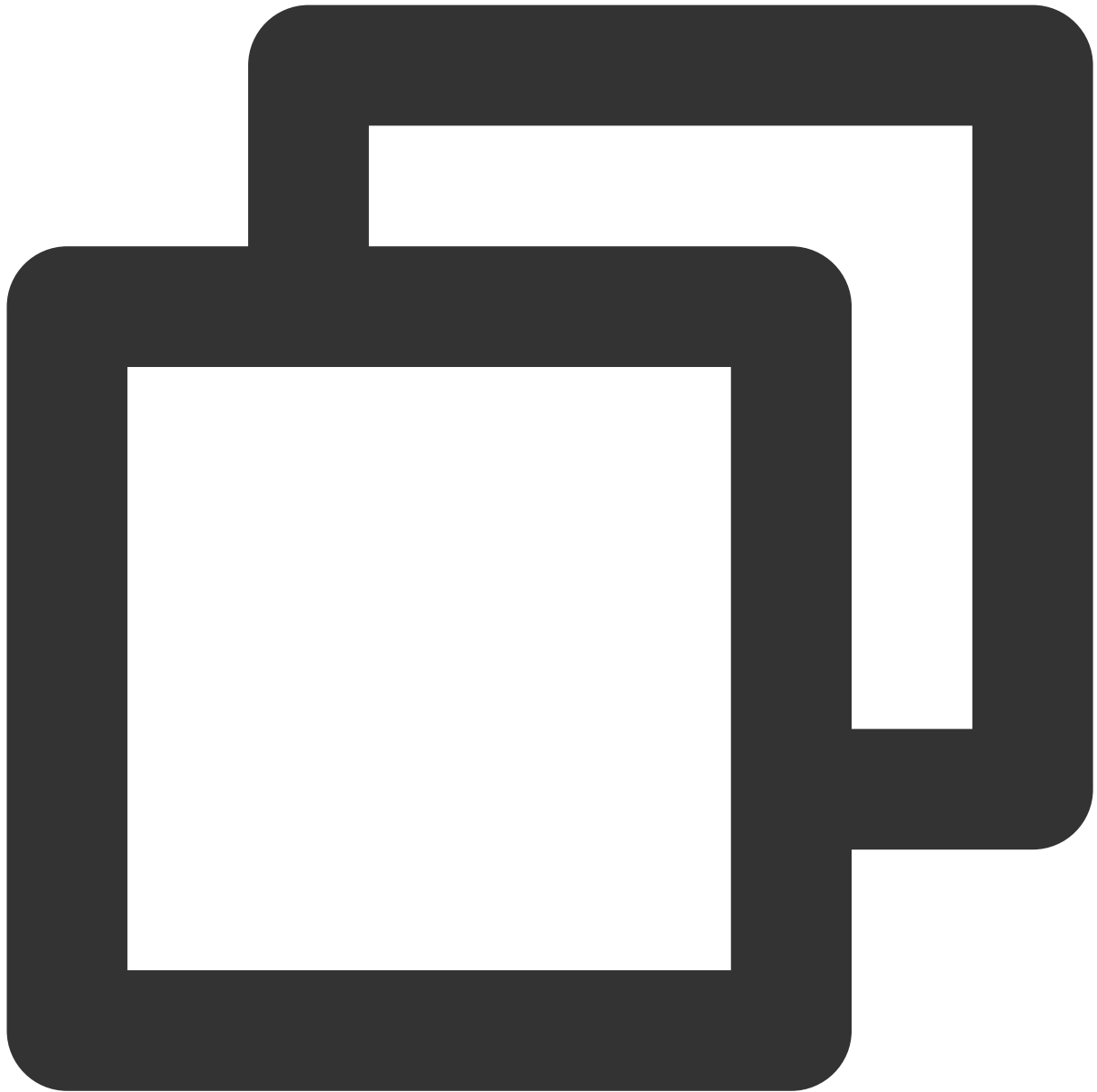
统计 return\_code 小于400的请求中， hit 为 “hit” 的请求百分比。



```
http_code<400 | select round(sum(case when hit='hit' then 1.00 else 0.00 end) / cas
```

#### 平均下载速度

统计一段时间内，总体下载量除以整体耗时获得平均下载速度。



```
* | select sum(rsp_size/1024.0) / sum(request_time/1000.0) as "平均下载速度(kb/s)"
```

运营商的下载次数、下载流量、速度

原理同上，使用 `ip_to_provider` 函数，将 `client_ip` 转化成对应的运营商。





```
* | select ip_to_provider(client_ip) as isp , sum(rsp_size)* 1.0 /(sum(request_time
```

#### 请求延时响应

将访问延时按照各窗口进行统计，可根据应用实际的情况来划分合适的延时时间窗口。



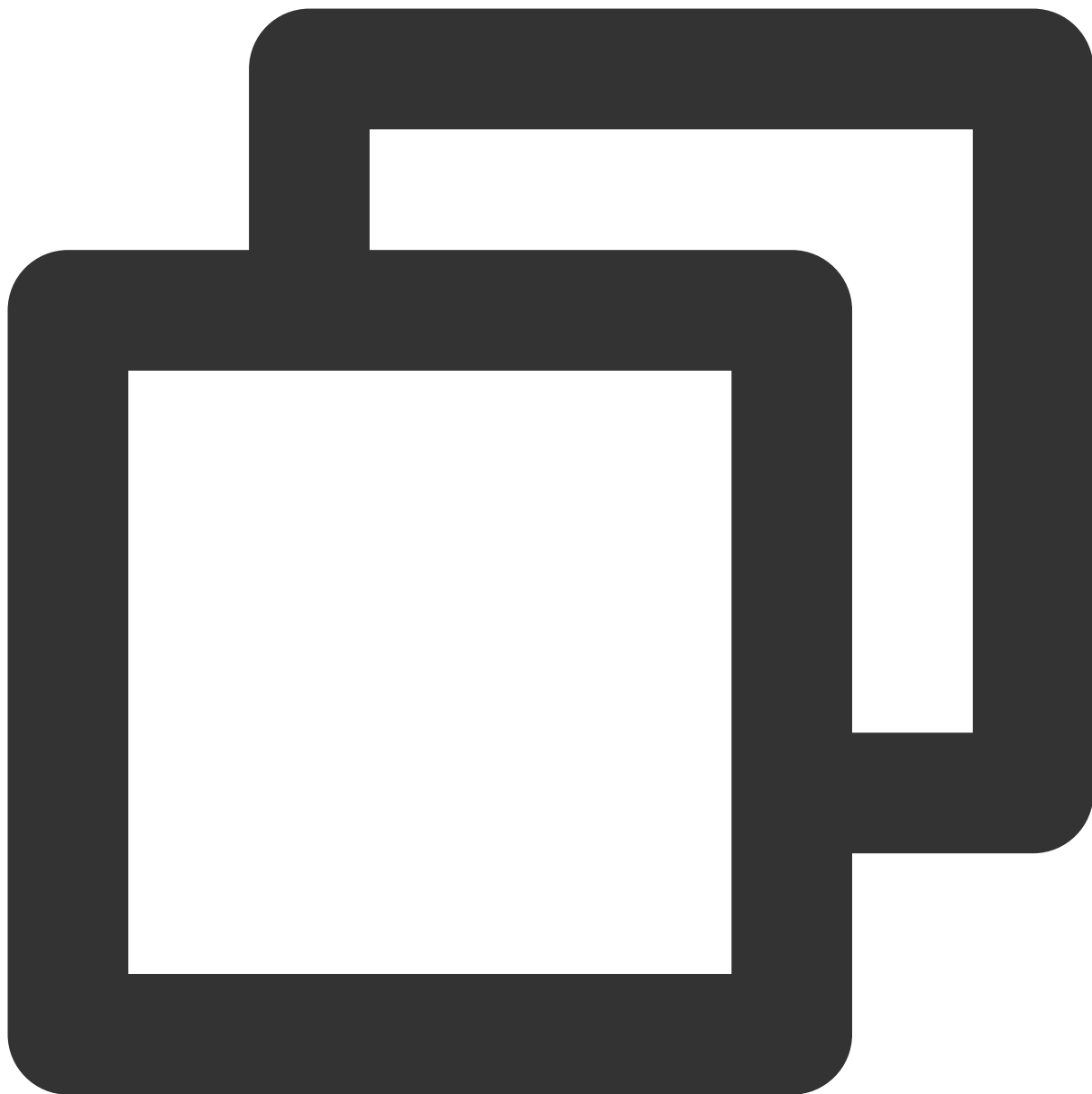
```
* | select case when request_time < 5000 then '~5s' when request_time < 6000 then
```

### CDN 质量和性能分析

访问错误一直是影响服务体验的重要一环，当出现错误的时候，需要快速定位当前错误 QPS 和比例是多少，哪些域名和 URI 影响最大，是否和地域、运营商有关，是不是发布的新版本导致。

#### 解决方案

查看4xx, 5xx错误码分布。



```
* | select http_code , count(*) as c where http_code >= 400 group by http_code or
```

从下面的错误分布图来看，主要的错误都是发生了404错误，说明被访问文件或内容不存在，这个时候就需要检查是不是资源已经被删除或者销毁。

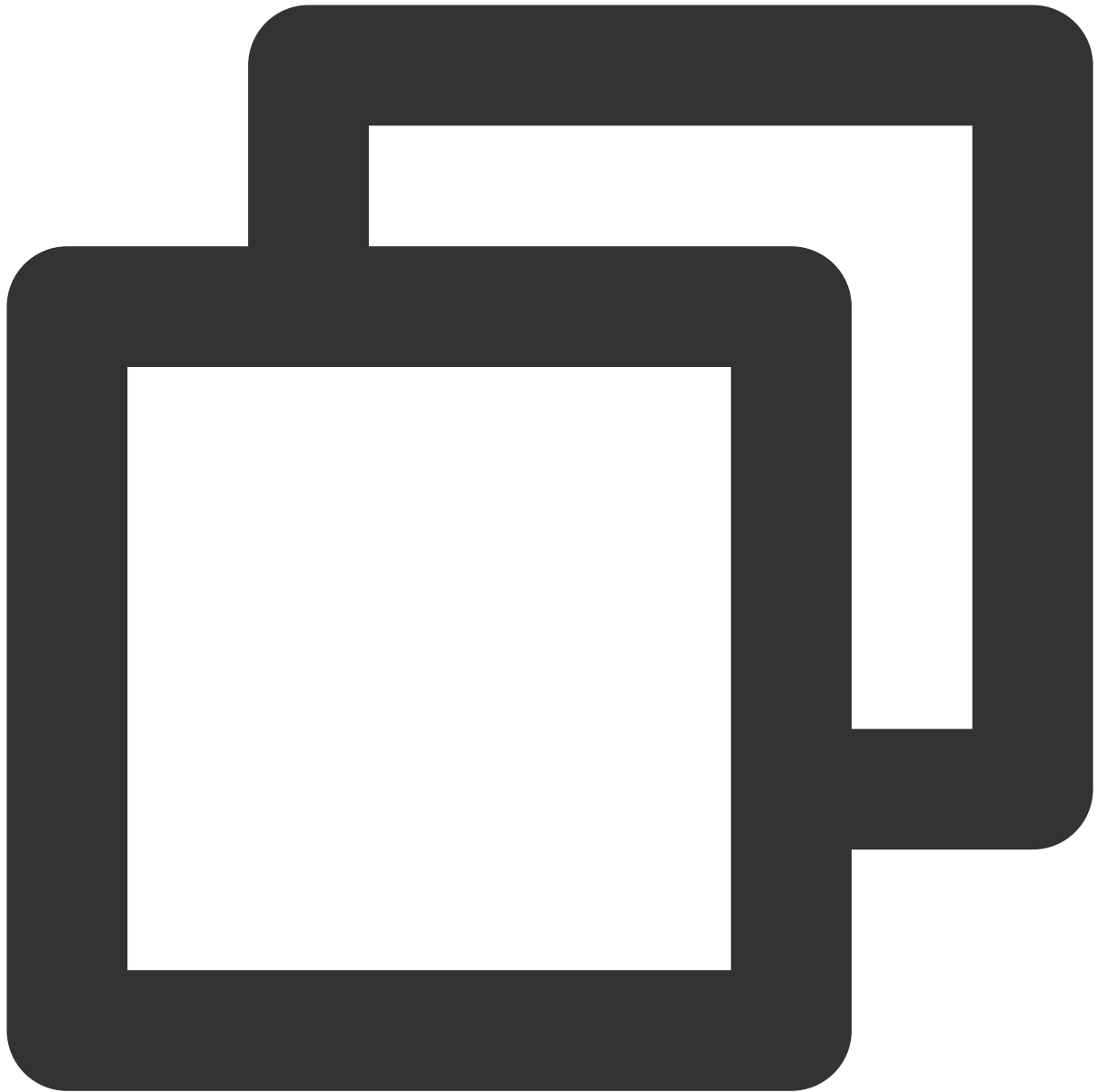
对于 `http_code > 400` 的请求，我们对其进行多维度分析，如按照域名和 uri 的维度进行 top 排序；省份，运营商角度查看错误次数；查看客户端分布。

**域名**



```
* | select host , count(*) as count where http_code > 400 group by host order
```

url



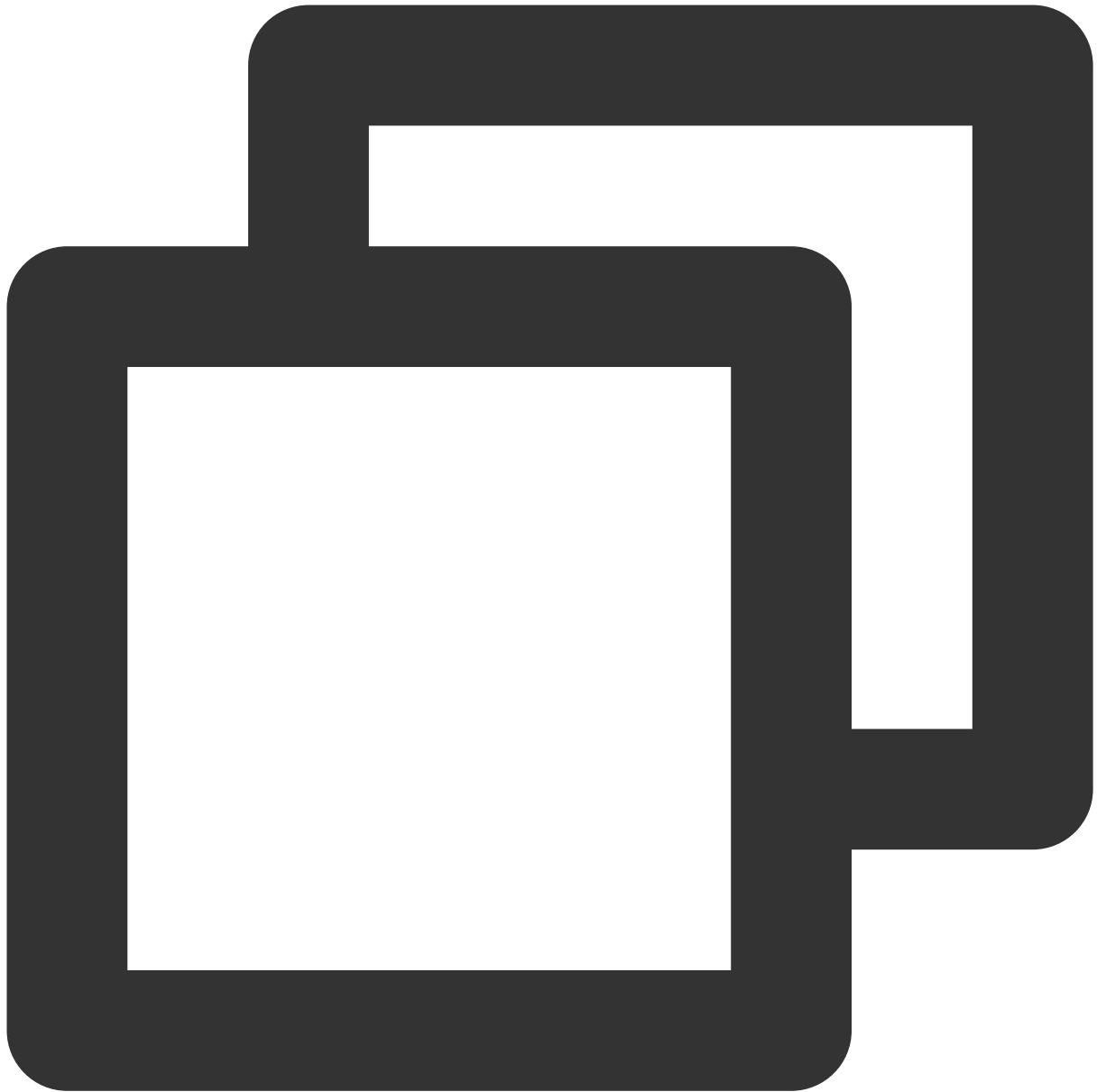
```
* | select url , count(*) as count where http_code > 400 group by url order by
```

省份, 运营商分析



```
* | select client_ip, ip_to_province(client_ip) as "province", ip_to_provider(client_ip) as "provider"
```

客户端分布



```
* | select ua as "客户端版本", count(*) as "错误次数" where http_code > 400 group by u
```

由图可看出，错误集中 Safari 客户端，定位问题后发现，是新版本 bug 导致在 Safari 浏览器窗口下，访问资源频繁失败。

## 用户行为分析

### 需求

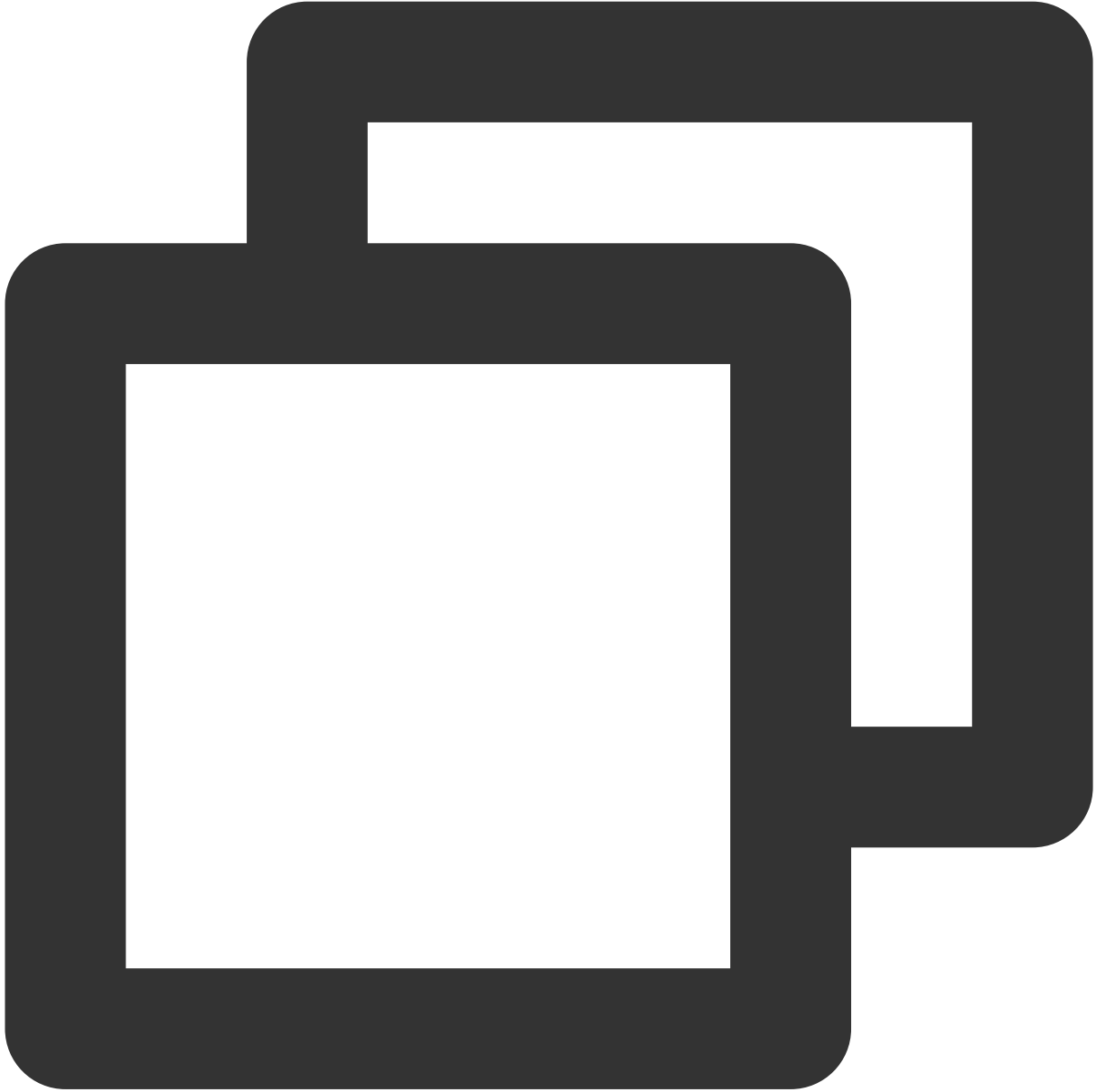
大部分用户是从哪里过来，是内部还是外部。

哪些资源用户是热门资源。

是否有用户在疯狂下载资源，行为是否符合预期。

### 解决方案

访问来源分析



```
* | select ip_to_province(client_ip) as province , count(*) as c group by province
```

访问 TopUrl





```
http_code < 400 | select url ,count(*) as "访问次数", round(sum(rsp_size)/1024.0/1024.0)
```

下载流量 Top 域名，按照各个域名下载数据量大小进行 Top 排序



```
* | select host, sum(rsp_size/1024) as "下载总量" group by host order by "下载总量"
```

下载量 Top 用户统计



```
* | SELECT CASE WHEN ip_to_country(client_ip)='香港' THEN concat(client_ip, ' ( Hong
```

有效访问 Top 用户统计



```
* | SELECT CASE WHEN ip_to_country(client_ip)='香港' THEN concat(client_ip, ' ( Hong
```

访问 PV、UV 统计，统计某一段时间内的访问次数和独立的 client ip 的变化趋势



```
* | select time_series(__TIMESTAMP__, '1m', '%Y-%m-%dT%H:%i:%s+08:00', '0') as time
```

# COS 访问日志分析

最近更新时间：2024-01-20 17:28:40

## 概述

**对象存储（Cloud Object Storage, COS）** 访问日志记录了用户对 COS 资源的访问信息，包括上传对象（PUT），删除对象（DELETE），访问对象（GET）等。通过分析访问日志，用户可以完成审计回溯，如删除资源记录，同时也可以完成资源热门相关的资源统计等能力。本文介绍 COS 如何访问日志。

## 前提条件

已将 COS 日志采集至日志服务（Cloud Log Service, CLS），详情请参见 [COS 开启实时日志](#)。

## 访问日志介绍

COS 访问日志记录了源存储桶，用户 ID，请求方法等信息。

字段序号	名称	含义	示例
1	eventVersion	记录版本	1.0
2	bucketName	存储桶名称	examplebucket-1250000000
3	qcsRegion	请求地域	ap-beijing
4	eventTime	事件时间（请求结束时间，UTC 0 时间戳）	2018-12-01T11:02:33Z
5	eventSource	用户访问的域名	examplebucket-1250000000.cos.ap-guangzhou.myqcloud.com
6	eventName	事件名称	UploadPart
7	remotelp	来源 IP	192.168.0.1
8	userSecretKeyId	用户访问 KeyId	AKIDNYVCdoJQyGJ5brTf

9	reservedFiled	保留字段	保留字段, 显示为 -
10	reqBytesSent	请求字节数 (Bytes)	83886080
11	deltaDataSize	请求对存储量的改变 (Bytes)	808
12	reqPath	请求的文件路径	/folder/text.txt
13	reqMethod	请求方法	put
14	userAgent	用户 UA	cos-go-sdk-v5.2.9
15	resHttpCode	HTTP 返回码	404
16	resErrorCode	错误码	NoSuchKey
17	resErrorMsg	错误信息	The specified key does not exist.
18	resBytesSent	返回字节数 (Bytes)	197
19	resTotalTime	请求总耗时 (毫秒, 等于响应末字节的时间-请求首字节的时间)	4295
20	logSourceType	日志源类型	USER (用户访问请求), CDN (CDN 回源请求)
21	storageClass	存储类型	STANDARD, STANDARD_IA, ARCHIVE
22	accountId	存储桶所有者ID	100000000001
23	resTurnAroundTime	请求服务端耗时 (毫秒, 等于响应首字节的时间-请求末字节的时间)	4295
24	requester	访问者	主账号 ID : 子账号 ID, 如果是匿名访问则显示 -。
25	requestId	请求 ID	NWQ1ZjY4MTBfMjZiMjU4NjRfOWI1N180NDBiYTY=
26	objectSize	对象大小 (Bytes)	808, 如果您使用分块上传, objectSize 字段只会在完成上传的时候显示, 各个分块上传期间该字段显示 -
27	versionId	对象版本 ID	随机字符串
28	targetStorageClass	目标存储类型, 发	STANDARD, STANDARD_IA, ARCHIVE

		起复制操作的请求 会记录该字段	
29	referer	请求的 HTTP referer	*.example.com 或者111.111.111.1
30	requestUri	请求 URI	"GET /fdgfdgsf%20/%E6%B5%AE%E7%82%B9%E6%95%B0 HTTP/1.1"

