日志服务

常见问题

常见问题

基本问题

日志服务是什么？

日志服务（Log Service，简称LOG）是对日志收集、存储、订阅平台化服务。服务提供各种类型日志的实时收集，中心化管理、消费功能。

日志服务可以用来做什么事情？

1. 多种方式（API、SDK及Logtail接入服务）的日志写入途径
2. 通过Logtail自由定义日志的收集以及解析方式
3. 利用机器组管理数以千计机器上的日志收集
4. 提供实时日志消费与订阅功能
5. 简单易用的控制台配置方式,所有操作都可以在Web端完成
6. 后台与阿里云多个云产品无缝对接

日志服务的基本概念有哪些？

1. 核心概念为：Project（项目、管理日志基础单元）、LogStore（日志库）、Shard（分区）、Topic（主题、对于LogStore二级分类）、Log（日志条数）、LogGroup（日志组）
2. 日志收集概念：Logtail配置（定义如何收集日志配置）、机器分组（分组）

日志服务有哪几部分组成？

主要有日志收集客户端、服务端以及其他系统。客户端目前有Windows、Linux版本日志收集Agent（Logtail），服务端处理日志服务API读写、以及配置操作，其它系统包括ODPS，即能向ODPS同步日志数据，后续会支持向OSS和AMR同步日志数据进行消费。

日志服务如何定义一条日志？

日志包含三部分：时间（必填），日志内容（Key：Value对组成），以及元数据（Source，日志来源IP）

日志服务如何和其他阿里云服务协作？

1. ECS：用户可以安装Logtail（Agent）收集用户云主机上的各种日志数据。

2. ODPS：日志服务中收集的日志，可以自动投递到用户的ODPS表中，满足在线查询与离线分析的双重需求。

日志管理

日志服务如何存储、管理用户的日志？

日志库（Logstore）是日志服务中的日志存储和查询的基本单元，通常用于存储一类日志数据。目前，支持在控制台或者通过API完成对日志库的增删改查操作。日志库创建完成后，用户通过API或SDK向指定日志库写入日志数据。如果用户希望收集阿里云ECS服务器的数据，日志服务提供的Logtail日志收集服务同样非常方便地收集到日志数据。

删除日志库，日志数据是否丢失？

删除日志库会导致日志数据丢失，请谨慎操作。

日志服务日志保存多长时间？可否修改这个保存时限？

日志服务有三项功能都与日志保存时间有关，分别如下：

- LogHub（日志中枢）/LogSearch（日志索引与查询）：根据需求自行设置
- LogShipper（日志投递）：日志投递至OSS、ODPS后，生命周期在以上产品中设置

能否把日志分享给别人看？

目前日志服务已经支持RAM服务，可以通过向子帐号授权进行数据分享。

日志收集

日志服务可以收集哪些日志？

日志服务支持带有时间戳的文本日志，日志的时间必须是最近7以内的，并且不能超过当前时间15分钟。

日志服务有哪些渠道收集日志？应该如何选择这些渠道？

日志服务支持用户直接使用API写入（现有java,python,php,c#4个语言的sdk）；日志服务提供linux和windows版本的Logtail，用于收集磁盘文件上实时更新的日志。

1. 如果应用程序生成的日志不落磁盘，则可直接使用API写入到日志服务。

2. 实时写入磁盘的日志，可以通过Logtail来收集。

日志服务如何收集ECS上的日志？

可以使用Logtail来收集ECS上落在磁盘上的日志，过程如下：

1. 在日志服务控制台上，首先创建一个LogStore；
2. 配置logtail收集的配置
3. 创建机器分组
4. 通过安装脚本自助安装Logtail客户端
5. 将logtail的配置应用到需要的机器分组即可

日志服务可以收集历史日志吗？

用户可以通过API写入7天以内的数据，7天之前的数据写入会失败。但是，Logtail暂不支持收集历史数据。

日志服务收集数据的能力如何？有何限制？

用户可根据需求调整日志库（Logstore）的分区（Shard）数量。在ECS环境，logtail收集的速率被限制在1MB/秒

Logtail收集NAS上的日志需要注意什么？

例如Nginx accesslog收集场景，web server的nginx配置一般来说都是相同的，传统的方式会写在不同机器上相同名称的文件（Logtail可以正常采集）。使用NAS后，如果多台机器的nginx日志写入了NAS的相同文件（并发写相同文件场景），Logtail采集日志会缺失或出错。因此，请注意在使用NAS时，保证不同web server的日志写入NAS中的不同文件。