## 场景示例

### 场景1：审计追溯

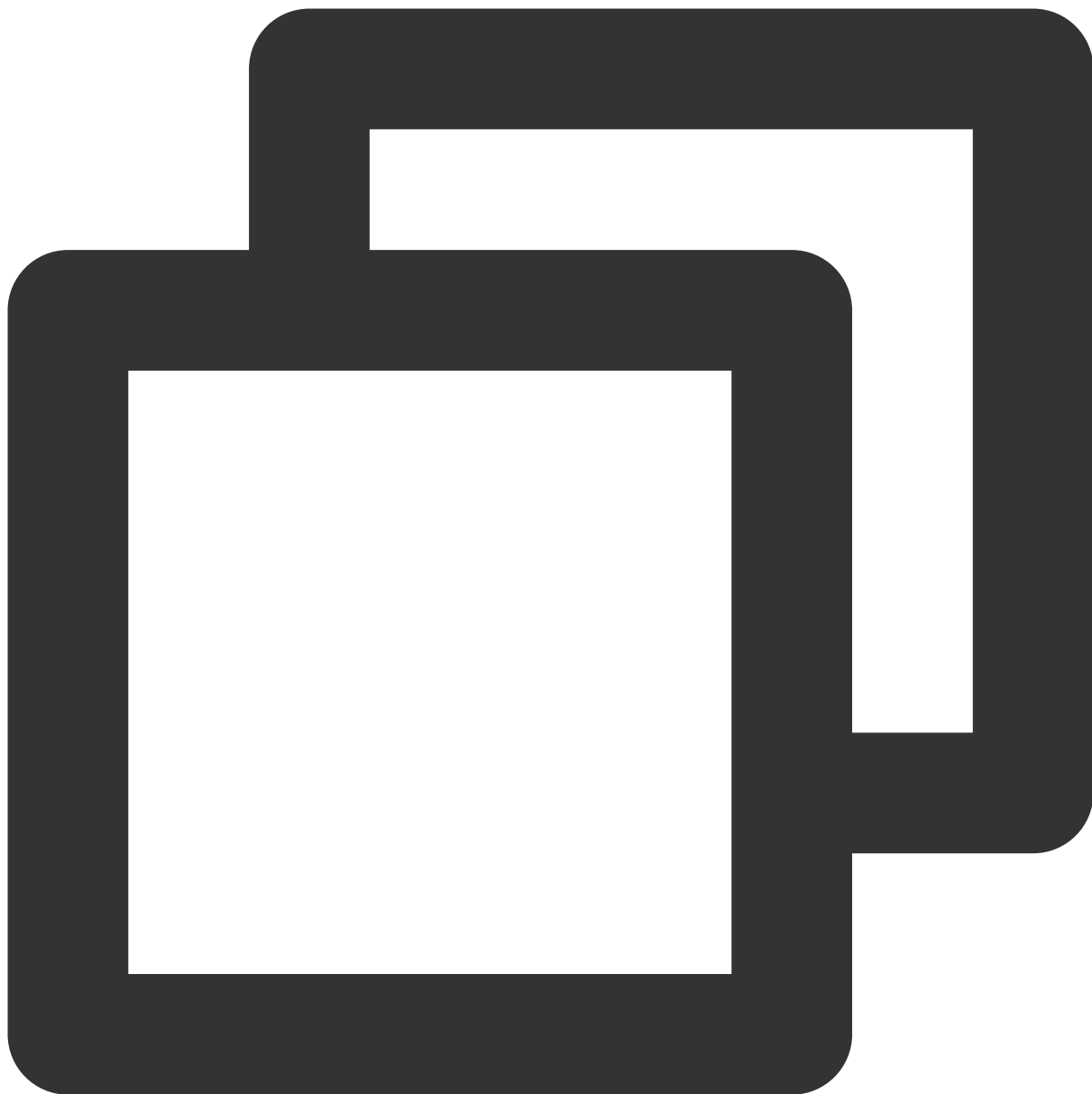
#### 需求场景

某个对象文件访问不了，定位原因。

#### 解决方案

进入 COS 访问日志检索页面，输入对象名称作为关键词检索日志。





```
json-log2019-05-09_00645d9a-1118-4d69-8411-cfd57ede9ea1_000
```

通过时间柱状图，得知近1天有14条日志记录。针对14条日志记录下钻分析，点击左侧字段快速分析栏，查看 **resHttpCode** 信息。

通过快速分析得知，6条非200的请求信息，其中5条 **resHttpCode** 为403的日志信息和一条 **resHttpCode** 为204日志信息，单击快速检索这两个 **httpcode** 的日志。

由日志可以得知，5条错误码为 **Access Deny** 日志均为访问对象失败日志，通过 **resHttpCode** 为204的日志发现，用户 `1000*****` 在8月24日20点16分，通过 **COS** 控制台执行了删除 **object** 操作，导致对象访问失败。

---

## 场景2：运营统计

### 需求场景

统计当天访问量 Top10热门的 bucket

统计当天某个 bucket 的访问趋势

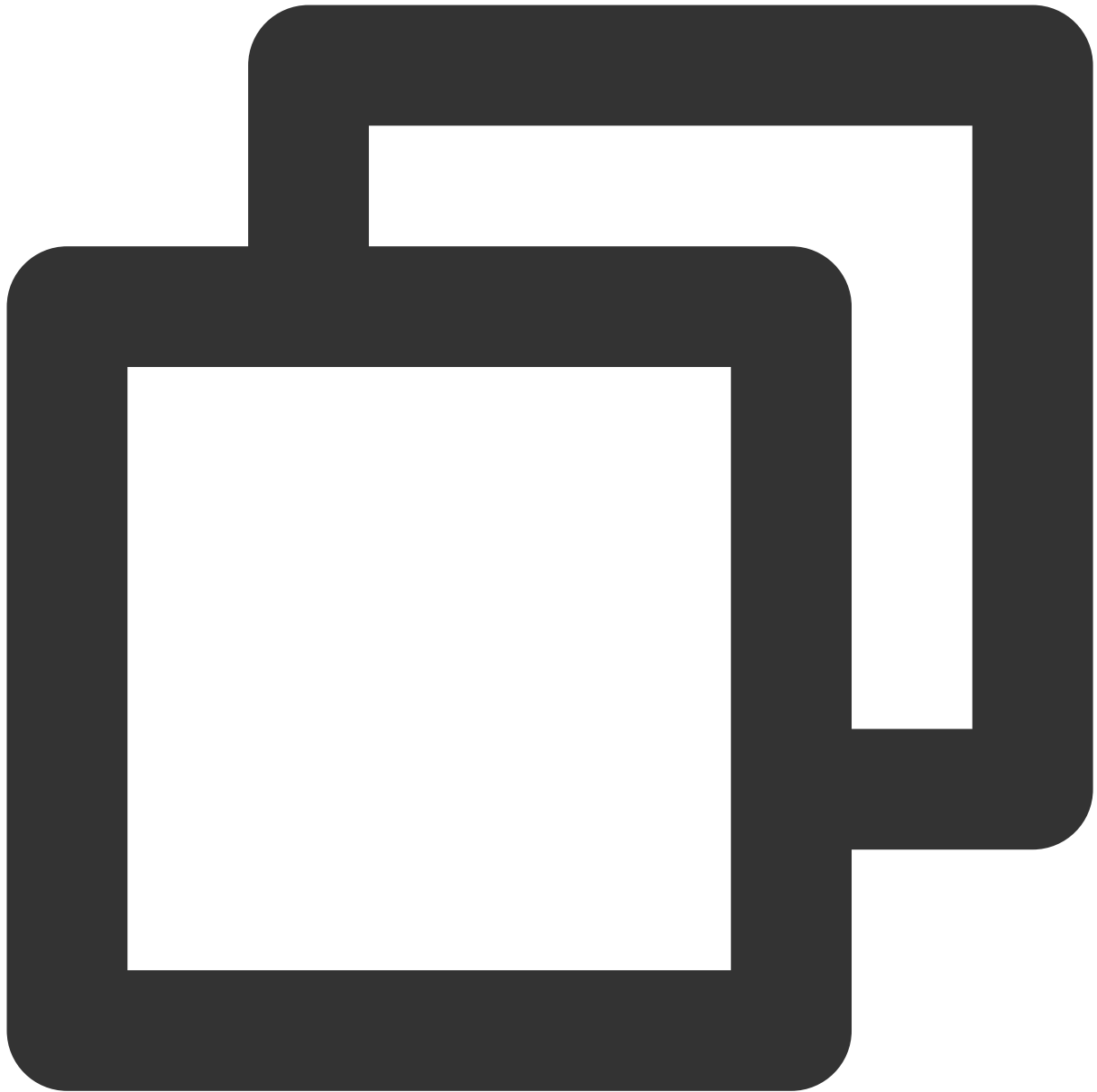
错误请求 Top10的访问者

失败操作的 bucket 分布

用户请求有效率趋势

### 解决方案

统计当天访问量 Top10热门的 bucket



```
(reqMethod:"GET") | select bucketName, count(*) group by bucketName
```

统计当天某个 bucket 的访问趋势



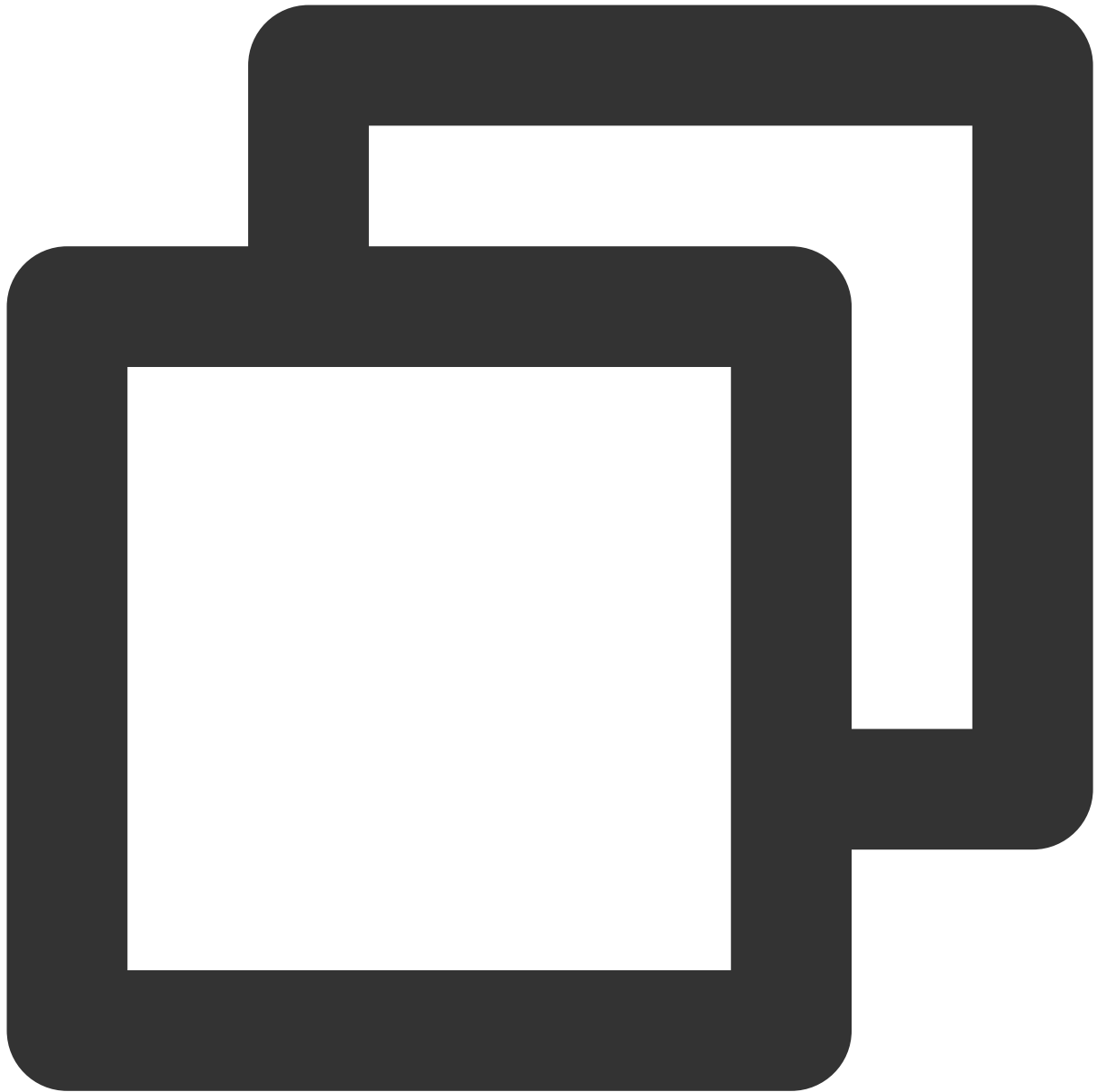
```
* | select time_series(TIMESTAMP, '1m', '%Y-%m-%dT%H:%i:%s+08:00', '0') AS time, co
```

错误请求 Top10的访问者



```
resHttpCode:>200 | select remoteIp, count(*) group by remoteIp
```

失败操作的 bucket 分布



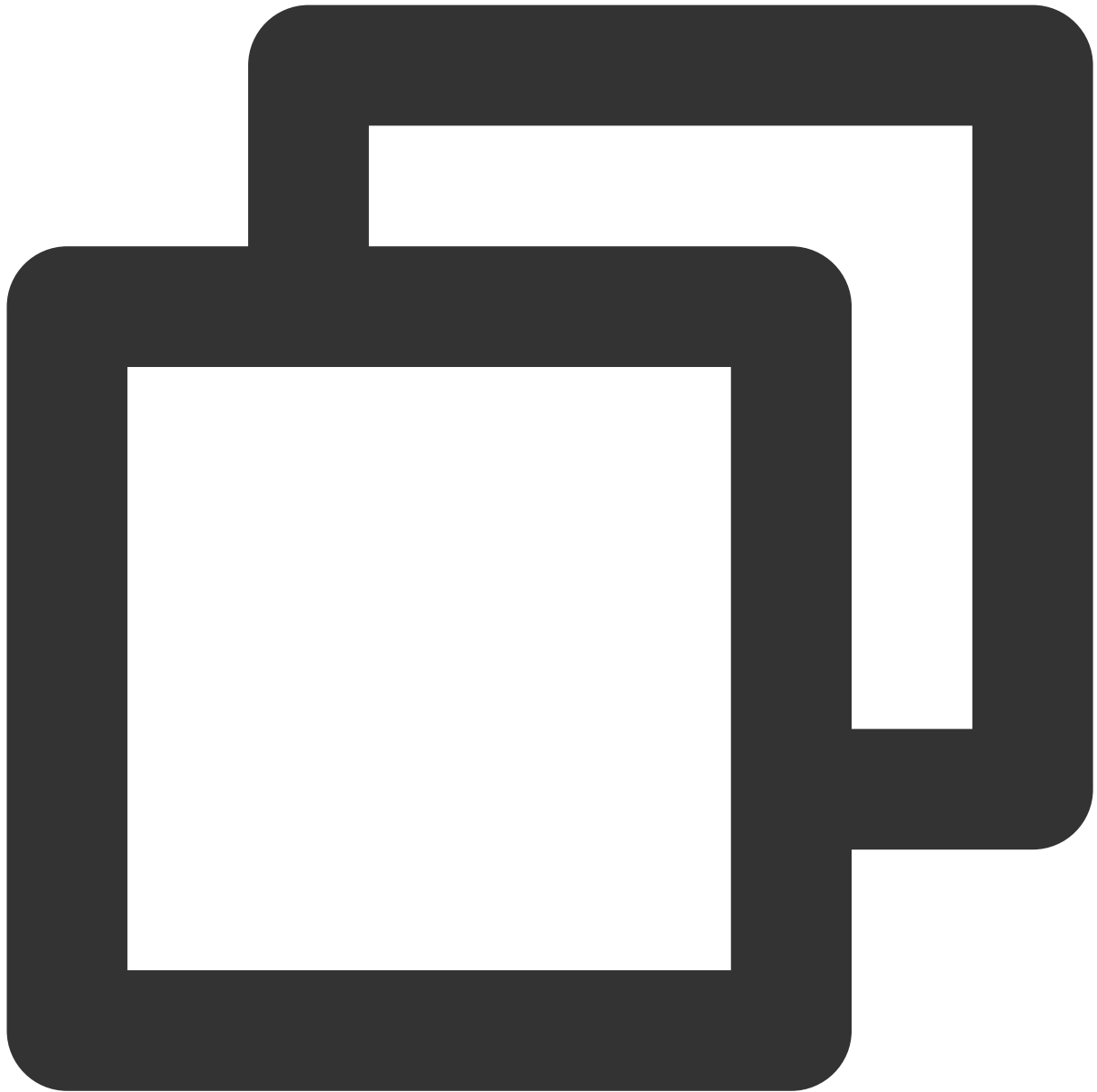
```
resHttpCode:>200 | select bucketName, count(*) group by bucketName
```

用户请求有效率趋势



```
* | select time_series(__TIMESTAMP__, '5m', '%Y-%m-%d %H:%i:%s', '0') as time,round
```

用户请求来源分布



```
* | select ip_to_province(remoteIp) as province , count(*) as c group by province
```



# Flowlog网络流日志分析

最近更新时间：2024-08-12 16:05:55

## 概述

[网络流日志 \(Flow Logs, FL\)](#) 为您提供全时、全流、非侵入的流量采集服务，可将采集的网络流量进行实时的存储、分析，适用于故障排查、合规审计、架构优化、安全检测等场景，让您的云上网络更加稳定、安全和智能。您可以创建指定采集范围（例如弹性网卡、NAT 网关、云联网跨地域流量）的网络流日志，来采集该范围内传入/传出的流量。

## 前提条件

已将 [云联网 \(Cloud Connect Network, CCN\)](#) 网络流日志采集至日志服务 (Cloud Log Service, CLS)，详见 [操作详情](#)。

如果您当前暂未将网络流日志采集至日志服务，您可使用日志服务免费提供的 Demo 日志主题来体验该功能，操作步骤详见 [使用 Demo 日志快速体验 CLS](#)。

## 场景示例

### CLS 分析云联网流日志

流日志 FlowLog 与 CLS 实现打通，用户可以将云联网流日志的数据实时投递至 CLS，并进一步使用 CLS 的检索和 SQL 分析能力，来满足不同场景下用户个性化的实时日志分析需求：

日志一键投递

百亿级日志，秒级分析

Dashboard 仪表盘实时日志可视化

一分钟实时告警

## 日志字段说明

云联网跨地域流量的网络流日志

其他类型的网络流日志

流日志将记录特定捕获窗口中，按**五元组 + 流量源地域 + 流量目的地域**规则过滤的网络流，即只有在捕获窗口中符合规则的网络流日志，才能记录为云联网跨地域流量的网络流日志记录。

## 五元组 + 流量源地域 + 流量目的地域

五元组即源 IP 地址、源端口、目的 IP 地址、目的端口和传输层协议这五个量组成的一个集合。

流量源地域指云联网跨地域流量发出的地域。

流量目的地域指云联网跨地域流量到达的地域。

## 捕获窗口

即一段持续时间，在这段时间内流日志服务会聚合数据，然后再发布流日志记录。捕获窗口大约为1分钟，推送时间约为5分钟。

字段	数据类型	说明
version	text	流日志版本。
region-id	text	记录日志的地域。
ccn-id	text	云联网唯一标识，如需确定云联网的信息请 <a href="#">联系我们</a> 。
srcaddr	text	源 IP。
dstaddr	text	目标 IP。
srcport	text	流量的源端口。该字段仅对 UDP/TCP 协议生效，当流量为其他协议时，该字段显示为“-”。
dstport	long	流量的目标端口。该字段仅对 UDP/TCP 协议生效，当流量为其他协议时，该字段显示为“-”。
protocol	long	流量的 IANA 协议编号。更多信息，请转到分配的 <a href="#">Internet 协议</a> 编号。
srcregionid	text	流量源地域。
dstregionid	text	流量目的地域。
packets	long	捕获窗口中传输的数据包的数量。当“log-status”为“NODATA”时，该字段显示为“-”。
bytes	long	捕获窗口中传输的字节数。当“log-status”为“NODATA”时，该字段显示为“-”。
start	long	当前捕获窗口收到第一个报文的时间戳，如果在捕获窗口内没有报文，则显示为该捕获窗口的起始时间，采用 Unix 秒的格式。
end	long	当前捕获窗口收到最后一个报文的时间戳，如果在捕获窗口内没有报文，则显示为该捕获窗口的结束时间，采用 Unix 秒的格式。
action	text	与流量关联的操作： ACCEPT：通过云联网正常转发的跨地域流量。 REJECT：因限速被阻止转发的跨地域流量。

log-status	text	流日志的日志记录状态： OK：表示数据正常记录到指定目标。 NODATA：表示捕获窗口中没有传入或传出网络流量，此时“packets”和“bytes”字段会显示为“-1”。
------------	------	--

流日志将记录特定捕获窗口中，按五元组规则过滤的网络流。

### 五元组

即源 IP 地址、源端口、目的 IP 地址、目的端口和传输层协议这五个量组成的一个集合。

### 捕获窗口

即一段持续时间，在这段时间内流日志服务会聚合数据，然后再发布流日志记录。捕获窗口大约为5分钟，推送时间约为5分钟。

字段	说明
version	流日志版本。
account-id	流日志的账户 AppID。
interface-id	弹性网卡 ID。
srcaddr	源 IP。
dstaddr	目标 IP。
srcport	流量的源端口。当流量为 ICMP 协议时，该字段表示 ICMP 的 id。
dstport	流量的目标端口。当流量为 ICMP 协议时，该字段表示 ICMP 的 type（高8bit）+code（低8bit）组合。
protocol	流量的 IANA 协议编号。更多信息，请转到分配的 <a href="#">Internet 协议</a> 编号。
packets	捕获窗口中传输的数据包的数量。
bytes	捕获窗口中传输的字节数。
start	捕获窗口启动的时间，采用 Unix 秒的格式。
end	捕获窗口结束的时间，采用 Unix 秒的格式。
action	与流量关联的操作： ACCEPT：安全组或网络 ACL 允许记录的流量。 REJECT：安全组或网络 ACL 未允许记录的流量。
log-status	流日志的日志记录状态： OK：表示数据正常记录到指定目标。 NODATA：表示捕获窗口中没有传入或传出网络流量，此时“packets”和“bytes”字段会显示为“-1”。

SKIPDATA：表示捕获窗口中跳过了一些流日志记录。可能是内部容量限制或内部错误引起的。

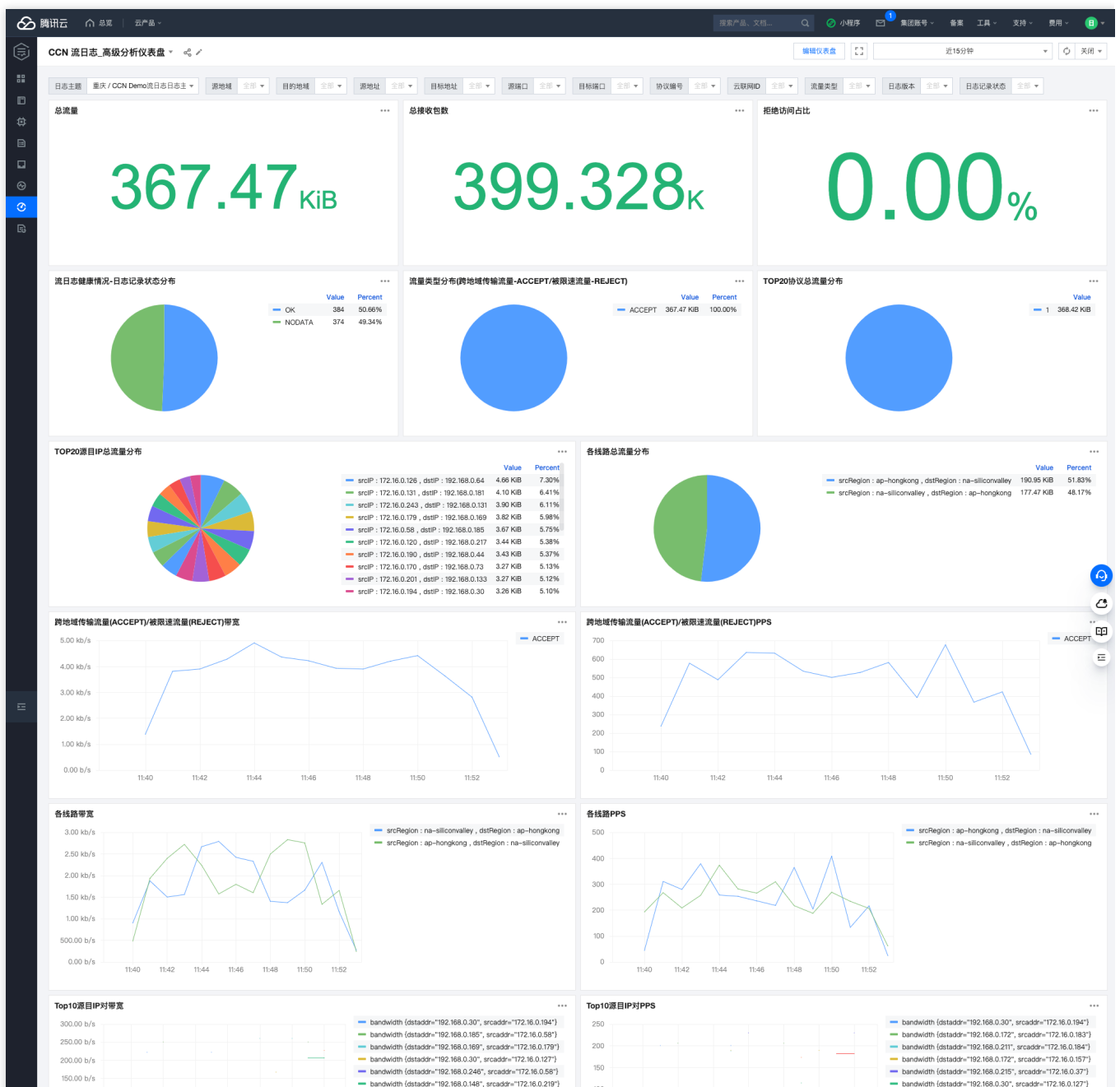
## 预置仪表盘

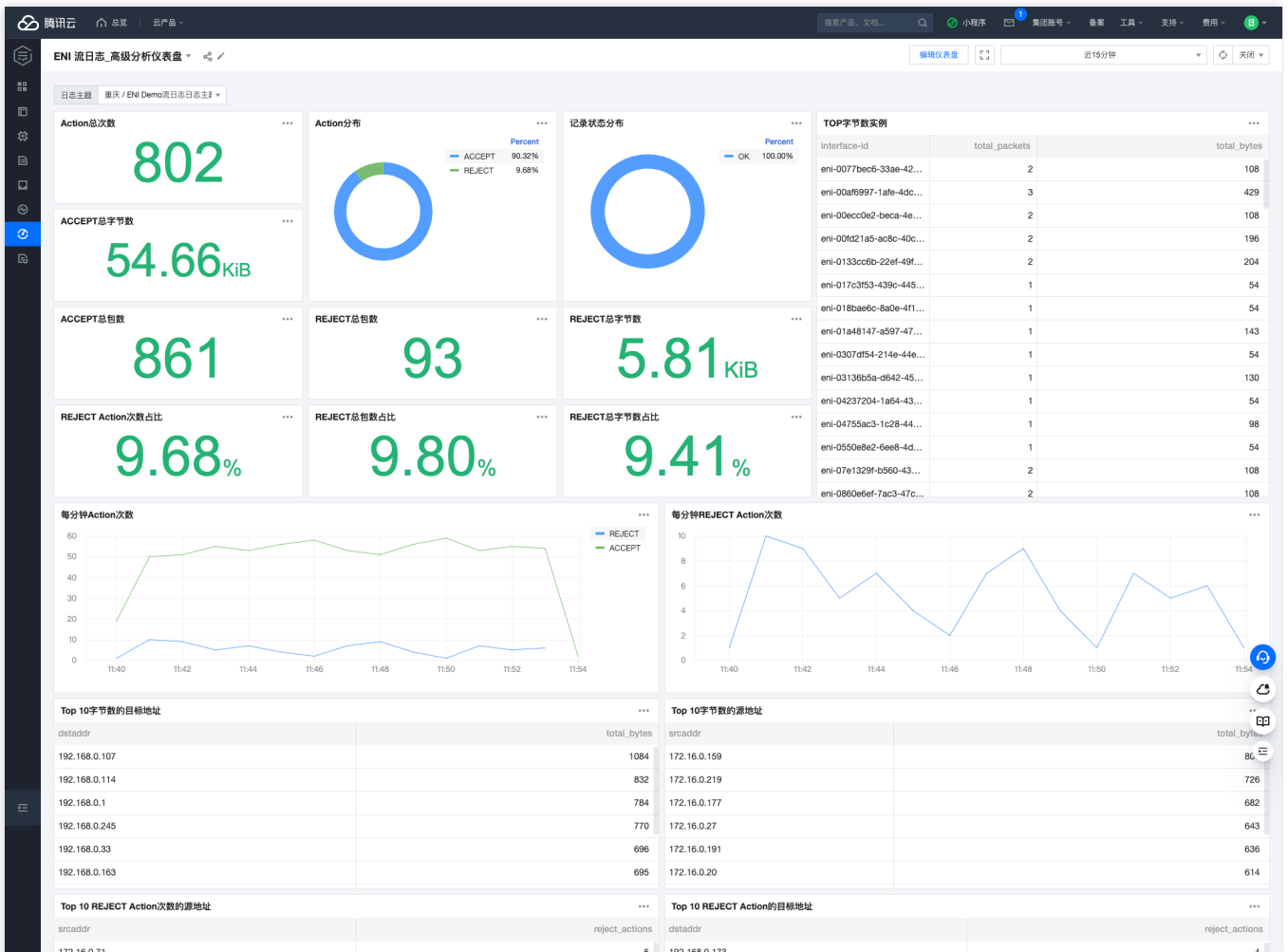
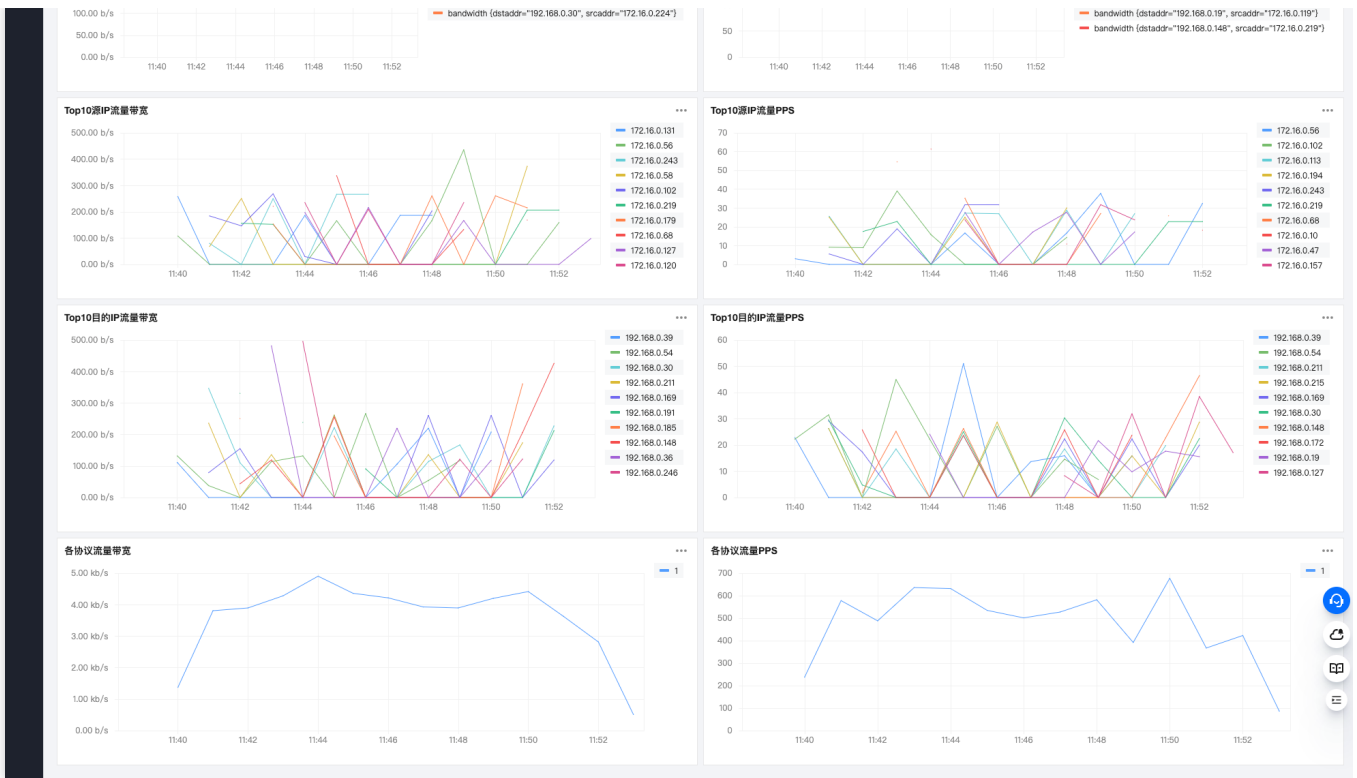
CLS 已将常用的云联网及弹性网卡流日志统计预置为仪表盘，您可以通过这些仪表盘快速了解当前网络状态。

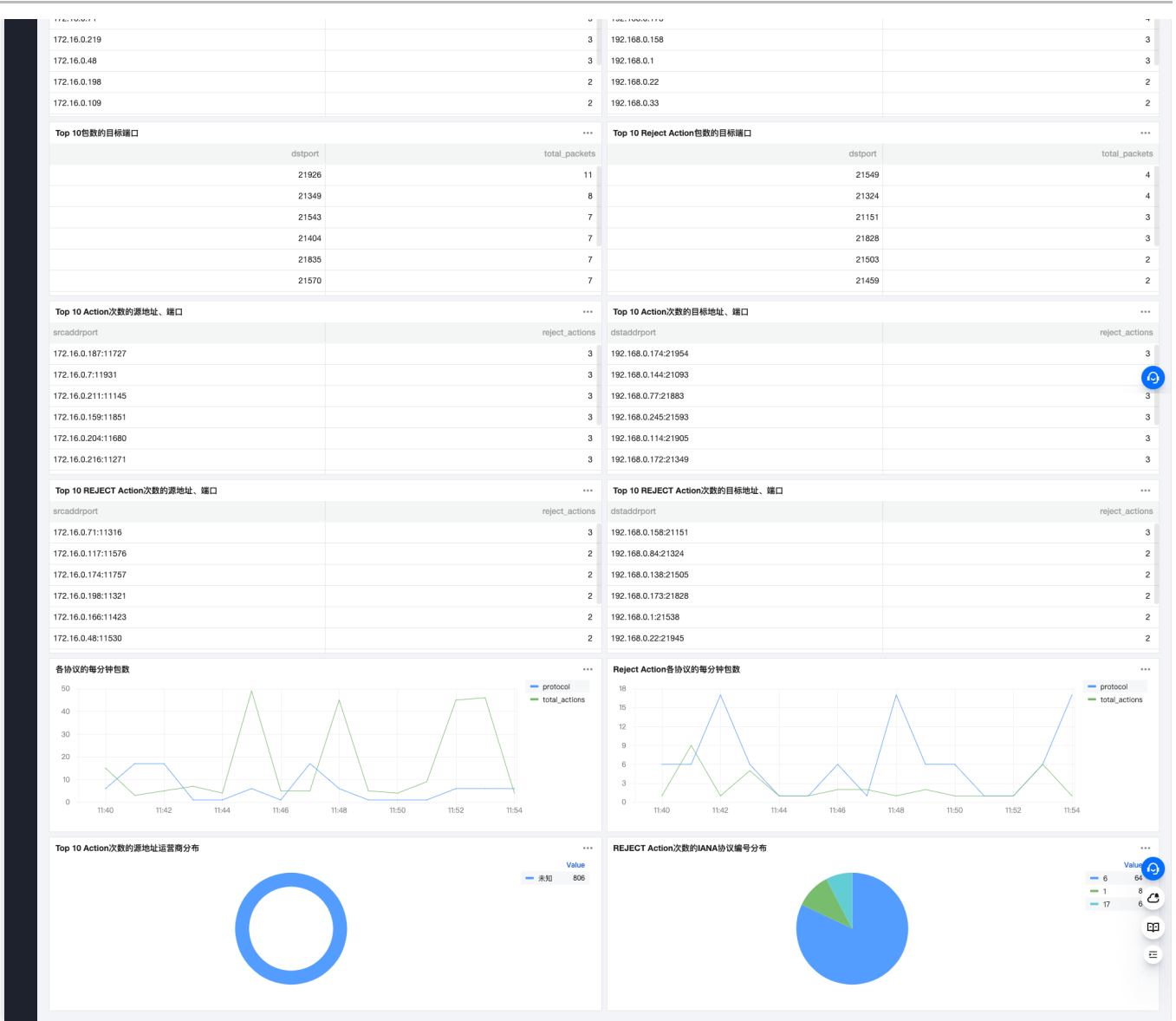
云联网：<https://consoleintl.cloud.tencent.com/cls/dashboard/d?templateId=flow-log-ccn-analysis-dashboard>

弹性网卡：<https://consoleintl.cloud.tencent.com/cls/dashboard/d?templateId=flow-log-eni-analysis-dashboard>

在仪表盘右上角单击**编辑仪表盘**可基于预置仪表盘进行编辑。



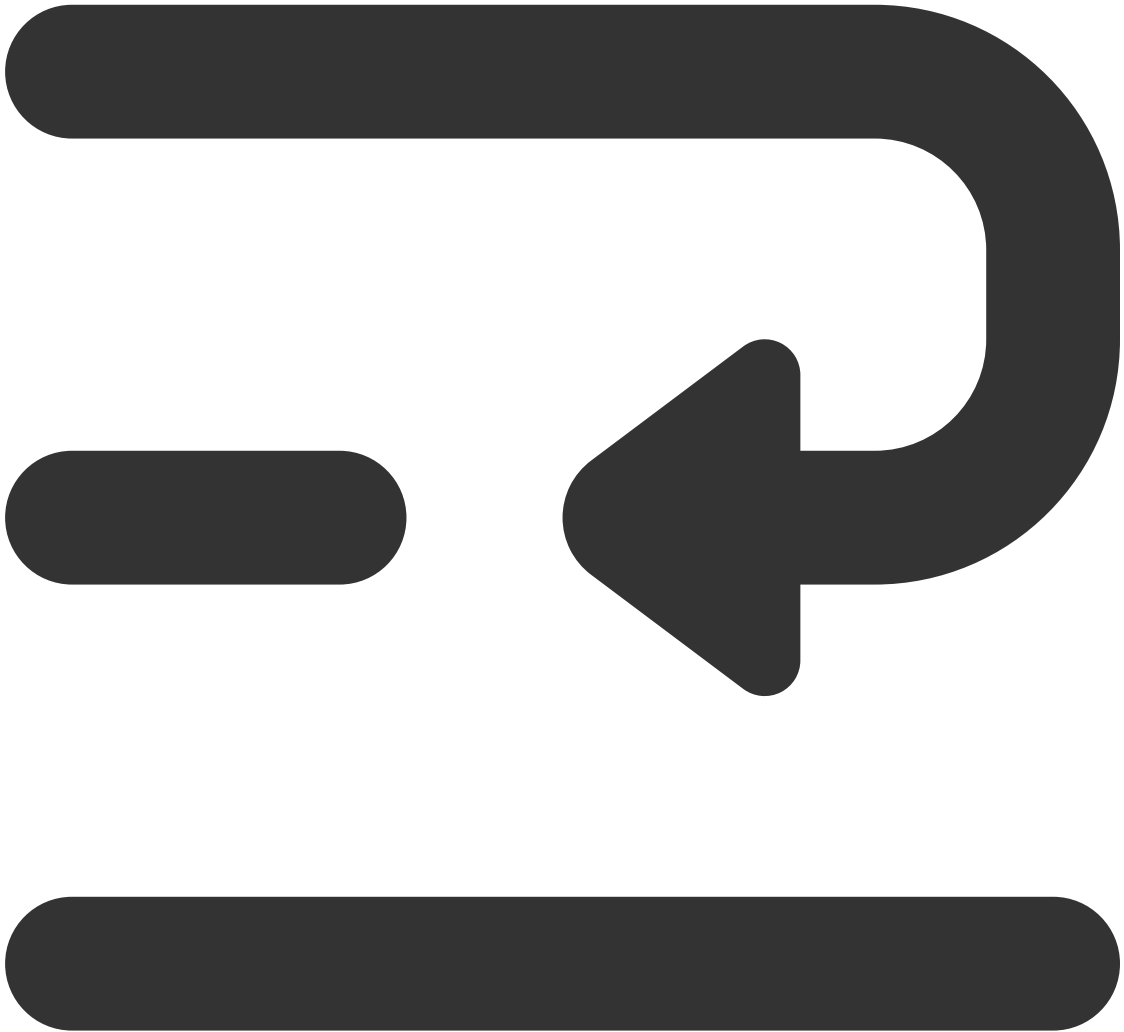


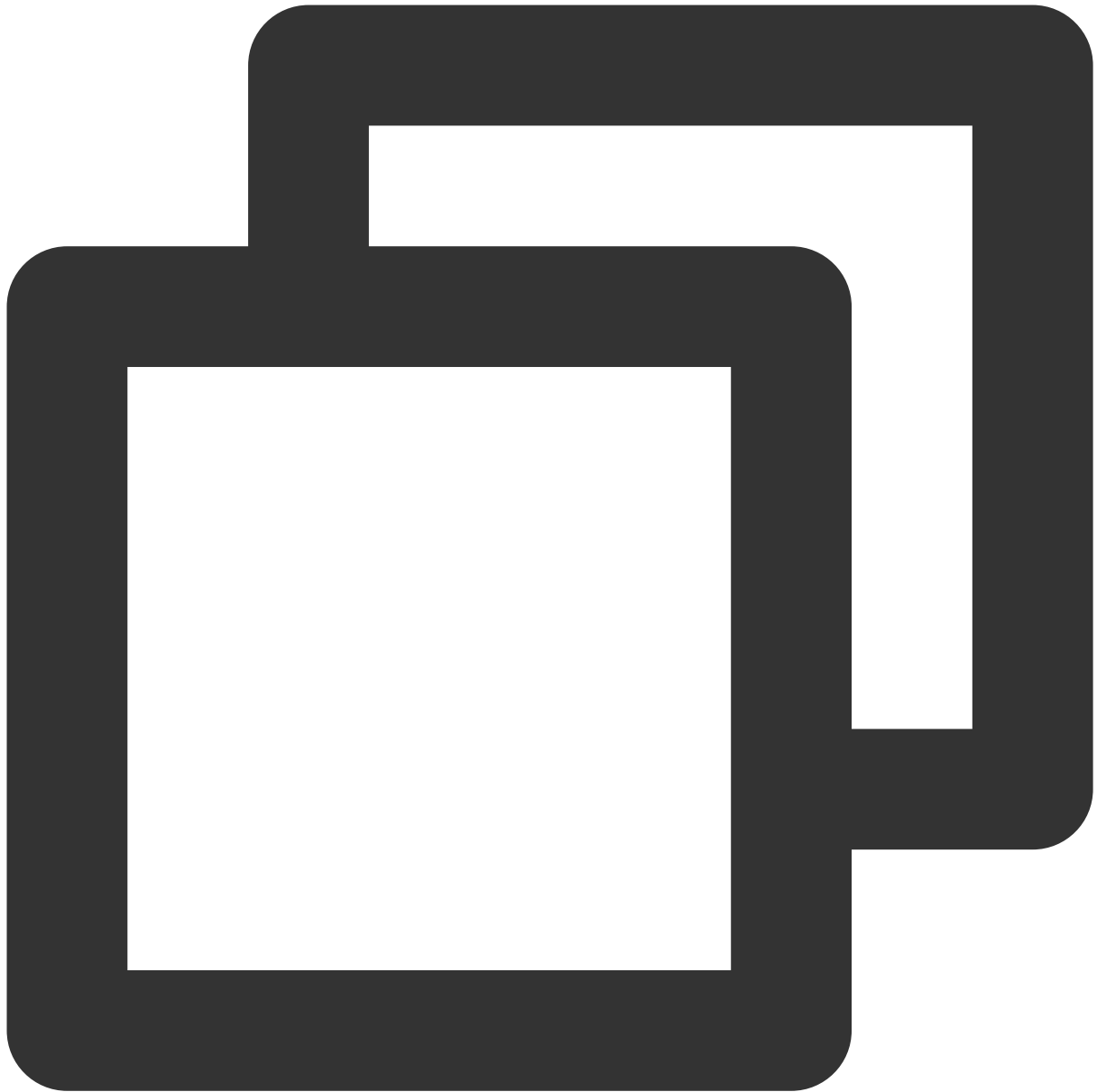


## 配置告警

例如为云联网中国香港-硅谷线路配置了带宽上限100Mbps，需监控当前带宽使用情况，连续10分钟带宽大于等于95Mbps时触发告警，以便在必要时对带宽上限进行调整。

1. 进入创建告警策略页面，操作步骤详见 [配置告警策略](#)。
2. 在执行语句中输入以下语句，时间范围选择1分钟，统计近1分钟内的中国香港-硅谷线路带宽使用情况。该执行语句的结果中 bandwidth 即为1分钟带宽，单位为 Mbps。

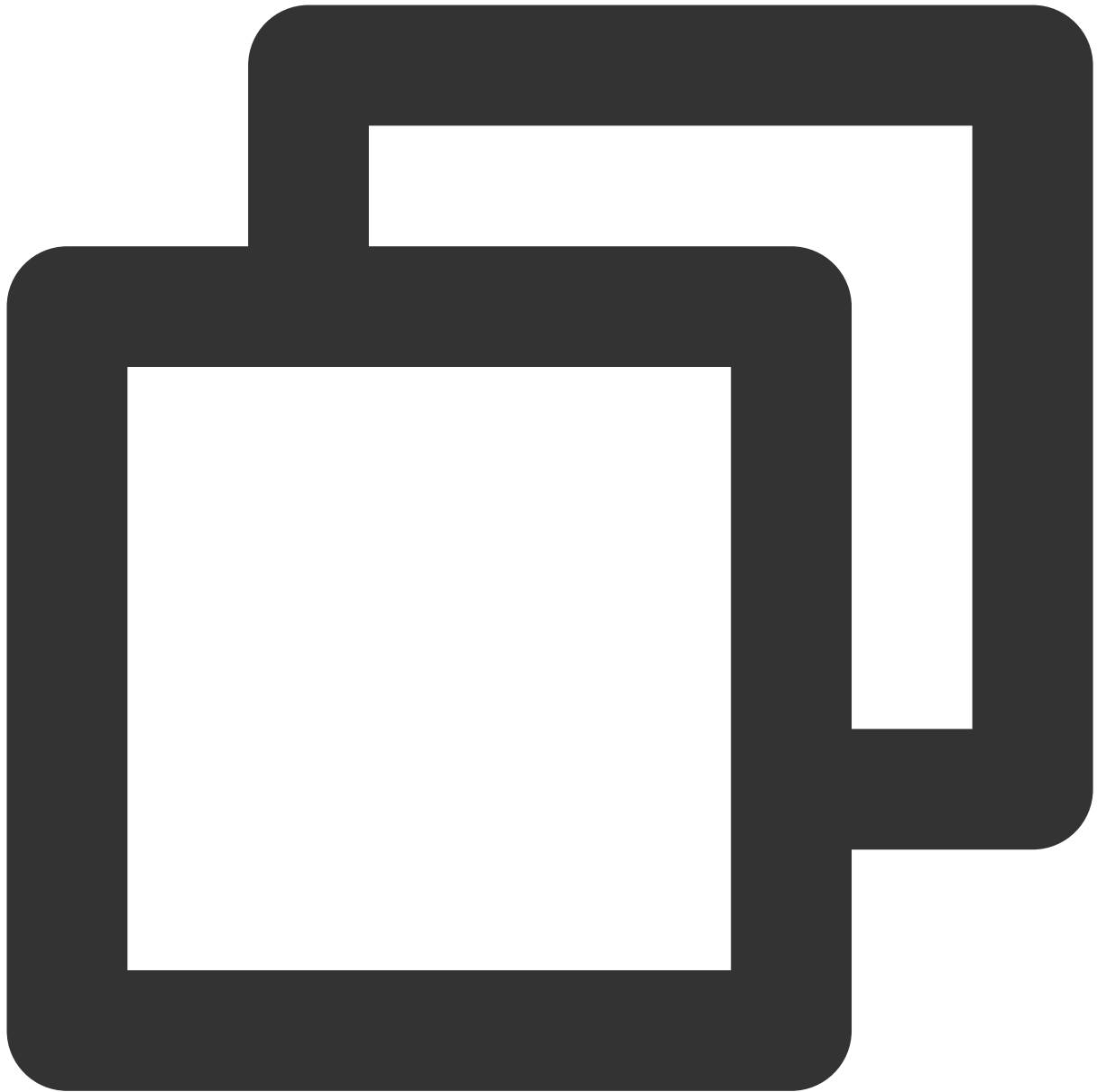




```
log-status:OK AND srcregionid:ap-hongkong AND dstregionid:na-siliconvalley | select
```

3. 触发条件如下，即带宽大于等于95Mbps 时，满足告警条件。





```
$1.bandwidth > 95
```

4. 执行周期：固定频率，每1分钟执行一次。

5. 告警通知-告警频率：持续10个周期满足触发条件则始终触发告警，即连续10分钟带宽大于等于95Mbps 时触发告警。

针对预置仪表盘中的图表，可以单击右上角的**添加到监控告警**将该图表中的指标添加到告警策略中。



# TKE 事件日志分析

最近更新时间：2024-01-20 17:28:40

## 概述

集群内的状况层出不穷，变化莫测，如节点状态异常，Pod 重启等，如果无法第一时间感知状况，会错过最佳的问题处理时间，待问题扩大，影响到业务时才发现往往已经为时已晚。

而事件日志（Event）记录了全面的集群状态变更信息，不仅可以帮助用户第一时间发现问题，也是排查问题的最佳帮手。

## 什么是事件日志

Event 是 Kubernetes 中众多资源对象中的一员，通常用来记录集群内发生的状态变更，大到集群节点异常，小到 Pod 启动、调度成功等等。我们常用的 `kubectl describe` 命令就可以查看相关资源的事件信息。

## 事件日志字段说明

```

{
  "event": {
    "firstTimestamp": "2020-11-25T14:10:17Z"
    "reason": "EvictionThresholdMet"
    "metadata": {
      "uid": "3ba47a24-135d-4eef-bbca-5b94a5d0b83e"
      "managedFields": [ ... ]
      "resourceVersion": "1219076"
      "creationTimestamp": "2020-11-25T14:10:25Z"
      "name": "172.16.18.13.164ac58ada1dec47"
      "namespace": "default"
      "selfLink": "/api/v1/namespaces/default/events/172.16.18.13.164ac58ada1dec47"
    }
    "involvedObject": {
      "uid": "172.16.18.13"
      "kind": "Node"
      "name": "172.16.18.13"
    }
    "reportingInstance": ""
    "lastTimestamp": "2020-11-28T07:45:22Z"
    "count": 23538
    "source": {
      "component": "kubelet"
      "host": "172.16.18.13"
    }
    "message": "Attempting to reclaim ephemeral-storage"
    "type": "Warning"
    "reportingComponent": ""
  }
}

```

- 级别 (Type)：目前仅有“Normal”和“Warning”，但是如果需要，可以使用自定义类型。
- 资源类型/对象 (Involved Object)：事件所涉及的对象，例如 Pod, Deployment, Node 等。
- 事件源 (Source)：报告此事件的组件；例如 Scheduler、Kubelet等。
- 内容 (Reason)：当前发生事件的简短描述，一般为枚举值，主要在程序内部使用。
- 详细描述 (Message)：当前发生事件的详细描述信息。
- 出现次数 (Count)：事件发生的次数。

## 如何使用事件日志去排查问题

日志服务 (Cloud Log Service, CLS) 提供针对 kubernetes 事件日志的一站式服务，包含采集，存储，检索分析能力。用户仅需一键开启集群事件日志功能，即可获取开箱即用的事件日志可视化分析仪表盘。通过可视化的图表，用户可以轻松通过控制台解决大多数常见的运维问题。

### 前提条件

已购买容器服务 (Tencent Kubernetes Engine, TKE)，并开启集群事件日志，详情请参考 [操作指南](#)。

### 场景1：一台 Node 节点出现异常，定位原因

1. 登录 [TKE 控制台](#)。

2. 在左侧导航栏中，单击**日志管理 > 事件日志**。
3. 在事件检索页面，选择**事件总览**页签，并在过滤项中输入异常节点名称。  
查询结果显示，有一条 **节点磁盘空间不足** 的事件记录查询结果如下图：

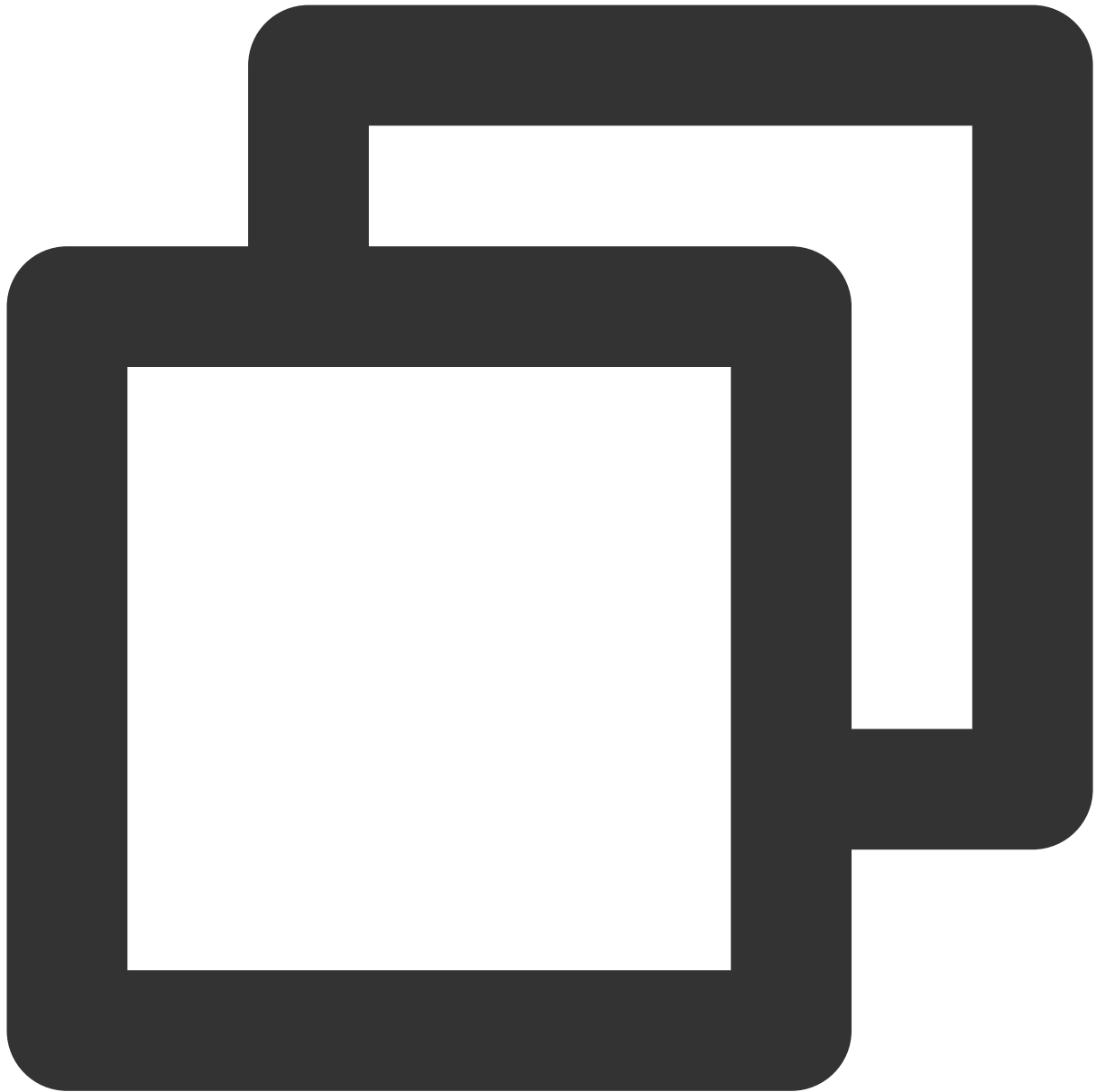
进一步查看异常事件趋势和异常 Top 事件：

可以发现，**2020-11-25** 号开始，节点 **172.16.18.13** 由于磁盘空间不足导致节点异常，此后 kubelet 开始尝试驱逐节点上的 pod 以回收节点磁盘空间。

## 场景2：节点触发扩容了，用户需要对扩容过程进行回溯，以确定具体原因

开启 **节点池**「弹性伸缩」的集群，CA（cluster-autoscaler）组件会根据负载状况自动对集群中节点数量进行增减。如果集群中的节点发生了自动扩（缩）容，用户可通过事件检索对整个扩（缩）容过程进行回溯。

1. 登录 **TKE 控制台**。
2. 在左侧导航栏中，单击**日志管理 > 事件日志**。
3. 在事件检索页面，单击**全局检索**页签，并输入以下检索命令：



```
event.source.component : "cluster-autoscaler"
```

4. 在左侧隐藏字段中，选

择 `event.reason`、`event.message`、`event.involvedObject.name`、`event.involvedObject.name` 进行显示，将查询结果按照 日志时间 倒序排列。

通过上图的事件流水，可以看到节点扩容操作在 2020-11-25 20:35:45 左右，分别由三个 nginx Pod(nginx-5dbf784b68-tq8rd、nginx-5dbf784b68-fpvbx、nginx-5dbf784b68-v9jv5) 触发，最终扩增了3个节点，后续的扩容由于达到节点池的最大节点数没有再次触发。

# TKE 审计日志分析

最近更新时间：2024-01-20 17:28:40

## 概述

以前，排查这些问题，对用户来说并不容易。生产环境中的 Kubernetes 集群通常是一个相当复杂的系统，底层是各种异构的主机、网络、存储等云基础设施，上层承载着大量的应用负载，中间运行着各种原生（例如：Scheduler、Kubelet）和第三方（例如：各种 Operator）的组件，负责对基础设施和应用进行管理和调度；此外不同角色的人员频繁地在集群上进行部署应用、添加节点等各种操作。因此在集群运维场景中，用户常见的问题有：

集群中的某个应用被删除了，谁干的？

Apiserver 的负载突然变高，大量访问失败，集群中到底发生了什么？

集群节点被封锁了，是谁在什么时候操作的？

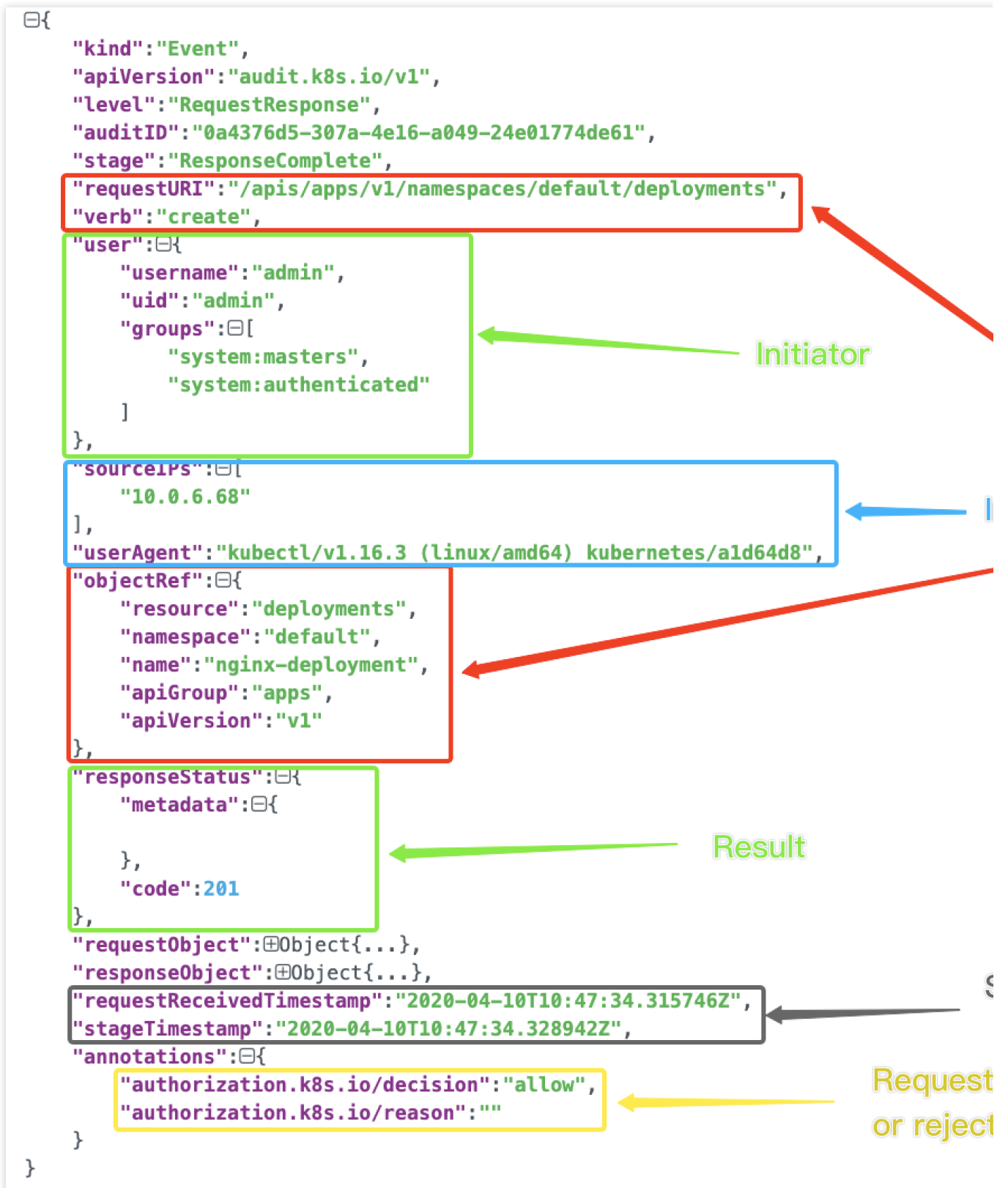
当前日志服务（Cloud Log Service, CLS）与 [容器服务（Tencent Kubernetes Engine, TKE）](#) 已实现打通。其中，Kubernetes 审计日志（Audit）将是可以帮助用户快速解决以上这些问题的重要工具。

## 什么是审计日志

在 Kubernetes 中，所有对集群状态的查询和修改都是通过向 Apiserver 发送请求，而审计日志是 Kube-apiserver 产生的可配置策略的结构化日志，记录了对 Apiserver 的访问事件。通过查看、分析审计日志，可以追溯对集群状态的变更；了解集群的运行状况；排查异常；发现集群潜在的安全、性能风险等等。

## 审计日志字段说明

每一条审计日志都是一个 JSON 格式的结构化记录，包括元数据（metadata）、请求内容（requestObject）和响应内容（responseObject）3个部分。其中元数据一定会存在，请求和响应内容是否存在取决于审计级别。元数据包含了请求的上下文信息，例如谁发起的请求，从哪里发起的，访问的 URI 等等。



## 如何使用审计日志去排查问题



CLS 提供针对 kubernetes 审计日志的一站式服务，包含采集，存储，检索分析能力。用户仅需一键开启集群审计日志功能，即可获取开箱即用的审计日志可视化分析仪表盘。通过可视化的图表，用户可以轻松通过控制台解决大多数常见的运维问题。

## 前提条件

已购买 TKE，并开启集群审计日志，详情请参考 [操作指南](#)。

### 场景1：集群中的某个应用被删除了，谁操作的？

1. 登录 [TKE 控制台](#)。
2. 在左侧导航栏中，单击**集群运维 > 审计检索**。
3. 在审计检索页面，单击**K8S对象操作概览**标签，指定操作类型为 delete 和资源对象 nginx。

查询结果如下图所示：

由图可见，是 `10001****7138` 这个账号，对应用「nginx」进行了删除。可根据账号 ID 在 [访问管理 > 用户列表](#) 中找到关于此账号的详细信息。

### 场景2：Apiserver 的负载突然变高，大量访问失败，集群中到底发生了什么？

1. 登录 [TKE 控制台](#)。
2. 在左侧导航栏中，单击**集群运维 > 审计检索**。
3. 在审计检索页面，单击**聚合检索**标签，该页签提供了从用户、操作类型、返回状态码等多个维度对于 Apiserver 访问聚合趋势图。

通过以上图表得知，用户 `tke-kube-state-metrics` 的访问量远高于其他用户，并且在“操作类型分布趋势”图中可以看出大多数都是 list 操作，在“状态码分布趋势”图中可以看出，状态码大多数为403，根据 `tke-kube-state-metrics` 关键词，检索日志。

结合业务日志可知，由于 RBAC 鉴权问题导致 `tke-kube-state-metrics` 组件不停的请求 Apiserver 重试，导致 Apiserver 访问剧增。

### 场景3：集群节点被封锁了，是谁在什么时候操作的？

1. 登录 [TKE 控制台](#)。
2. 在左侧导航栏中，单击**集群运维 > 审计检索**。
3. 在审计检索页面，单击**节点操作概览**标签，填写被封锁的节点名。

查询结果如下图所示：

由图可见，是 `10001****7138` 这个账号在 `2020-11-30T06:22:18` 时对 `172.16.18.13` 这台节点进行了封锁操作。

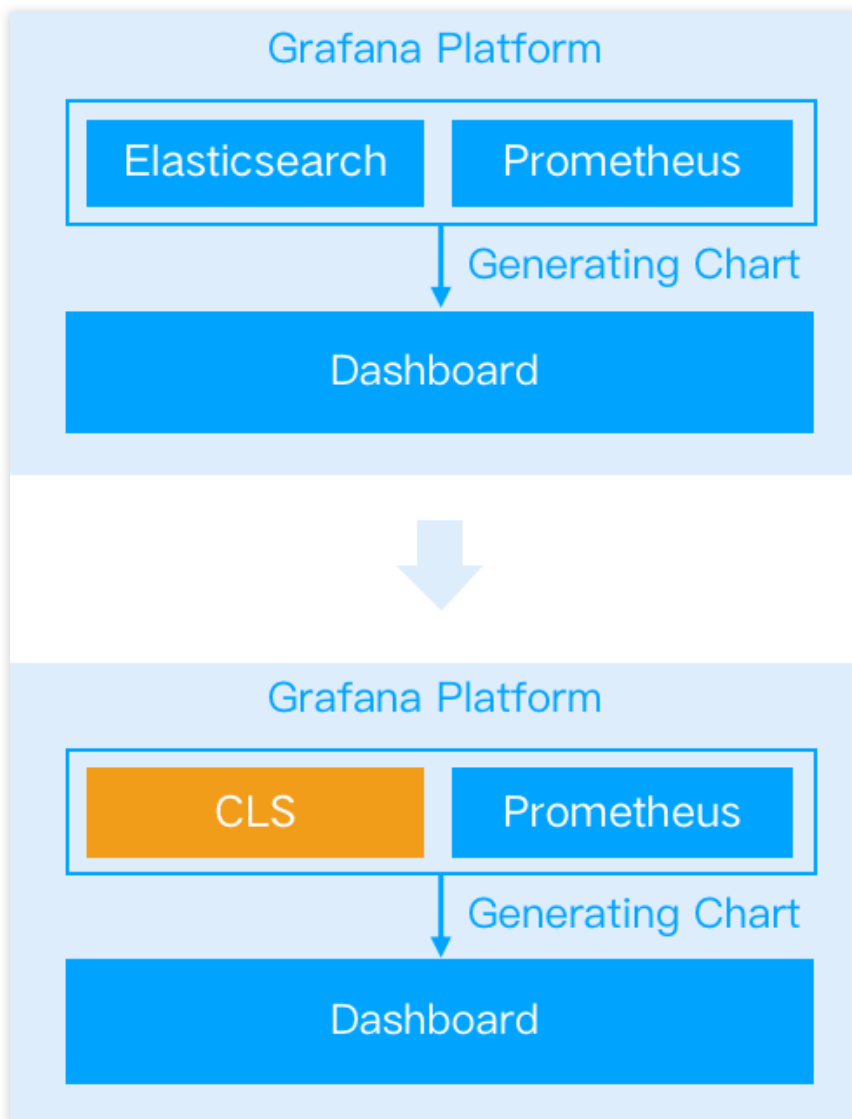
# 仪表盘

## 把 Grafana 的 ES 数据源迁移为 CLS 数据源

最近更新时间：2024-01-20 17:28:40

### 背景

在日志服务（Cloud Log Service, CLS）使用场景里，从其他日志工具迁移到 CLS 是非常常见的情况。其中，存在用户使用 Grafana 做可视化监控工具，例如 ES + Grafana 的组合。当数据源迁移到 CLS 后，用户依托 Grafana 制作的各种仪表盘资源，搭建的运维工具和平台就都失去了作用。为了避免重建这套体系，需要 CLS 对接 Grafana，替换 ES 数据源。



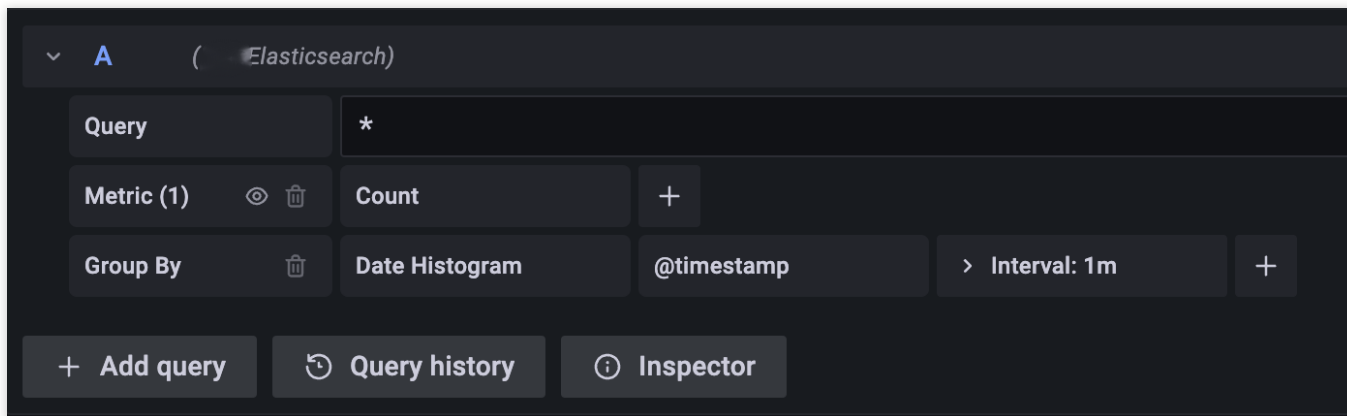
## 安装 CLS-Grafana 插件

CLS 数据源由腾讯云日志服务团队进行维护，已经通过 [官方签名认证](#)，可以在 Grafana 设置页面一键安装。具体接入步骤请参考 [CLS 对接 Grafana](#)。

## 使用 CLS 数据源替换 ES 数据源

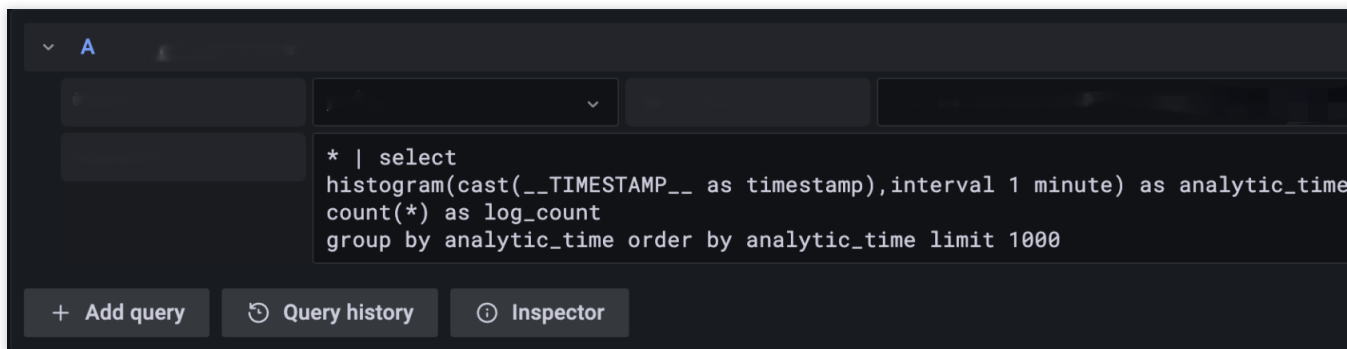
### 数据源配置区域对比

**ES 数据源**：查询语句界面分为顶部的 **Query输入区**和其余的**辅助输入功能区**。Query 输入区可输入 Lucene 语句，用于对日志进行过滤。辅助输入区通过单击填写，生成 DSL 内容，用于数据聚合，相当于 CLS 的 SQL。



**CLS 数据源**：查询语句界面分为**地域与日志主题**选择和**检索分析语句**两个部分。地域与日志主题选择模块可以快速进行日志主题切换，而检索分析语句则用于输入 CLS 查询语句。

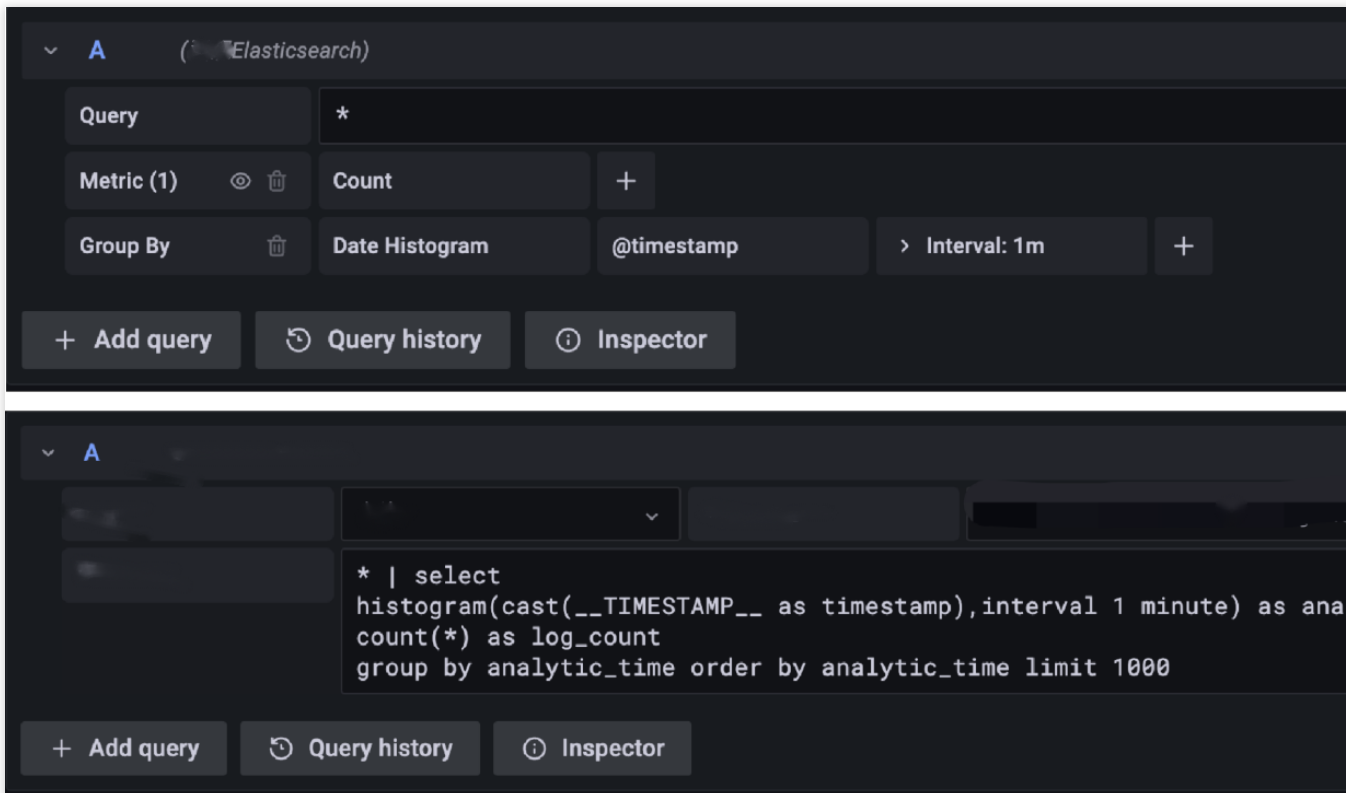
CLS 查询语句分为 Lucene 和 SQL 两个部分，两个部分之间使用管道符“|”进行分隔。其中 Lucene 部分和 ES 的 Query 输入区内容相同。SQL 输入内容除了支持标准的 SQL 语法外，还支持大量的 SQL 函数，SQL 区域内容和 ES 输入区的辅助输入模块完成对标。更多请参考的 [CLS 语法规则](#)。



### 操作实践

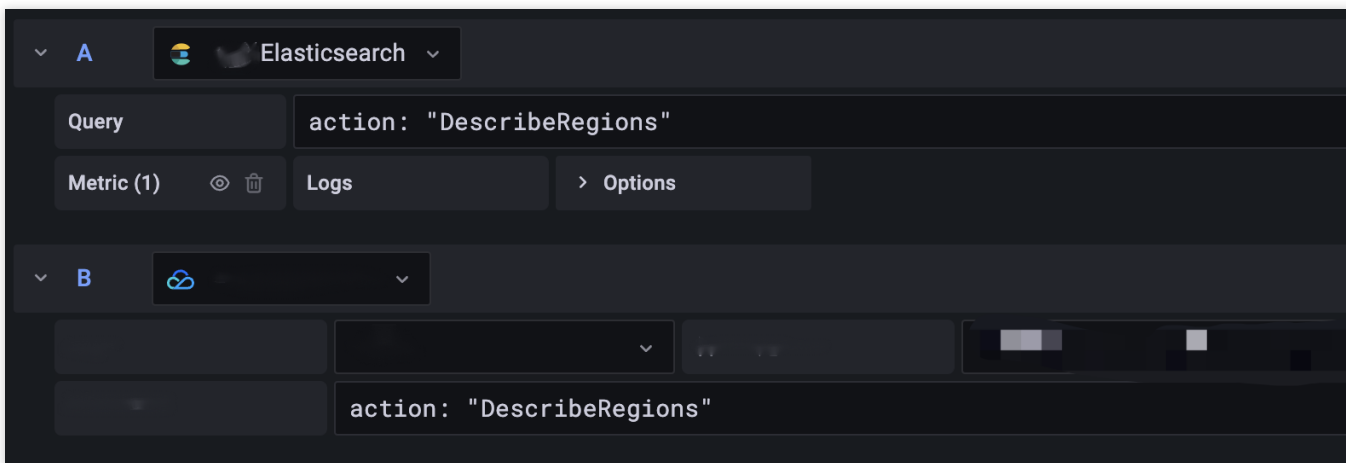
### 统计日志条数

对于想要绘制随时间变化的日志条数，ES 数据源将 Metric 选中 Count， GroupBy 选中 Histogram。CLS 的检索语句可以使用 Histogram 结合聚合函数 Count 完成。类似的，对于 Max、Min、Distinct 等其他 [通用聚合函数](#) 使用上也完全一致，直接将 Count 函数进行替换即可。



### 查看原始日志

想要直接查看符合条件的日志，ES 数据源需要将 Metric 选中 Logs 模式，而 CLS 只需要输入对于的 Lucene 语句即可。输入语句比对：



展示效果：

```

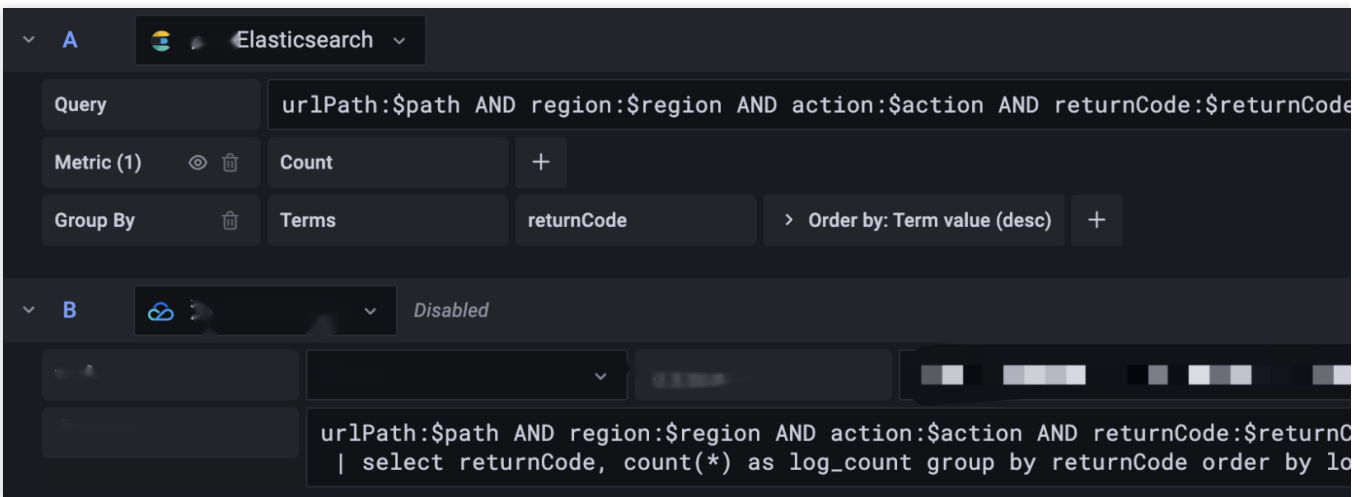
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
> {"logType":"clientlog","host":"VM-206-80-centos","referer":"https://console.cloud.tencent.com
    
```

### 聚合统计 - 错误码占比

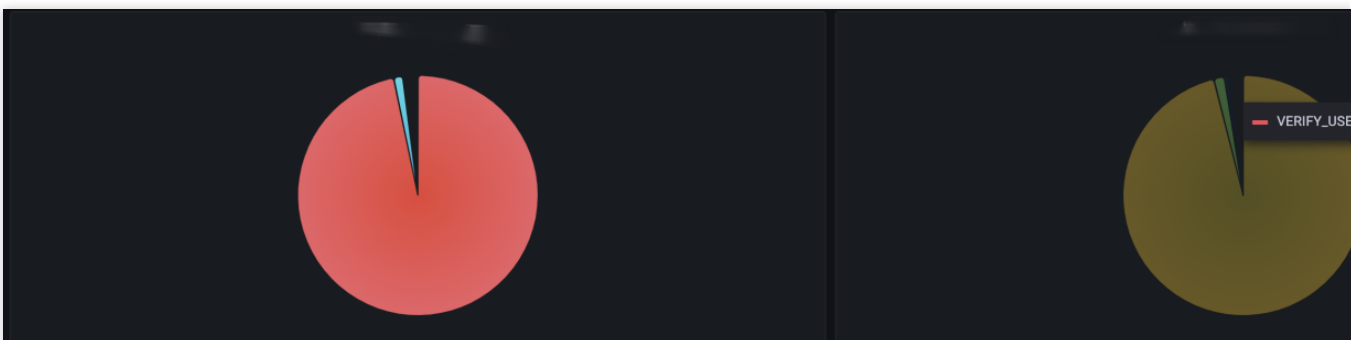
根据错误码进行聚合，展示各个错误码的日志数量。此处可以看到，语句中包含变量 `$path`。CLS 数据源插件进行了变量功能的相关适配，允许直接使用 Grafana 的变量能力。

#### 注意：

绘制饼图时右侧图表选项请选择 **ValueOptions-AllValues**。

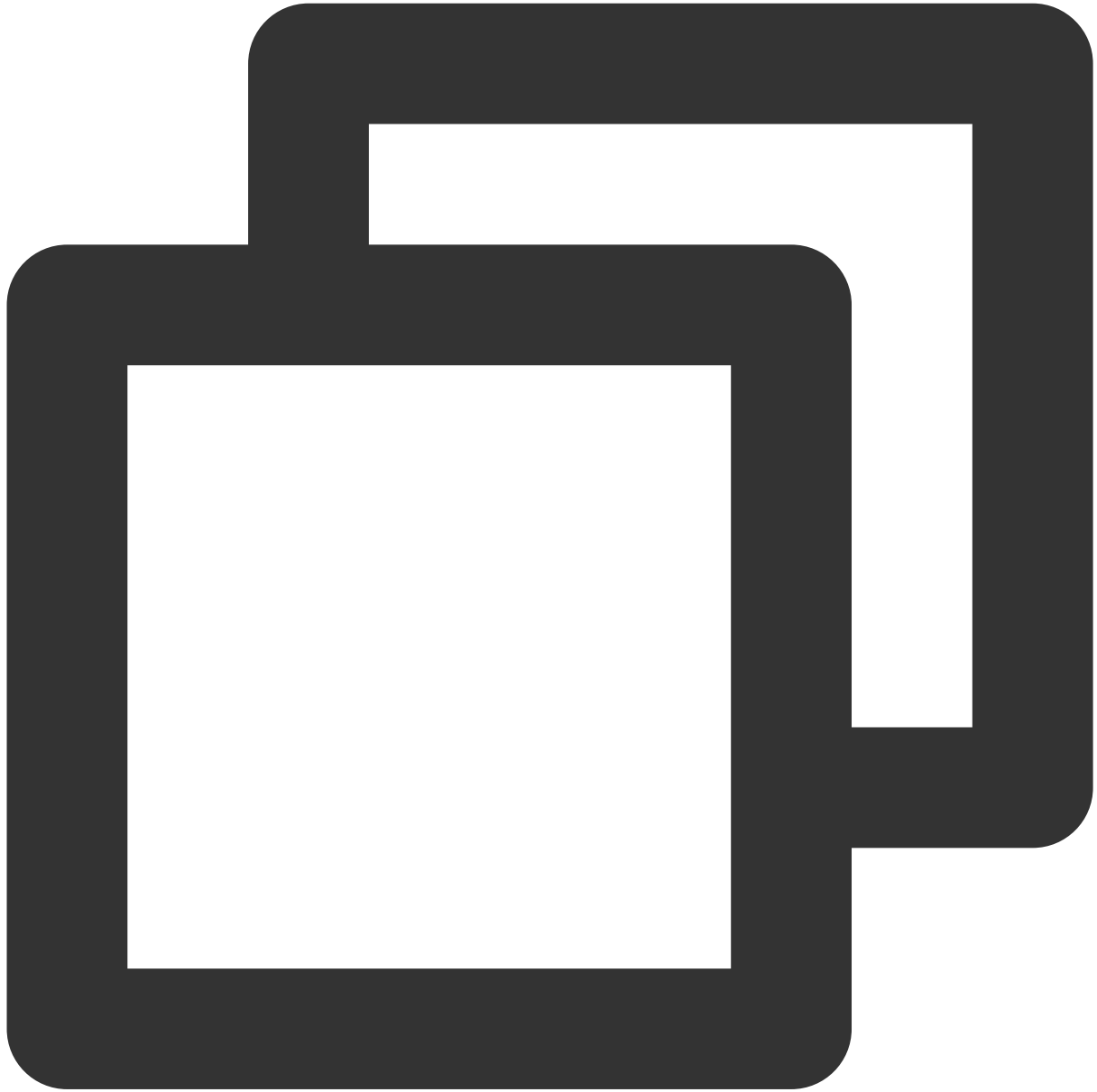


展示效果：

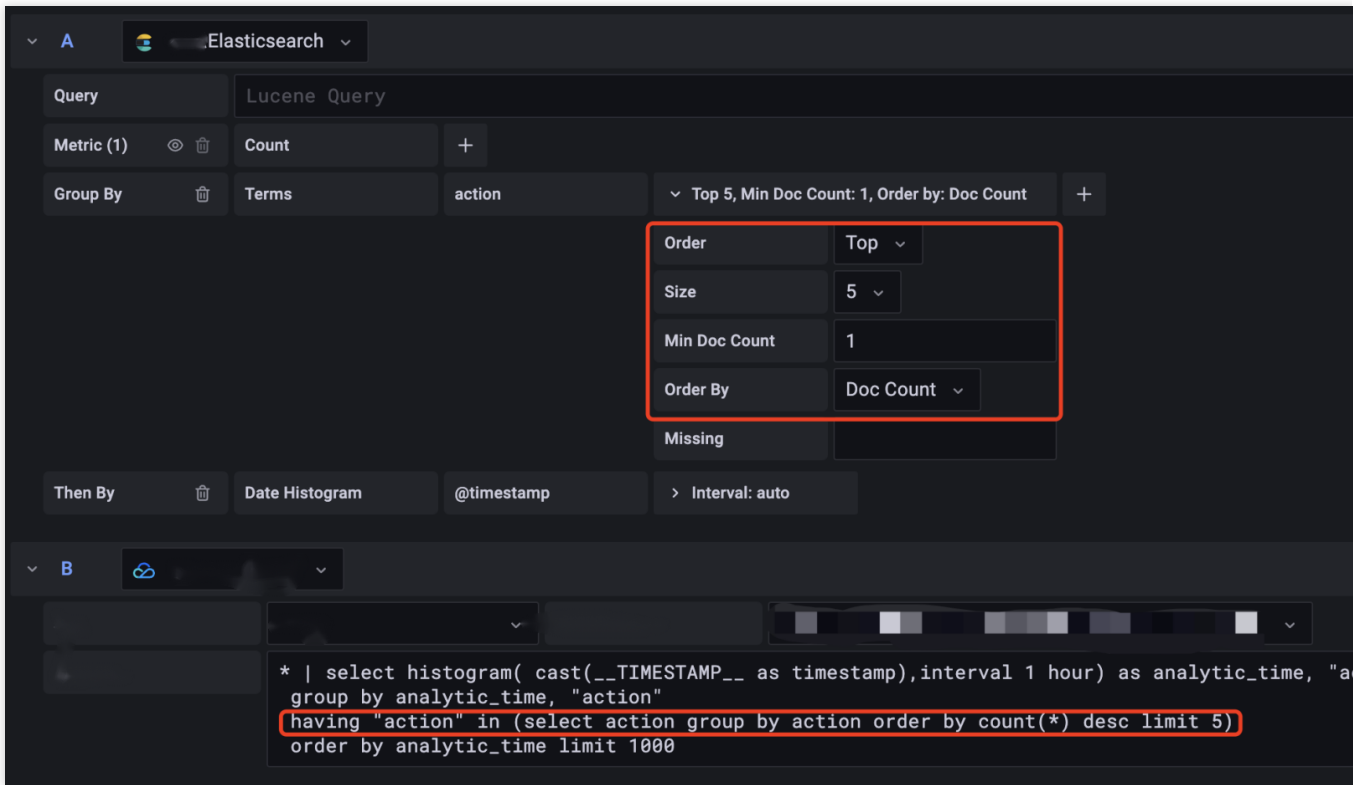


## 聚合统计 - Top5请求的数量变化情况

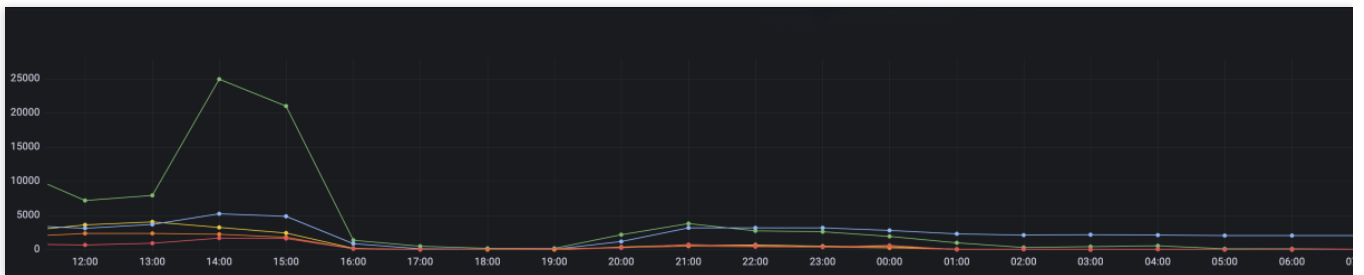
ES 数据源中，GroupBy 聚合选项允许填写 Size 值，支持选中出现频率最高的N个值，再进行聚合。  
此情况在 CLS 数据源 SQL 中，可以通过 having 语句搭配嵌套子查询实现。



```
*|select histogram( cast (__TIMESTAMP__ astimestamp),interval1hour)as analytic_time,
```



查询结果可以看到，图中共有5条曲线。

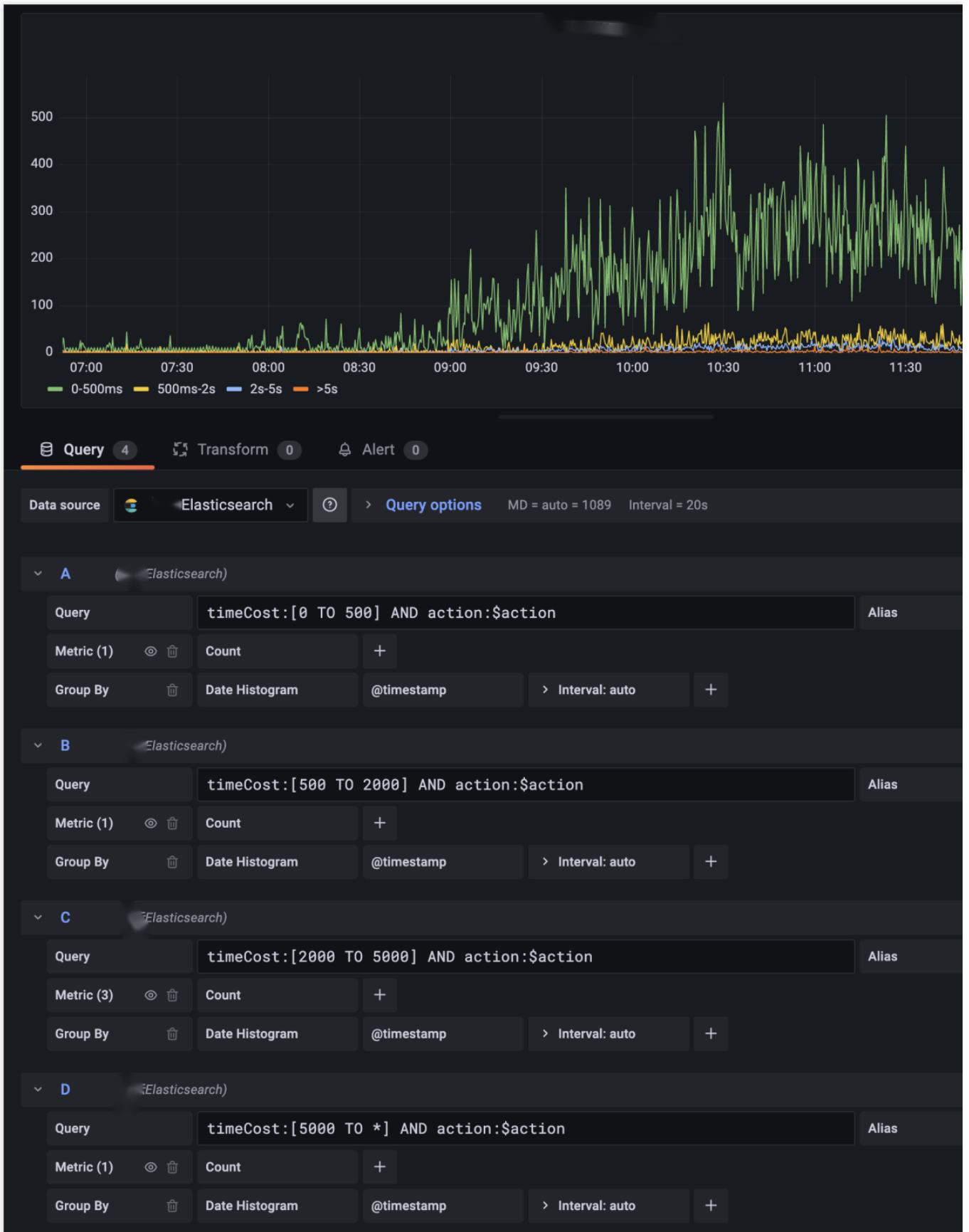


通过以上的语句搭配使用，已经可以满足大部分的检索分析场景。

### 统计接口耗时的分段情况

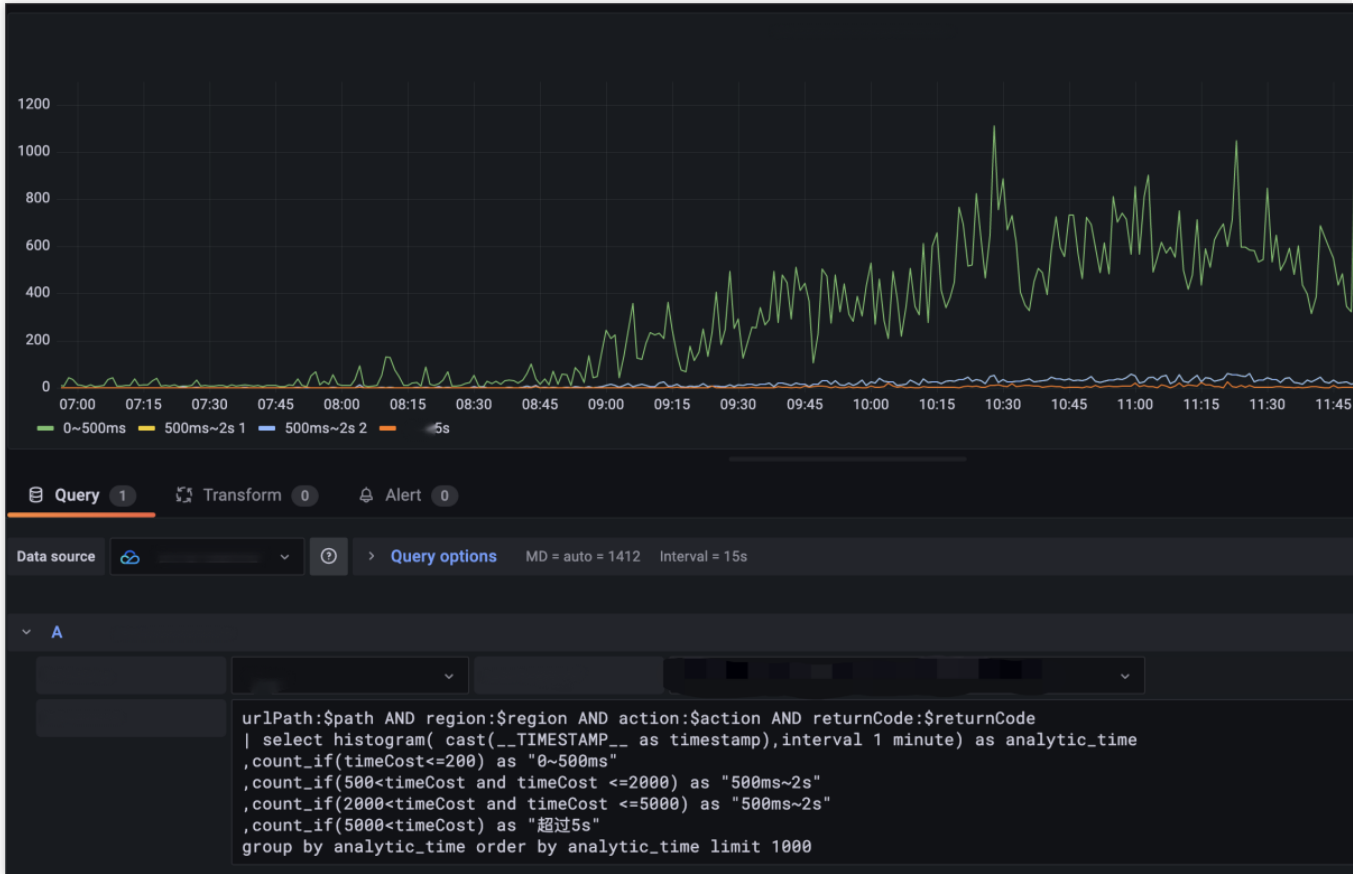
在 ES 数据源仪表盘中，有一个配置项繁多，但场景适用广的例子：根据不同的时间范围，绘制在这个时间范围的请求数量。

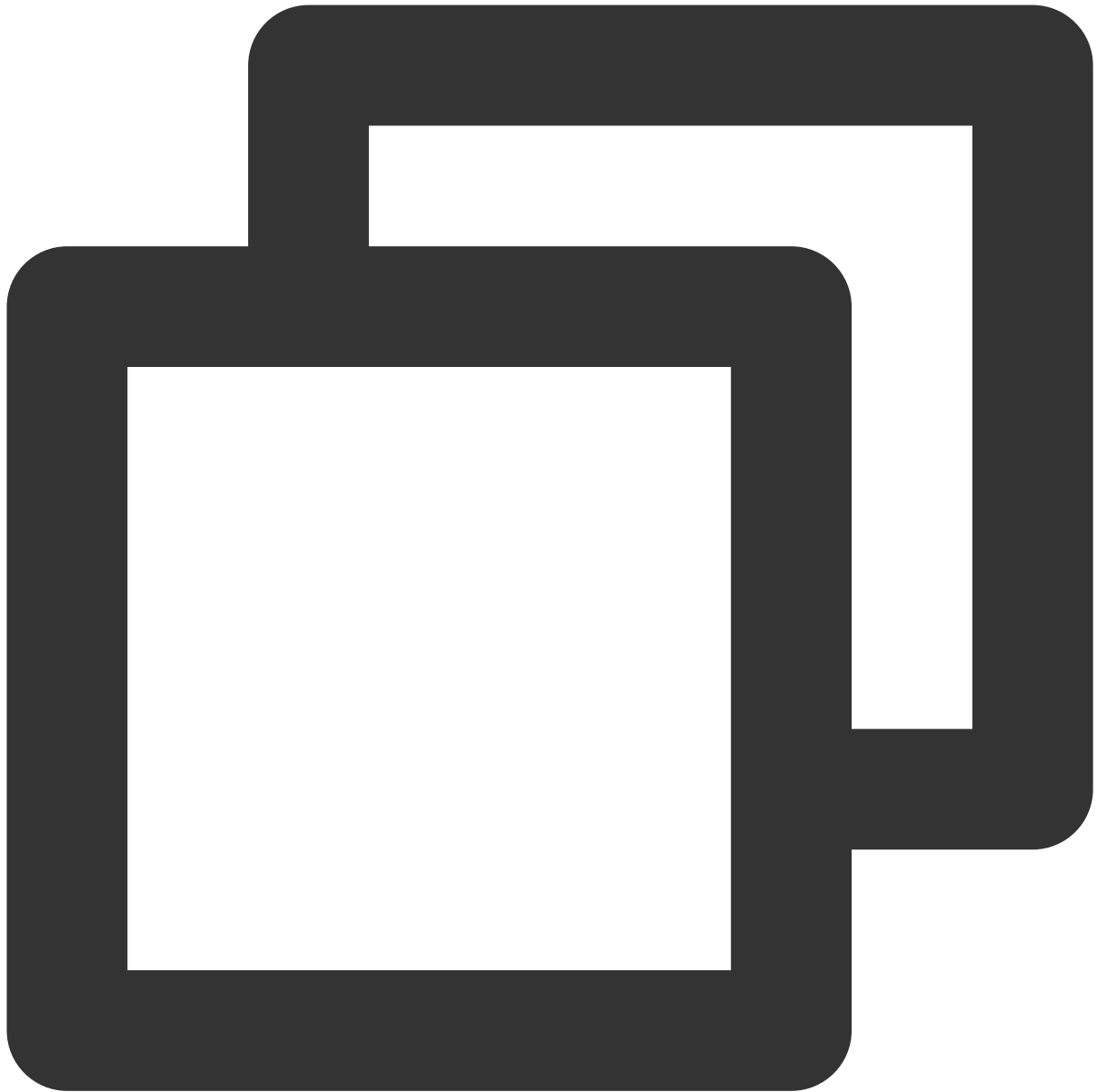
这个案例，统计了接口在0到500ms，500ms到2s，2s到5s，以及大于5秒的请求个数。



对应的，迁移到 CLS 数据源，也可以使用类似的多条语句进行绘制。但 CLS 本身的 SQL 能力更强，可以将相关的统计处理合并成一条 SQL 语句：

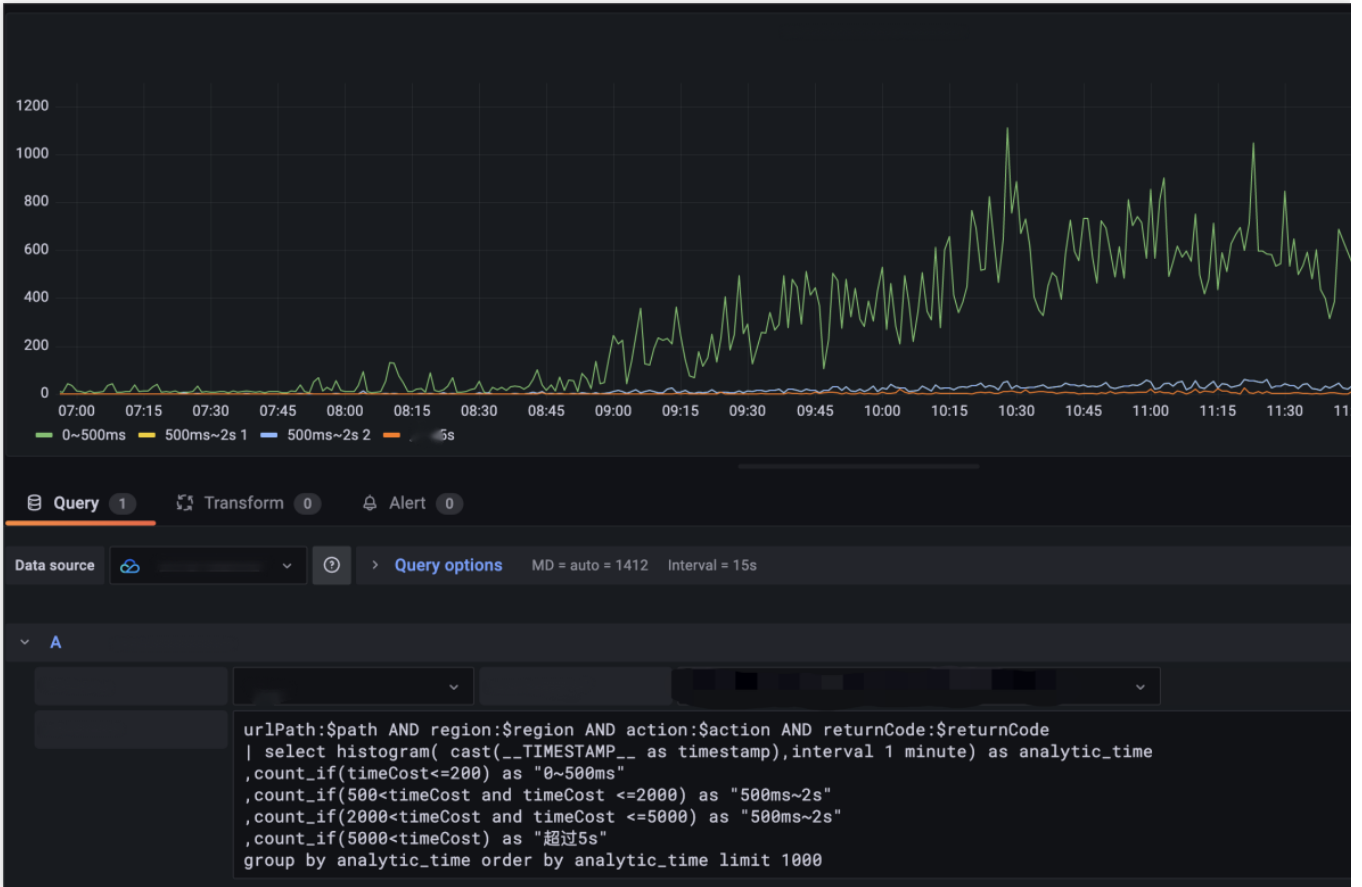


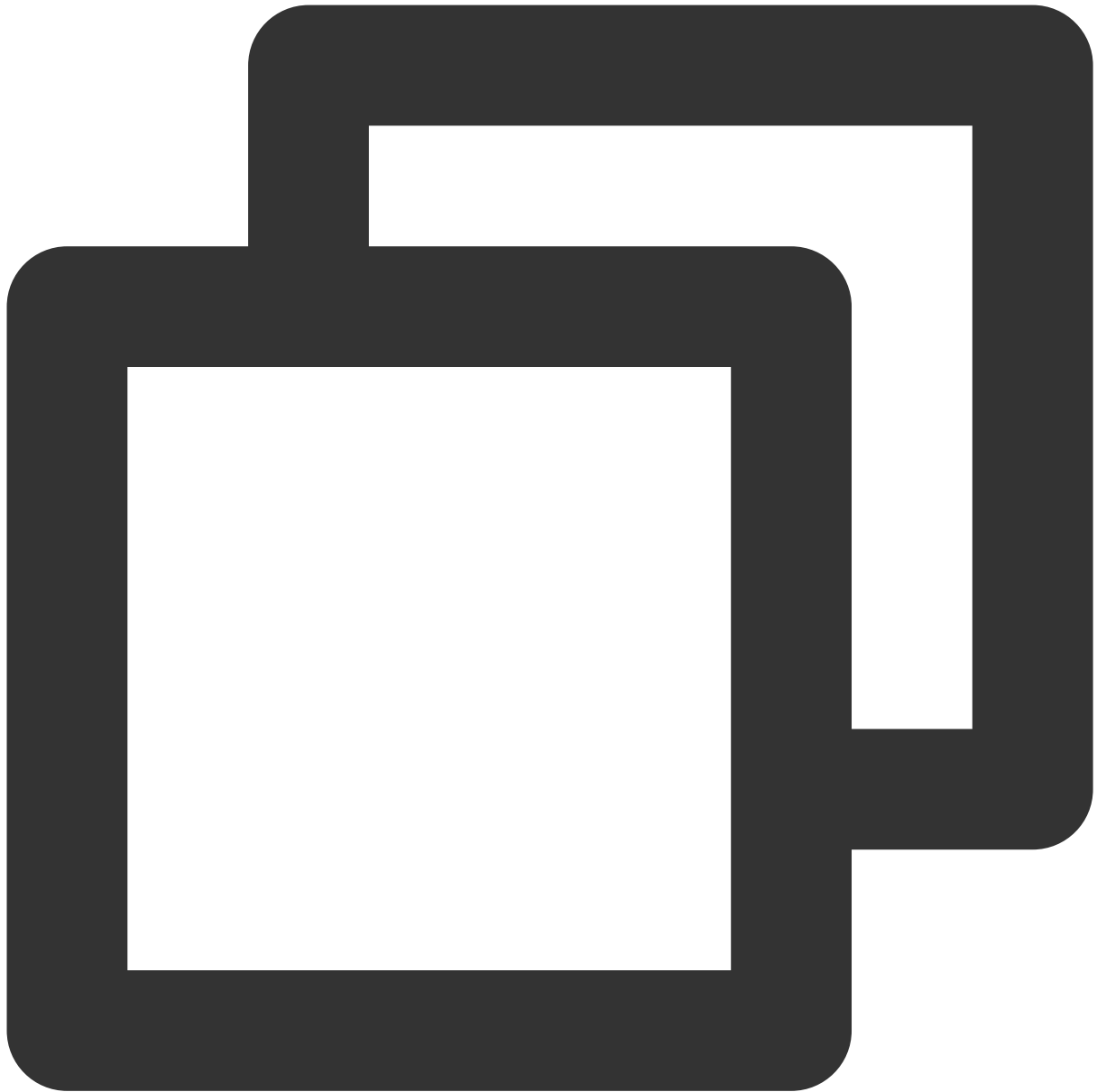




```
urlPath:$path AND region:$region AND action:$action AND returnCode:$returnCode | se
```

类似的场景，我们也可以写出使用估算函数 `approx_percentile` 分析得出的耗时相关情况。





```
urlPath:$path AND region:$region AND action:$action AND returnCode:$returnCode | s
```

### 模板变量能力

在以上的案例中，不同程度的出现了 Grafana 变量功能的身影。对于变量功能，Grafana 变量的类型种类繁多。常量类型、Textbox 输入框类型对各类数据源来说，是完全相同的，无需进行迁移。这里主要介绍如何迁移 Query 类型变量。

**ES 版本的 \$action 变量：**用于展示出现的接口种类，ES 数据源的版本使用 DSL 进行描述，语义上是找到符合 query 条件为 `urlPath:$path AND region:$region` 的内容，再选取 action 字段，并按照出现次数排序。

### General

Name	action	Type	Query
Label		Hide	
Description	descriptive text		

### Query Options

Data source	Elasticsearch	Refresh	On time range change
Query	{"find": "terms", "field": "action", "query": "(urlPath:\$path AND region:\$region)", "orderB", "min_doc_count":1}		
Regex	/.*(-(<text>.*)-(<value>.*)-.*/		
Sort	Disabled		

### Selection options

Multi-value	<input checked="" type="checkbox"/>
Include All option	<input checked="" type="checkbox"/>
Custom all value	*

**CLS 版本的 \$action 变量**：使用体验上与在图表编辑的输入行为上，保持一致。选择服务类型为日志服务并选中对应的日志主题后，输入 SQL 语句，即可达到相同效果。

### General

Name	action	Type	Query
Label		Hide	
Description	descriptive text		

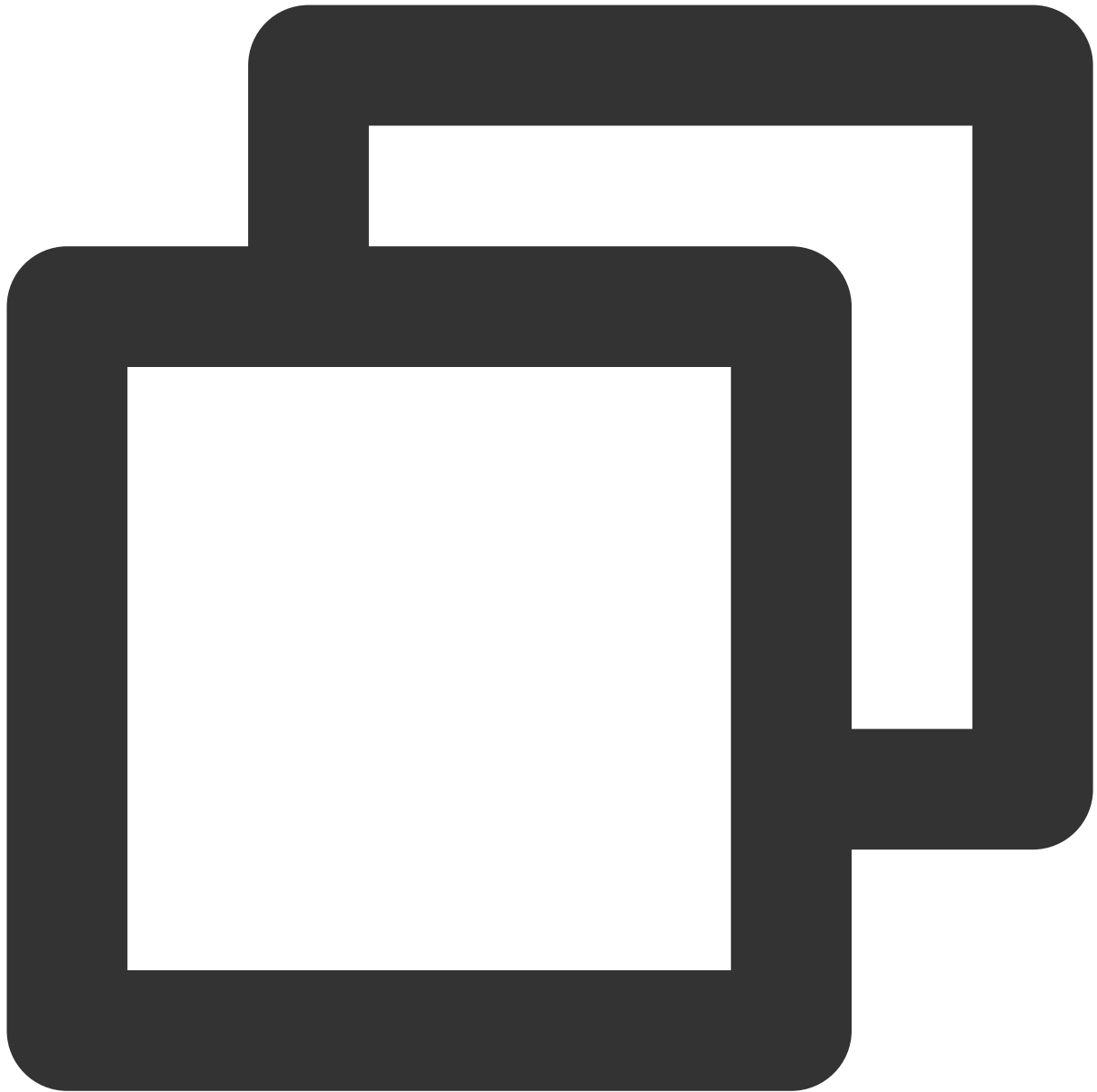
### Query Options

Data source	<span style="border: 1px solid #eee; padding: 2px;">▼</span>	Refresh	<span style="border: 1px solid #eee; padding: 2px;">i</span>	On time range change	<span style="border: 1px solid #eee; padding: 2px;">▼</span>
	[Blurred content]				
	[Blurred content]				
	[Blurred content]				
	[Blurred content]				
Regex	<span style="border: 1px solid #eee; padding: 2px;">i</span>	/.*(?<text>.*)(?<value>.*).*/			
Sort	<span style="border: 1px solid #eee; padding: 2px;">i</span>	Disabled <span style="border: 1px solid #eee; padding: 2px;">▼</span>			

### Selection options

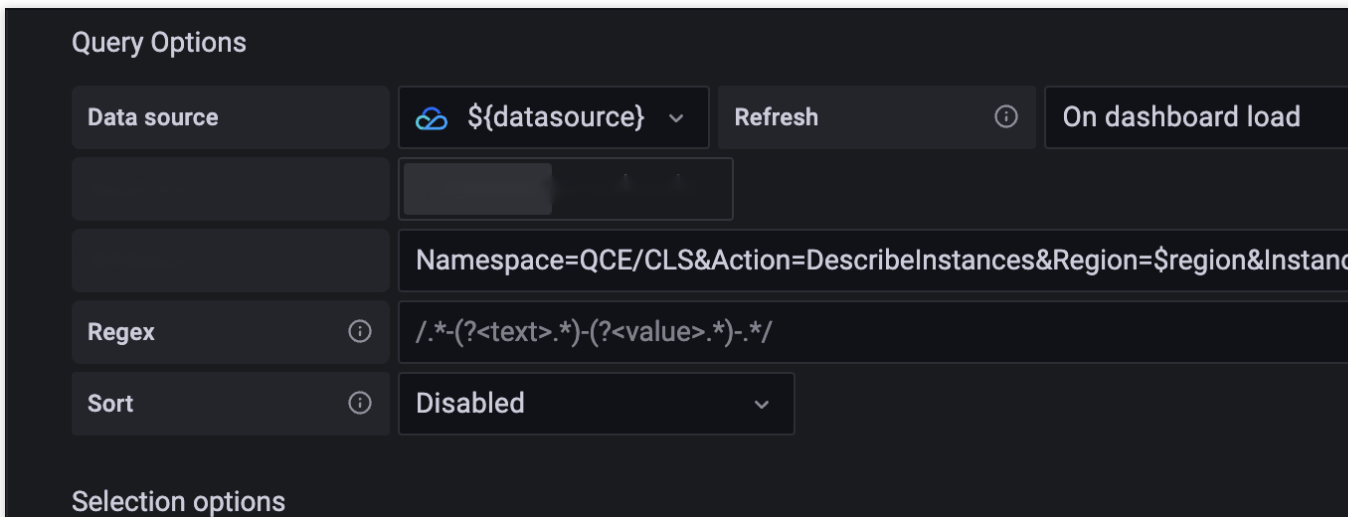
Multi-value	<span style="border: 1px solid #eee; padding: 2px;">i</span>	<input checked="" type="checkbox"/>
Include All option	<span style="border: 1px solid #eee; padding: 2px;">i</span>	<input checked="" type="checkbox"/>
Custom all value	<input type="text" value="*"/>	

除了使用 CLS 的检索语句进行变量查询，还可以使用云监控的资源查询功能，将腾讯云上的服务资源，作为列表内容进行展示。功能文档可查看 [云监控数据源模板变量功能](#)。如使用语句：



```
Namespace=QCE/CLS&Action=DescribeInstances&Region=$region&display=${TopicName}/${To
```

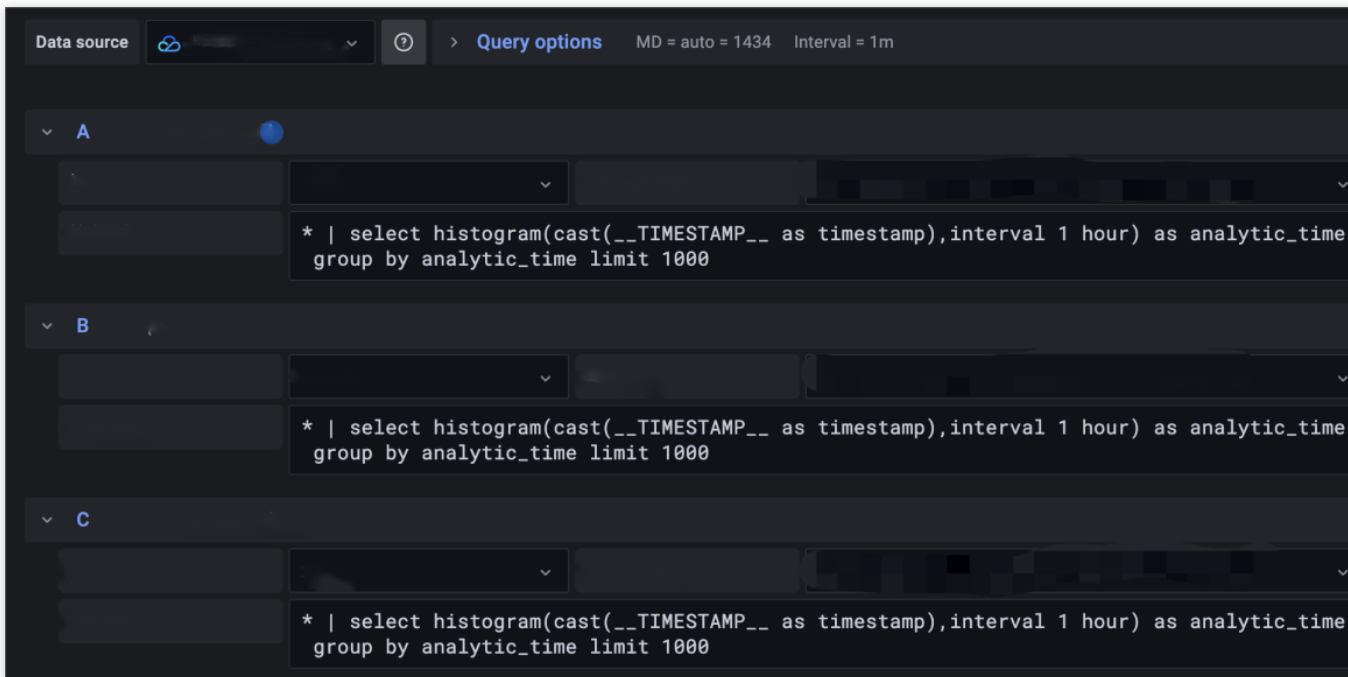
查询日志主题列表：



### 合并不同地域的请求数据内容

在原本的实现中，部分用户会存在所有数据都存储在同一台 ES 实例上的情况。在使用 CLS 之后，采用就近原则创建了多个日志主题。此时，用户可能会想要将多个日志主题内容合并到图表中。

对于3条来自不同地域的日志查询：



我们可以使用 Transform 模块，实现数据求和的效果，并选用需要的图表进行展示。



analytic_time	shanghai	guangzhou	beijing
2022-03-25 11:00:00	1316	1316	
2022-03-25 12:00:00	3600	3600	
2022-03-25 13:00:00	3600	3600	
2022-03-25 14:00:00	3600	3600	
2022-03-25 15:00:00	3600	3600	
2022-03-25 16:00:00	3600	3600	
2022-03-25 17:00:00	3600	3600	

Query 3 Transform 3

Outer join

Field name Choose

Add field from calculation

Mode Reduce row

Field name analytic\_time shanghai guangzhou beijing

Calculation Total

Alias Total

Replace all fields

Filter by name

Identifier Regular expression pattern analytic\_time shanghai guangzhou beijing Total

## 总结

对于存量的 ES 仪表盘，重复以上的迁移步骤，就可以将一个 ES 数据源的仪表盘，完全转化成为 CLS 数据源的仪表盘。

ES 到 CLS 数据源的迁移，可以让用户从自建 ELK 迁移到腾讯云日志服务后，积累的可视化资源得到继续的利用。转化之后的仪表盘，不仅在能力上完全对标ES数据源版本，还可以结合数据源插件的一些其他能力（腾讯云可观测平台模板变量），更好地与腾讯云生态进行融合。

## 监报告警

# 按时间段分别设置告警触发条件

最近更新时间：2024-01-20 17:28:40

## 简介

因为业务性质等原因，配置告警策略时，需要针对不同时间段分别设置不同的告警触发条件。例如：工作时间段（09点 - 18点）包含“error”的日志超过100条触发告警，非工作时间（19点 - 次日08点）包含“error”的日志超过10条触发告警。

## 配置方式

[配置告警策略](#) 过程中，填写如下执行语句及触发条件：

**执行语句1:**



```
error | select count(*) as error_count
```

统计包含“error”的日志条数。

**执行语句2:**



```
* | select hour(now()) as hour limit 1
```

使用 [日期和时间函数](#) 获取告警执行时刻的小时部分，即告警时属于几点。

**触发条件：**



```
($1.error_count>100 && $2.hour>=9 && $2.hour<=18 ) || ($1.error_count>10 && ($2.hour
```

使用 [触发条件表达式](#) 指定具体的告警触发条件及阈值。其中， `$1.error_count>100 && $2.hour>=9 && $2.hour<=18` 表示告警执行时间在09点 - 18点时， `error_count` 大于100才会触发告警； `$1.error_count>10 && ($2.hour<9 || $2.hour>18)` 表示告警时间在9点前及18点后（即19点 - 次日08点）时， `error_count` 大于10即会触发告警。

# 使用同环比作为告警触发条件

最近更新时间：2024-01-20 17:28:40

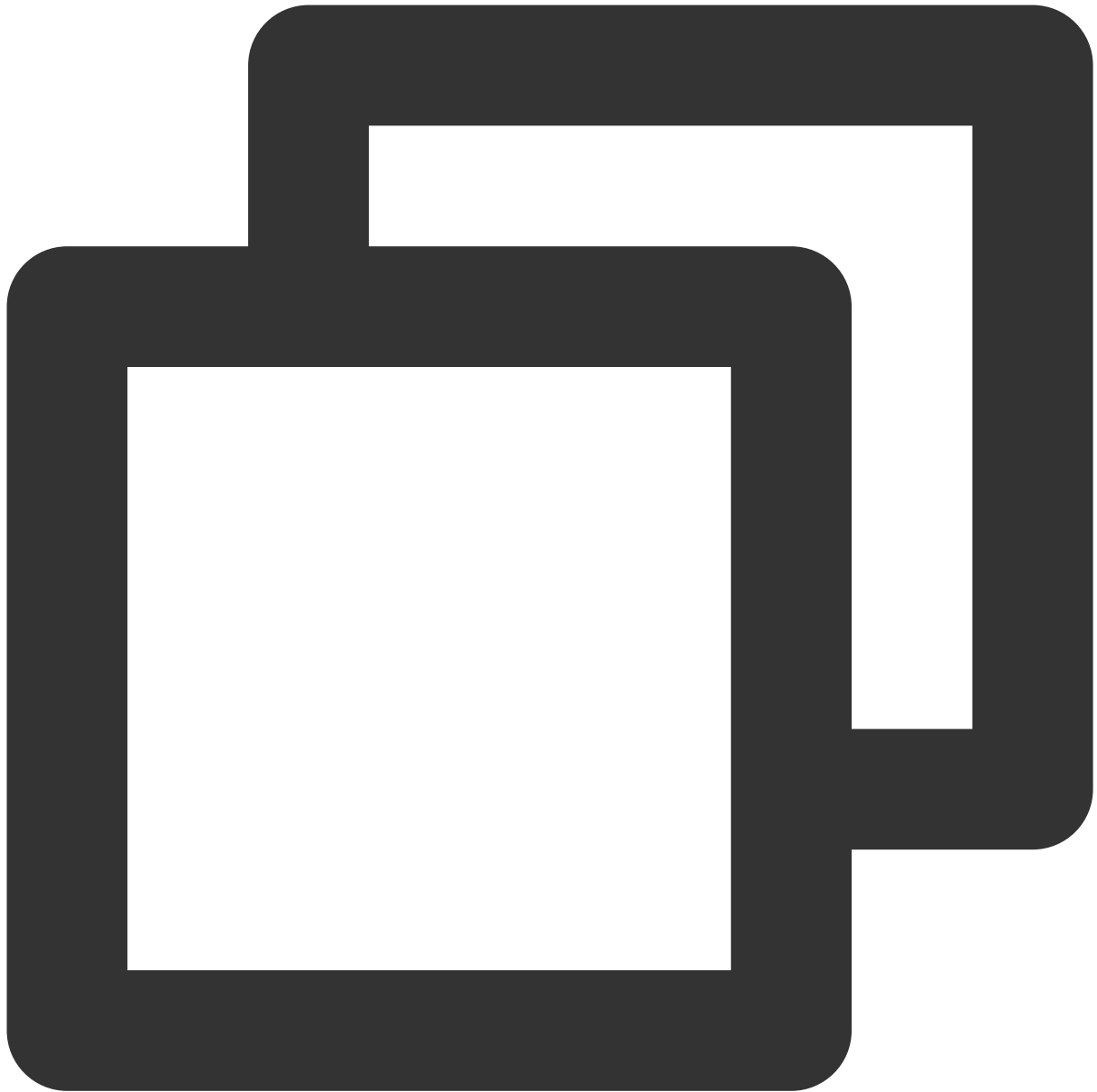
## 简介

设定告警触发条件时，由于业务特点，常需要对指标进行环比，变化超过一定阈值再触发告警。例如接口响应时间相比昨天同时间段上升超过50%即触发告警。

## 配置方式

[配置告警策略](#) 过程中，填写如下执行语句及触发条件：

**执行语句:**



```
* | select
  round(compare[3], 4) as ratio,
  compare[1] as current_avg_request_time,
  compare[2] as yesterday_avg_request_time
from
  (
    select compare(avg_request_time, 86400) as compare
    from
      (
        select avg("request_time") as avg_request_time
      )
  )
```

```
)
```

上述语句的执行结果中：

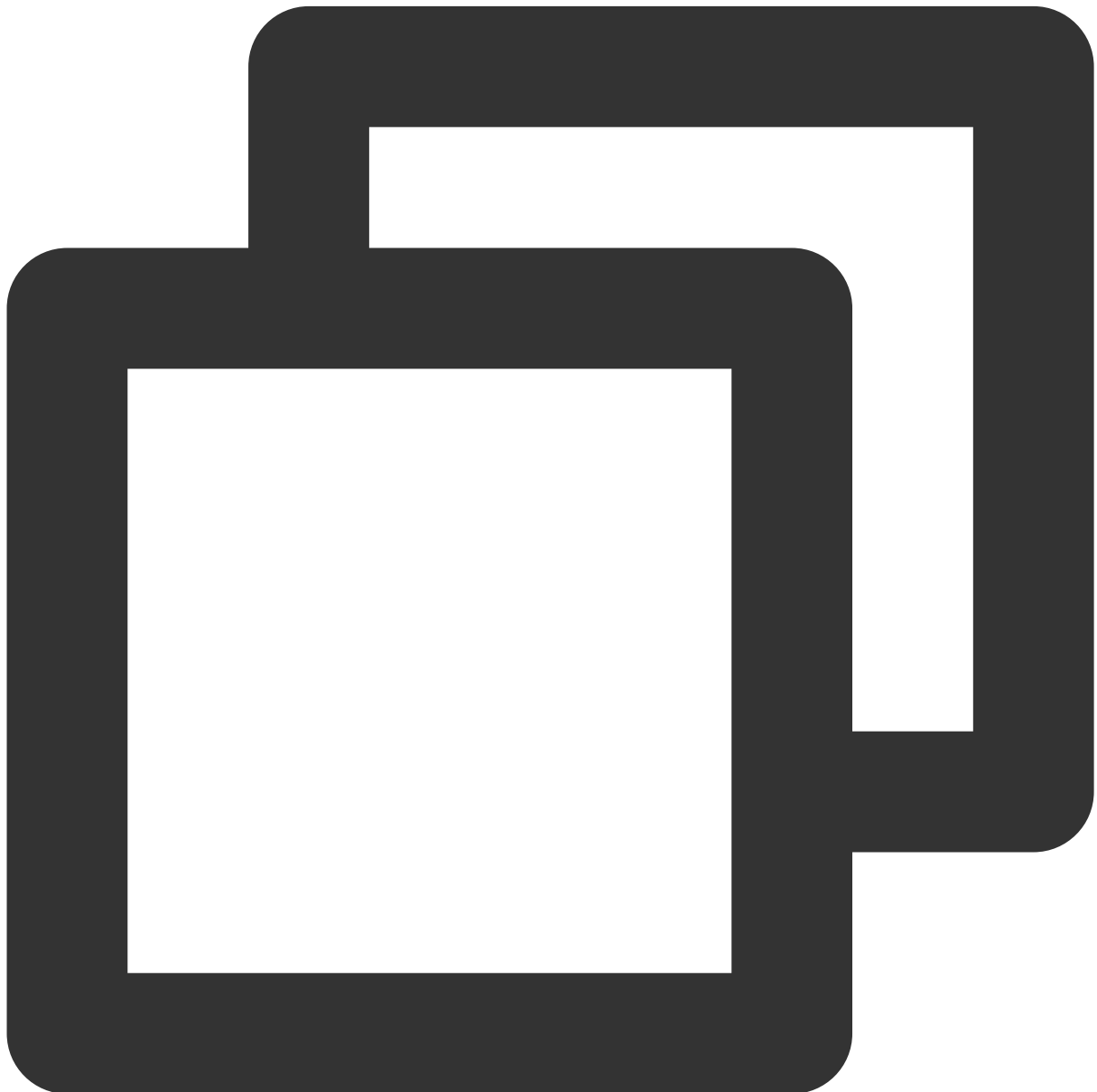
`ratio`代表：当前接口平均响应时间相比昨天（即86400秒前）的比值

`current_avg_request_time`代表：当前接口平均响应时间

`yesterday_avg_request_time`代表：昨天同时间段接口平均响应时间

上述语句中主要使用了`compare`函数，详细说明参见 [同环比函数](#)。

**触发条件：**



```
$1.ratio > 1.5
```



---

ratio大于1.5即触发告警，即相比昨天上升超过50%。

# 投递和消费

## 使用 Flink 消费 CLS 日志

最近更新时间：2024-01-20 17:28:40

本文详细描述了如何使用 Flink 实时消费 CLS 日志，使用 Flink-sql 分析 Nginx 日志数据，计算 Web 端的 PV/UV 值，并将结果数据实时写入到自建的数据库 MySQL 数据库。

文中使用的组件/应用及版本如下：

技术组件	版本
Nginx	1.22
CLS 日志服务	-
Java	openjdk version "1.8.0_232"
Scala	2.11.12
Flink sql	flink-1.14.5
MySQL	5.7

## 操作步骤

### 步骤1：安装腾讯云 Nginx 网关

1. 购买腾讯云主机 CVM，请参考 [通过购买页创建实例](#)。
2. Nginx 安装，请参考 [LINUX安装nginx详细步骤](#)。
3. 成功通过浏览器访问 nginx，并可下图说明安装成功：

# Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to [nginx.org](https://nginx.org).  
Commercial support is available at [nginx.com](https://nginx.com).

*Thank you for using nginx.*

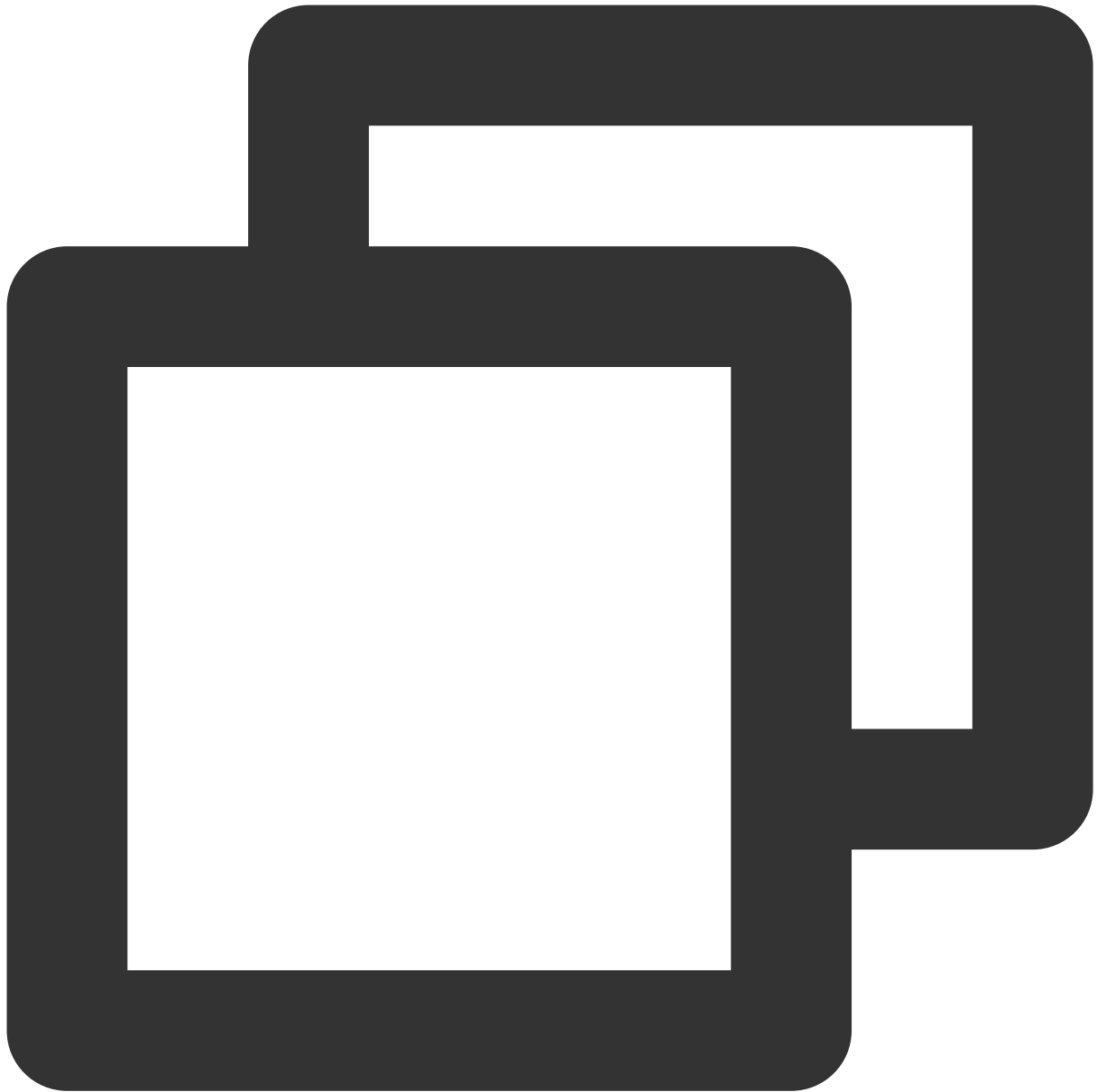
## 步骤2：采集 Nginx 日志到腾讯云 CLS 日志服务

1. 配置 [Nginx 日志采集](#)。
2. CLS 日志服务采集终端 [Loglistener的安装](#)，Loglistener 类似于开源组件 Beats，用来采集日志数据的 Agent。
3. 日志主题开启索引后，可以正常查询到 Nginx 的日志数据，如下图所示：
4. 最后，在 CLS 控制台 [开启 kafka 消费](#)，使用 Kafka 协议消费功能，您可以将一个日志主题，当作一个 Kafka Topic 来消费。本文就是使用流计算框架 Flink，实时消费 Nginx 日志数据，将实时计算的结果写入到 MySQL。

## 步骤3：搭建 MySQL 数据库

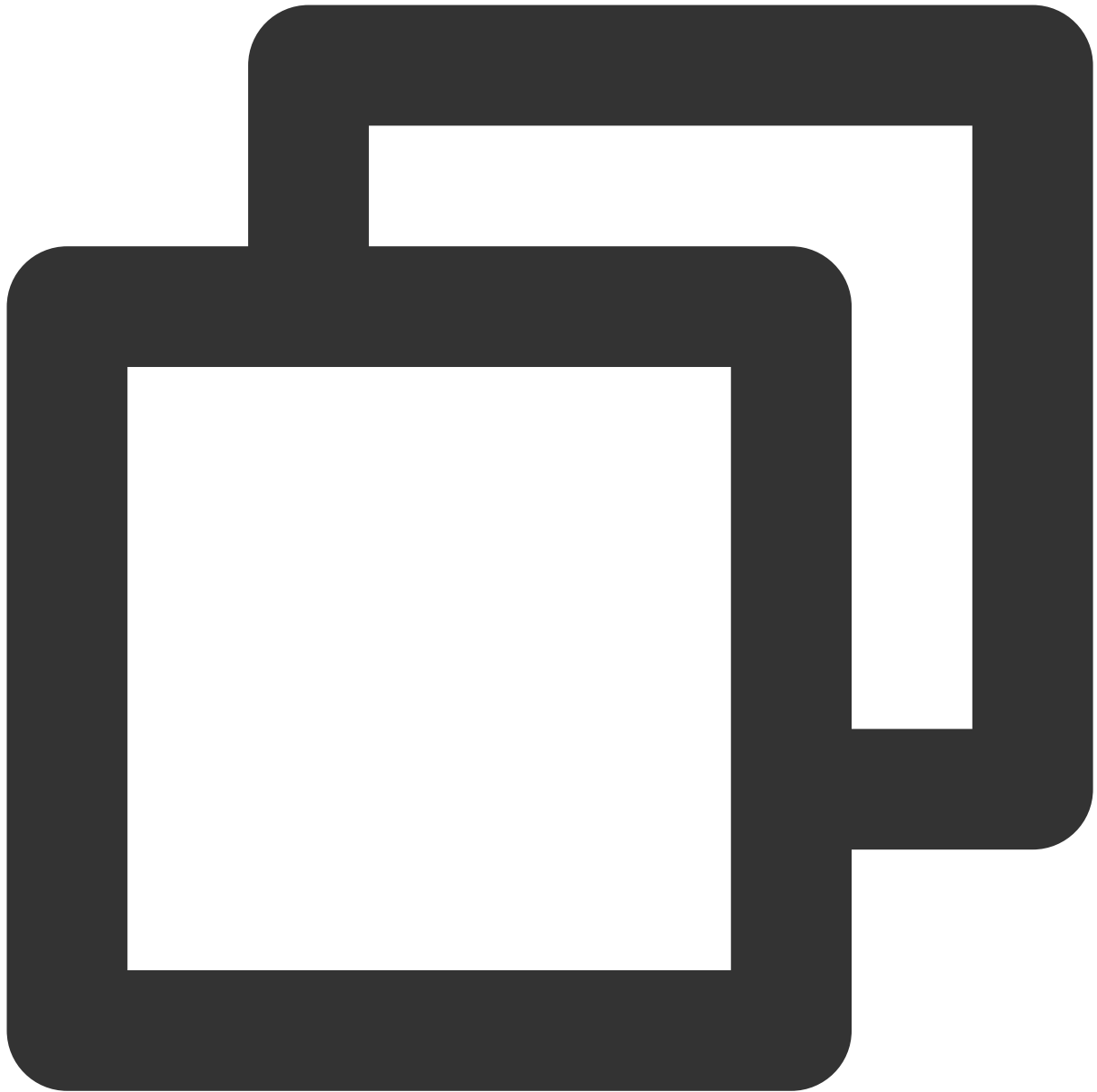
参考文档：[创建 MySQL 实例](#)

1. 登录数据库：



```
mysql -h 172.16.1.1 -uroot
```

2. 新建需要使用的 database 和表，例子中的 database 名为 `flink_nginx`，表名为 `mysql_dest`。



```
create database if not exists flink_nginx;
create table if not exists mysql_dest (
  ts timestamp,
  pv bigint,
  uv bigint
);
```

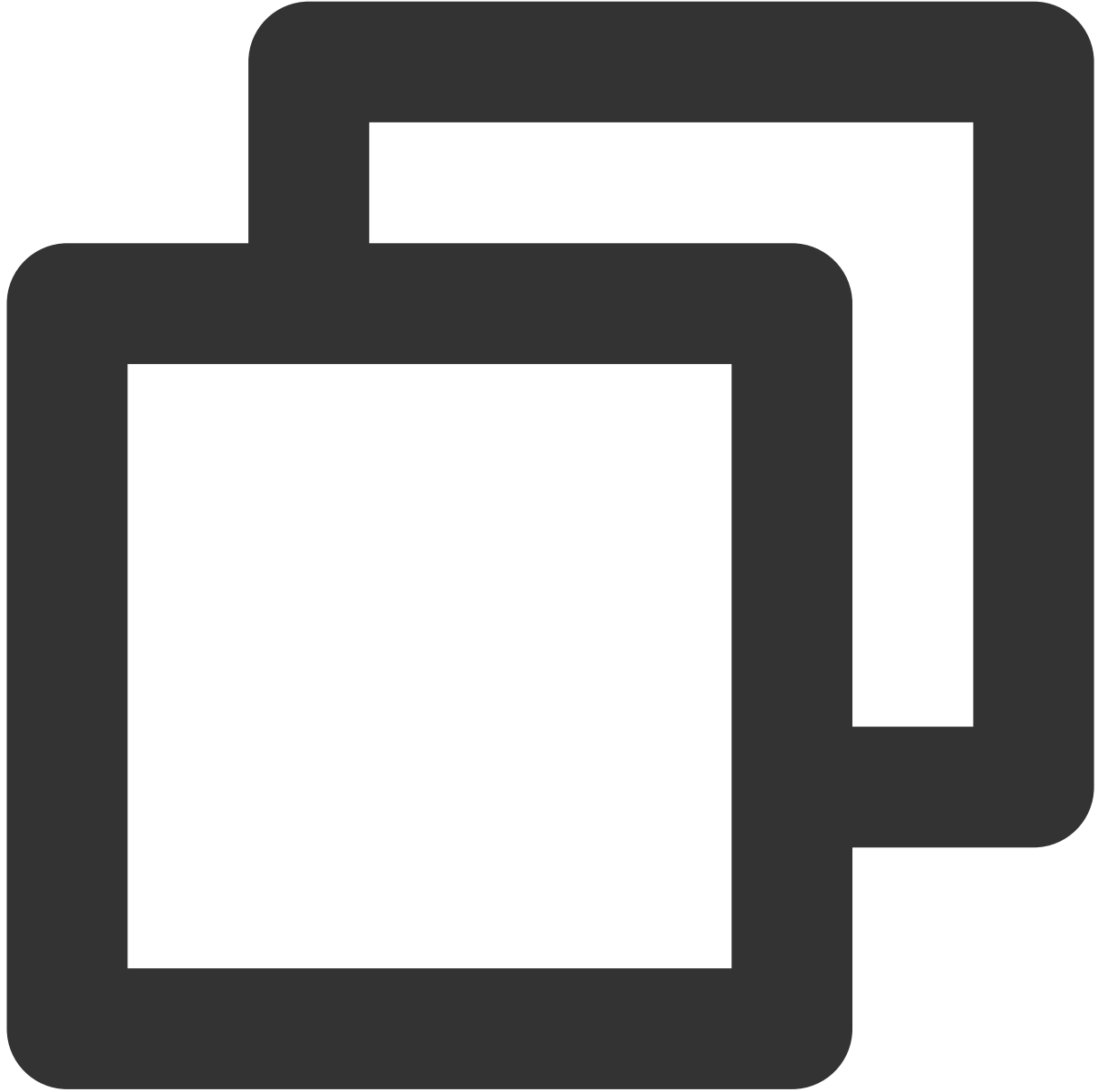
#### 步骤4：部署 Flink

1. 部署 Flink 时，建议使用如下版本，否则可能会安装不成功。

[购买腾讯云主机 CVM](#)

[安装Scala 2.11.12](#)

2. 安装 Flink 1.14.15，并进入 SQL 界面，从 [Apache Flink 官网](#) 下载 Flink 二进制代码包并开始安装。



```
# 解压缩 Flink 二进制包
tar -xf flink-1.14.5-bin-scala_2.11.tgz
cd flink-1.14.5

# 下载 kafka 相关依赖
wget https://repo1.maven.org/maven2/org/apache/flink/flink-connector-kafka_2.11/1.1
mv flink-connector-kafka_2.11-1.14.5.jar lib
```

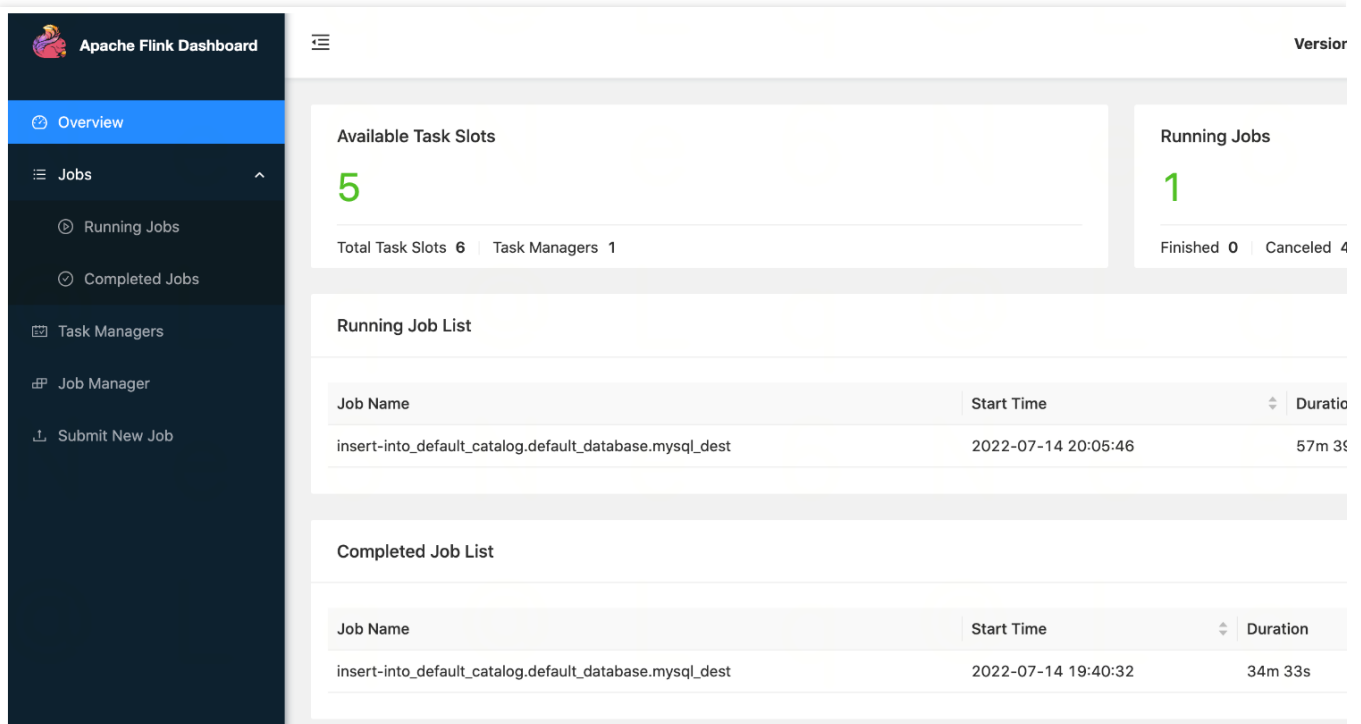
```
wget https://repo1.maven.org/maven2/org/apache/kafka/kafka-clients/2.4.1/kafka-clients-2.4.1.jar lib

# 下载 MySQL 相关依赖
wget https://repo1.maven.org/maven2/org/apache/flink/flink-connector-jdbc_2.11/1.14.5/flink-connector-jdbc_2.11-1.14.5.jar lib
wget https://repo1.maven.org/maven2/mysql/mysql-connector-java/8.0.11/mysql-connector-java-8.0.11.jar lib
wget https://repo1.maven.org/maven2/org/apache/flink/flink-table-common/1.14.5/flink-table-common-1.14.5.jar lib

# 启动 Flink
bin/start-cluster.sh
bin/sql-client.sh
```

3. 当出现以下画面则说明安装成功。注意默认的网页端口是8081。

```
[root@vm-14 ~]# cd ~/flink-1.14.5 && bin/start-cluster.sh
Starting cluster.
Starting standalone session daemon on host VM-1
```



The screenshot shows the Apache Flink Dashboard interface. On the left is a navigation sidebar with options like Overview, Jobs, Running Jobs, Completed Jobs, Task Managers, Job Manager, and Submit New Job. The main content area displays:

- Available Task Slots:** 5 (Total Task Slots: 6, Task Managers: 1)
- Running Jobs:** 1 (Finished: 0, Canceled: 4)
- Running Job List:** A table with columns Job Name, Start Time, and Duration. One job is listed: 'insert-into\_default\_catalog.default\_database.mysql\_dest' starting at 2022-07-14 20:05:46 with a duration of 57m 35s.
- Completed Job List:** A table with columns Job Name, Start Time, and Duration. One job is listed: 'insert-into\_default\_catalog.default\_database.mysql\_dest' starting at 2022-07-14 19:40:32 with a duration of 34m 33s.

```
~/flink-1.14.5]$ bin/sql-client.sh

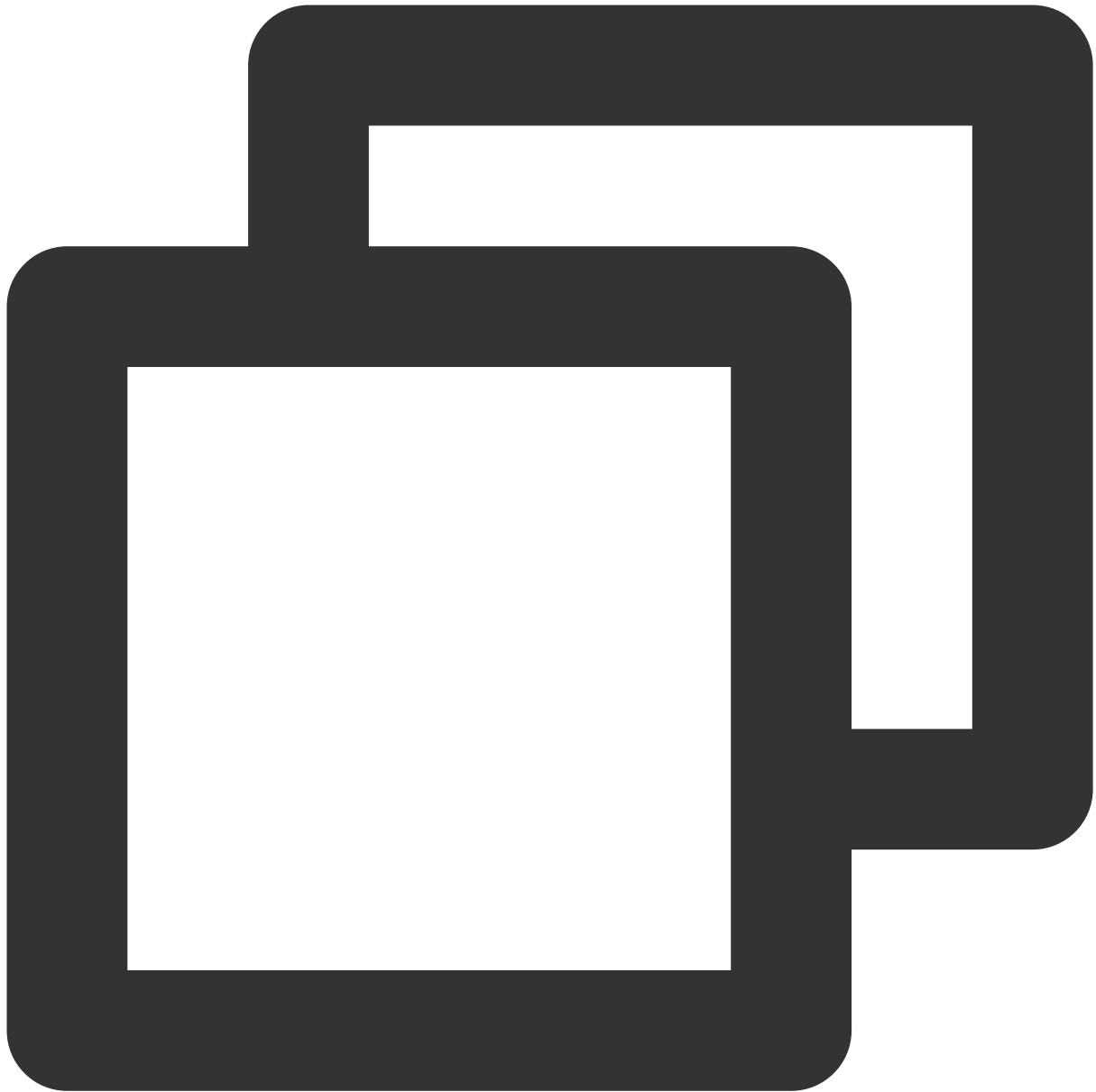
Command history file path: /data/home/llongzhu/.flink-sql-history

                                BETA
Flink SQL Client
Welcome! Enter 'HELP;' to list all available commands, 'QUIT;' to exit.
```

### 步骤5：使用 Flink 消费 CLS 日志数据

1. 在 SQL Client 界面中，执行如下 SQL：





```
-- 建数据源表消费 kafka 数据
CREATE TABLE `nginx_source`
(
  `remote_user` STRING,           -- 日志中字段, 客户端名称
  `time_local` STRING,           -- 日志中字段, 服务器本地时间
  `body_bytes_sent` BIGINT,      -- 日志中字段, 发送给客户端的字节数
  `http_x_forwarded_for` STRING, -- 日志中字段, 当前端有代理服务器时, 记录客户端真实 IP 地址
  `remote_addr` STRING,         -- 日志中字段, 客户端 IP 地址
  `protocol` STRING,            -- 日志中字段, 协议类型
  `status` INT,                 -- 日志中字段, HTTP 请求状态码
  `url` STRING,                 -- 日志中字段, url 地址
)
```

```

`http_referer` STRING,          -- 日志中字段, 访问来源的页面链接地址
`http_user_agent` STRING,       -- 日志中字段, 客户端浏览器信息
`method` STRING,                -- 日志中字段, HTTP 请求方法
`partition_id` BIGINT METADATA FROM 'partition' VIRTUAL,    -- kafka分区
`ts` AS PROCTIME()
) WITH (
    'connector' = 'kafka',
    'topic' = 'YourTopic', -- cls kafka协议消费控制台给出的主题名称, 例如out-633a268c-XX
    'properties.bootstrap.servers' = 'kafkaconsumer-ap-guangzhou.cls.tencentcs.com:90
    'properties.group.id' = 'kafka_flink', -- kafka 消费组名称
    'scan.startup.mode' = 'earliest-offset',
    'format' = 'json',
    'json.fail-on-missing-field' = 'false',
    'json.ignore-parse-errors' = 'true' ,
    'properties.sasl.jaas.config' = 'org.apache.kafka.common.security.plain.PlainLogi
    'properties.security.protocol' = 'SASL_PLAINTEXT',
    'properties.sasl.mechanism' = 'PLAIN'

);

--- 建立目标表, 写入mysql
CREATE TABLE `mysql_dest`
(
    `ts` TIMESTAMP,
    `pv` BIGINT,
    `uv` BIGINT
) WITH (
    'connector' = 'jdbc',
    'url' = 'jdbc:mysql://11.150.2.1:3306/flink_nginx?&serverTimezone=Asia/Shan
    'username' = 'username', -- mysql账号
    'password' = 'password', -- mysql密码
    'table-name' = 'mysql_dest' -- mysql表名
);

--- 查询 kafka 数据源表, 计算后写入 mysql 目标表
INSERT INTO mysql_dest (ts,uv,pv)
SELECT TUMBLE_START(ts, INTERVAL '1' MINUTE) start_ts, COUNT(DISTINCT remote_addr)
FROM nginx_source
GROUP BY TUMBLE(ts, INTERVAL '1' MINUTE);
    
```

2. 在 Flink 的任务监控页, 我们可以看到任务的监控数据:

The screenshot shows the Apache Flink Dashboard interface. The job name is `insert-into_default_catalog.default_database.mysql_dest` and it is in a **RUNNING** state with 2 tasks. The job ID is `2cb9060f4816b5f48f200ca82976bda8`, started at `2022-07-14 20:05:46`, and has a duration of `1h 17m 47s`. The dashboard includes a sidebar with navigation options like Overview, Jobs, Task Managers, and Job Manager. The main area displays a task diagram with a source task and an aggregate task connected by a GLOBAL arrow. Below the diagram is a table of task metrics:

Name	Status	Bytes Received	Records Received	Bytes Sent
Source: KafkaSource-default_catalog.default_database.nginx_source	RUNNING	7.23 MB	0	5.13 MB
GroupWindowAggregate(window=[TumblingGroupWindow('w\$', \$...)	RUNNING	5.13 MB	25,737	0 B

3. 进入 MySQL 数据库，即可看到计算 PV、UV 的结果数据实时写入：

```
MySQL [flink_nginx]> select * from mysql_dest;
+-----+-----+-----+
| ts          | pv  | uv  |
+-----+-----+-----+
| 2022-07-14 20:16:00 | 60  | 1   |
| 2022-07-14 20:17:00 | 62  | 2   |
| 2022-07-14 20:18:00 | 64  | 3   |
| 2022-07-14 20:19:00 | 63  | 2   |
| 2022-07-14 20:20:00 | 59  | 1   |
| 2022-07-14 20:21:00 | 59  | 1   |
| 2022-07-14 20:22:00 | 60  | 1   |
+-----+-----+-----+
7 rows in set (0.05 sec)
```

# 使用 DLC (Hive) 分析 CLS 日志

最近更新时间：2024-01-20 17:28:40

## 概述

当您需要将日志服务 CLS 中的日志投递到 Hive 进行 OLAP 计算时，可以参考本文进行实践。您可以通过腾讯云数据湖计算 DLC (Data Lake Compute, DLC) 提供的数据分析与计算服务，完成对日志的离线计算和分析。

## 操作步骤

### CLS 日志投递至 COS

#### 创建投递任务

1. 登录日志服务控制台，选择左侧导航栏中的 [投递任务管理](#) > [投递至COS](#)。
2. 在“投递至COS”页面中，单击 [添加投递配置](#)，在弹出的“投递至COS”窗口中，配置并创建投递任务。

如下配置项需要注意：

配置项	注意事项
目录前缀	日志文件会投递到对象存储桶的该目录下。在数据仓库模型中，一般对应为 table location 的地址。
分区格式	投递任务可按照创建时间进行自动分区，分区格式建议按照 hive 分区表格式指定。例如，按天分区可以设置为 /dt=%Y%m%d/test，其中 dt= 代表分区字段，%Y%m%d 代表年月日，test 代表日志文件前缀，因投递文件默认是以下划线(_)开头，大数据计算引擎会将这类文件忽略，导致查询不到数据，故需增加一个前缀，实际分区目录名称为 dt=20220424。
投递间隔时间	可在5 - 15分钟范围内选择，建议选择15分钟，250MB，这样文件数量会比较少，查询性能更佳。
投递格式	推荐 JSON 格式。

#### 查看投递任务结果

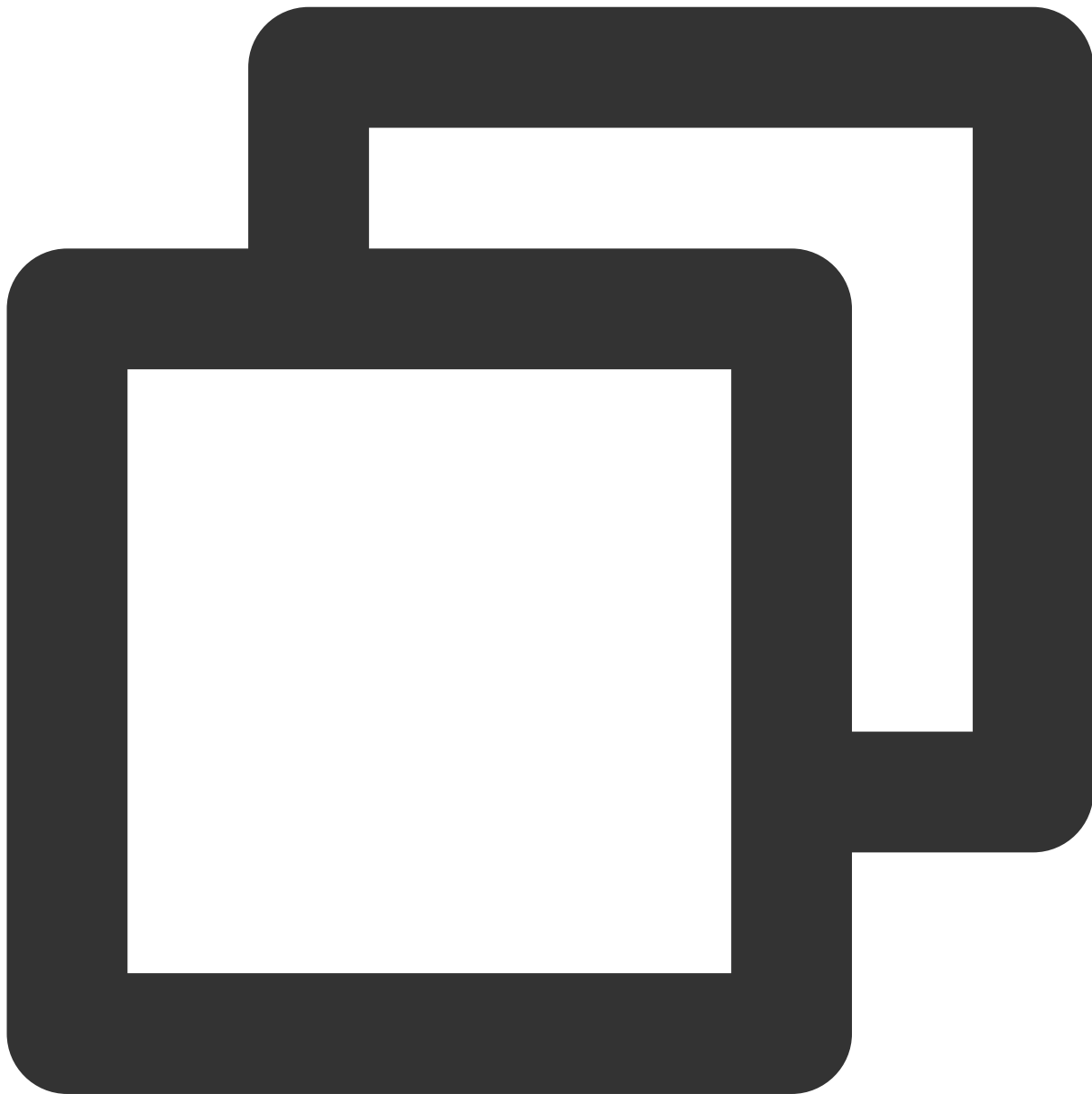
通常在启动投递任务15分钟后，可以在对象存储控制台查看到日志数据，如果在 log\_data 日志集设置了按天分区，则目录结构类似下图，分区目录下包含具体的日志文件。

## DLC (Hive) 分析

### DLC 创建外部表并映射到对象存储日志目录

日志数据投递至对象存储后，即可通过 DLC 控制台 → 数据探索功能创建外部表，建表语句可参考如下 SQL 示例，**需特别注意分区字段以及 location 字段要与目录结构保持一致。**

DLC 创建外表向导提供高级选项，可以帮助您推断数据文件表结构自动快捷生成 SQL，因为是采样推断所以需要您进一步根据 SQL 判断表字段是否合理，例如以下案例，**TIMESTAMP** 字段推断出为 int，但可能 bigint 才够用。



```
CREATE EXTERNAL TABLE IF NOT EXISTS `DataLakeCatalog`.`test`.`log_data` (  
  `__FILENAME__` string,  
  `__SOURCE__` string,  
  `__TIMESTAMP__` bigint,  
  `appId` string,  
  `caller` string,
```

```
`consumeTime` string,  
`data` string,  
`datacontentType` string,  
`deliveryStatus` string,  
`errorResponse` string,  
`eventRuleId` string,  
`eventbusId` string,  
`eventbusType` string,  
`id` string,  
`logTime` string,  
`region` string,  
`requestId` string,  
`retryNum` string,  
`source` string,  
`sourceType` string,  
`specversion` string,  
`status` string,  
`subject` string,  
`tags` string,  
`targetId` string,  
`targetSource` string,  
`time` string,  
`type` string,  
`uin` string  
) PARTITIONED BY (`dt` string) ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.Json
```

如果是按分区投递，**location** 需要指向 `cosn://coreywei-1253240642/log_data/` 目录，而不是 `cosn://coreywei-1253240642/log_data/20220423/` 目录。

使用推断功能，需要将目录指向数据文件所在的子目录即：`cosn://coreywei-1253240642/log_data/20220423/` 目录，推断完成后在 SQL 中 **location** 修改回 `cosn://coreywei-1253240642/log_data/` 目录即可。

适当分区会提升性能，但分区总数建议不超过1万。

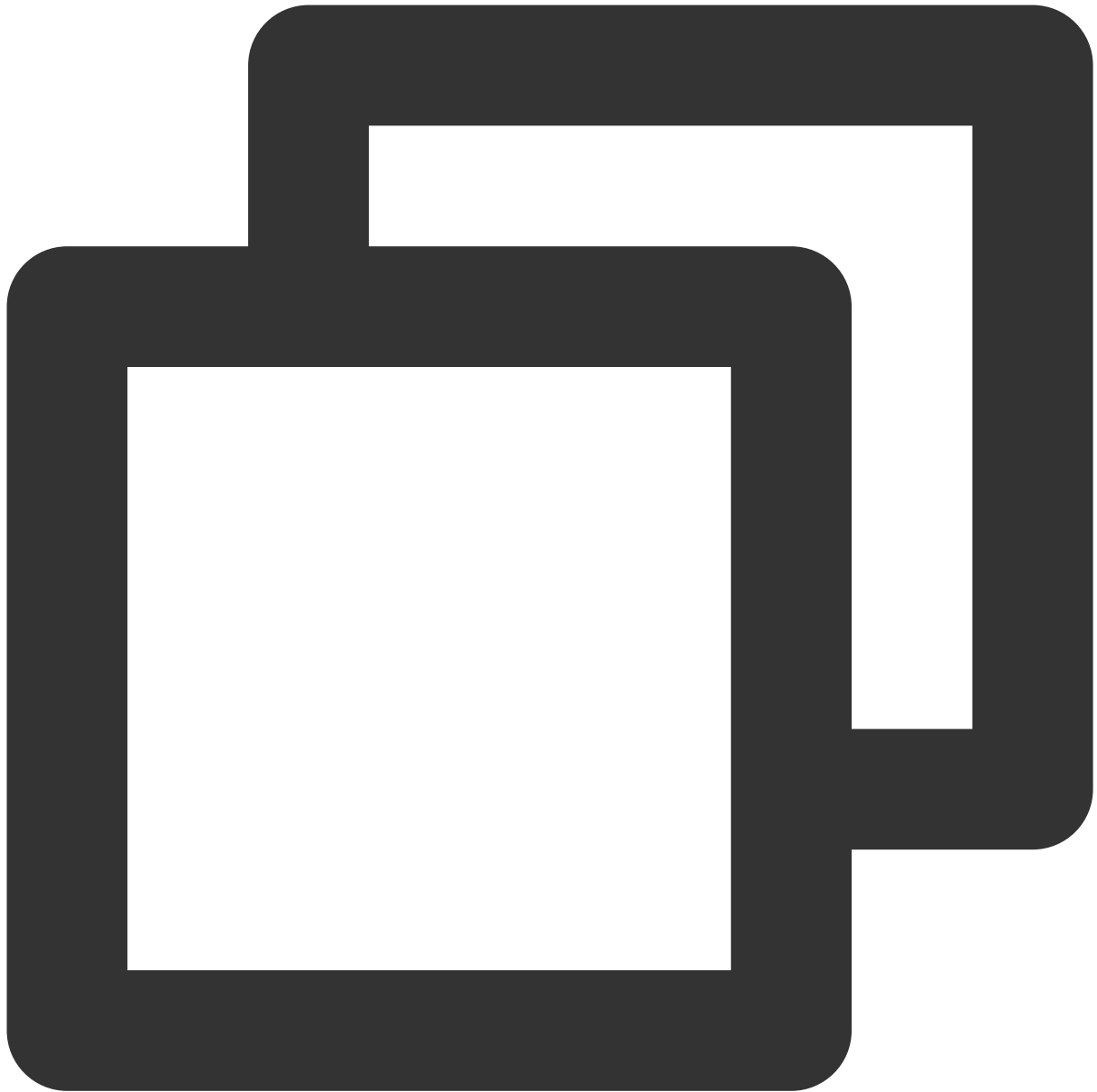
## 添加分区

分区表需要在添加分区数据后，才能通过 **select** 语句获取数据。您可以通过如下两种方式添加分区：

历史分区添加

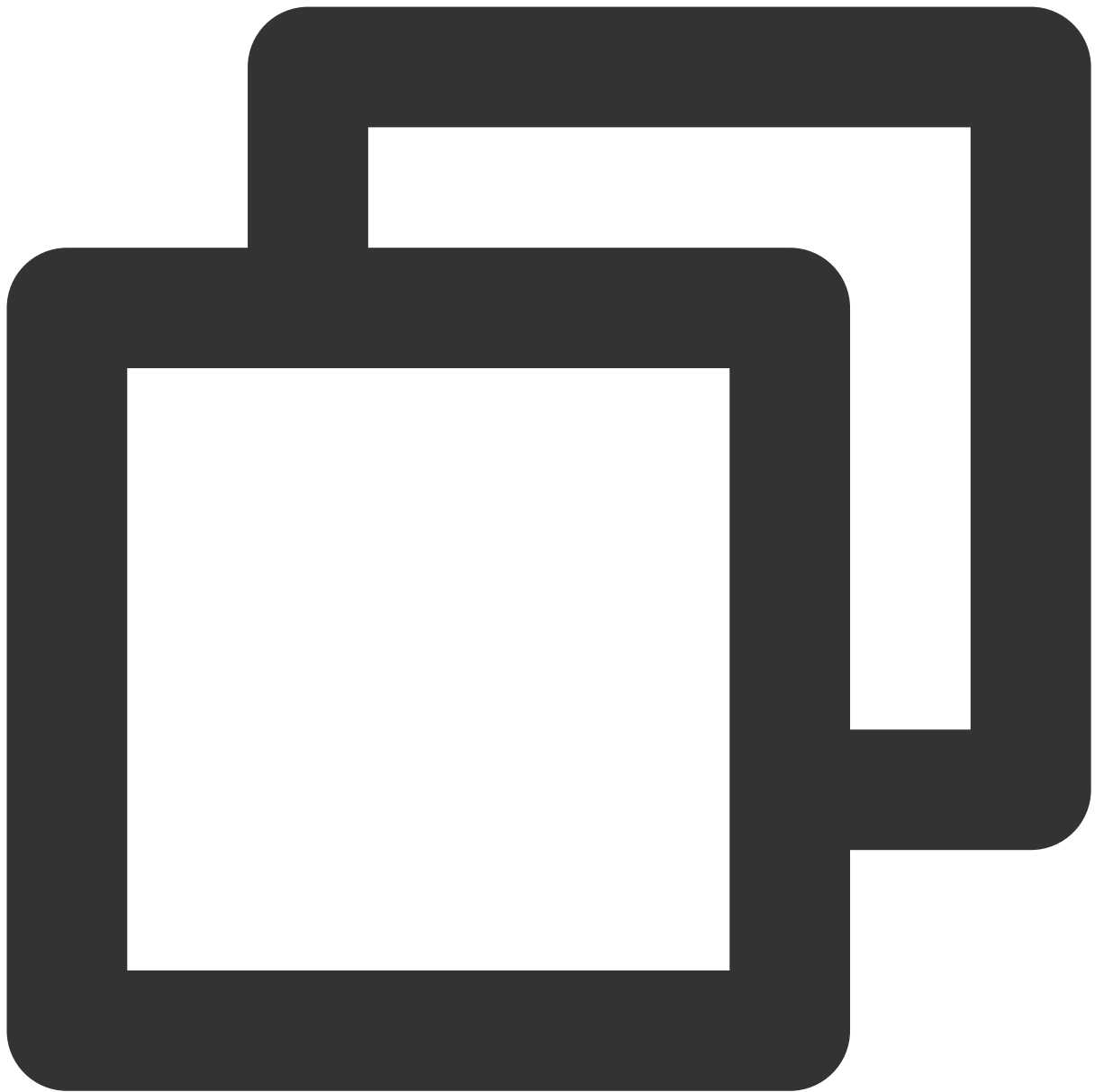
增量分区添加

该方案可一次性加载所有分区数据，运行较慢，适用首次加载较多分区场景。



```
msck repair table DataLakeCatalog.test.log_data;
```

在加载完历史分区之后，增量分区还会定期增加。例如，每天新增一个分区，则可以通过该方案进行增量添加。



```
alter table DataLakeCatalog.test.log_data add partition(dt='20220424')
```

### 分析数据

添加完分区后，即可通过 DLC 进行数据开发或分析。





```
select dt,count(1) from `DataLakeCatalog`.`test`.`log_data` group by dt;
```

# 成本优化

## 节省产品使用成本

最近更新时间：2024-01-20 17:28:40

### 成本节省建议

节省日志服务成本时，优先考虑费用较高的 [计费项](#)。通常为索引流量及索引存储这两项，费用一般是写流量和日志存储的数倍，其主要原因是因为它们按照未压缩的日志数据量进行计量计费。而使用 [LogListener](#) 上传日志时会进行数据压缩，写流量和日志存储是按照压缩后的日志数据量计量计费的，日志压缩率一般为1:4 - 1:10。

您可以结合自身需求按照如下方式调整日志主题配置，以达到节省成本的目的。

- 1. 降低日志上传数据量**：包括索引流量及索引存储在内的大部分计费项都与日志上传数据量相关，降低日志上传数据量可从源头实现成本节省。您可以适当的减少部分不必要的日志采集，或者使用 [LogListener](#) 采集配置 中的过滤器仅采集符合过滤器规则的日志，例如仅采集包含 **Error**、**Warning** 级别的日志。
- 2. 缩短日志存储周期**：日志存储周期越长，日志存储费用及索引存储费用越高。对于较老的日志，如果已经不再需要日常检索分析，只是需要作为一个历史数据进行备份，可 [投递至 COS](#) 中。
- 3. 精简索引配置**：简化索引配置可以降低索引流量及索引存储费用。需要注意的是当您同时开启全文索引和键值索引时，索引流量并不会重复计算，而是按类似并集的方式计算。因此当全文索引处于开启状态时，减少键值索引字段数量并不能降低索引流量和索引存储。您如果不需要进行全文检索，可以关闭全文索引，再减少一些不需要的键值索引字段。
- 4. 使用低频存储**：低频存储定位是以低成本方案解决海量低频日志的检索和存储问题，适用于用户对日志无统计分析要求，且日志保存时间较长的场景，能够节省80%左右的成本。详细说明参见 [低频存储简介](#)。
- 5. 使用数据加工分发日志主题**：数据加工可将原始日志投递至不同的日志主题中，例如按照日志级别：**ERROR**、**WARNING**、**INFO** 将日志分类，然后分发到不同的日志主题。您可以创建一个源日志主题用来接收所有的日志数据（这个日志主题可以不开启索引，不产生索引流量及索引存储费用，成本较低），然后使用数据加工按后续的日志检索分析需求将原始日志分别投递至不同存储周期、不同索引配置或不同存储类型的日志主题中，以更好的平衡使用需求及成本。详细说明参见 [创建加工任务](#)。