Logtail

Logtail是什么？

Logtail是日志服务提供的一种便于日志接入的日志收集客户端。通过在你的机器上安装Logtail来监听指定的日志文件并自动把新写入到文件的日志上传到你所指定的日志库。

Logtail是否可以收集静态不变的日志文件？

Logtail基于文件系统的修改事件来监听文件的变化，并将实时产生的日志发送到日志服务。如果日志文件没有发生任何修改行为，日志文件内容将不会被Logtail采集。

Logtail支持哪些平台？

目前支持Linux 64位系统。

如何安装、升级Logtail客户端？

安装：目前需要用户通过安装脚本自助安装Logtail客户端。升级：Logtail客户端的升级由日志服务定期完成，升级过程数据收集不中断。

如何配置使用Logtail客户端？

请参考：控制台配置Logtail收集日志说明。

Logtail如何工作？

1. 用户在控制台配置需要监控的目录、日志文件名以及相应的解析规则（正则表达式）等。
2. 用户机器上，日志文件发生修改，Logtail收到来自文件系统的事件并读取新产生的日志。
3. Logtail根据正则表达式解析日志格式并发往日志服务。

Logtail是否支持日志轮转？

对于日志文件a.LOG，当文件达到一定大小或创建超过一定时间后，a.LOG被mv为a.LOG.1（或其它名称），然后新建一个a.LOG继续写入日志。这个过程称为轮转。Logtail基于文件系统的事件通知，可以自动处理日志轮转的场景。

Logtail如何处理网络异常？

网络异常、写入Quota满时，Logtail会将采集到的日志内容写入本地磁盘缓存，并在稍后进行重试。磁盘缓存最大支持500MB，新缓存会覆盖旧缓存；超过24小时未发送成功的缓存文件将被自动删除。

Logtail日志收集延时如何？

Logtail基于事件进行日志收集，一般会在3秒内将日志发往日志服务。

Logtail如何处理历史日志？

Logtail只用于收集实时日志，如果日志内容的时间与Logtail处理该日志的系统时间相差5分钟以上，即被认为是历史日志。

日志服务修改日志收集配置后多久可以生效？

用户在控制台应用配置到机器组后，Logtail最迟会在3分钟之内加载新配置并生效。

如何调查Logtail收集日志问题？

常见问题如下：

1. 查看Logtail心跳是否正常，如不正常，请尝试重新安装Logtail。
2. 确认日志收集配置中的日志文件是否在实时产生。
3. 查看日志收集配置的正则表达式是否与日志内容相匹配。如正则匹配错误，可以在Logtail运行日志查看到相关错误（Linux:/usr/local/ilogtail/ilogtail.LOG）。

为什么我的Logtail心跳状态不正常？

目前，Logtail客户端目前只支持64位Linux操作系统。

如果发现心跳状态不正常，可参考如下步骤进行诊断：

- 检查Logtail进程是否存在。请执行如下命令确认进程是否存在。如果不存在请重新安装Logtail，否则进行下一步诊断操作。

```
sudo /etc/init.d/ilogtaild status
```

请执行如下这些命令检查网络联通性

经典网络

```
telnet logtail.cn-<region>-intranet.log.aliyuncs.com 80
```

VPC网络

```
telnet logtail.cn-<region>-vpc.log.aliyuncs.com 80
```

如果无法联通，请确认如下两点：

1. 如果本机设置了主机名（执行hostname命令查看）绑定(在文件/etc/hosts中)，需要确认绑定的IP是否与日志服务机器组中维护的IP一致；
2. 如果没有设置主机名绑定，请查看本机的第一块网卡IP，确认该IP与日志服务机器组维护的IP一致。

如果无法联通，服务端将无法接收这台机器的心跳数据包。请联系日志服务产品技术支持团队进行问题排查。

如根据上述原因排查后，仍然不能解决您的问题，请通过工单系统提交工单，将由日志服务产品技术支持为您进行问题排查。

Logtail 机器无心跳

目前，Logtail客户端目前只支持64位Linux操作系统和windows如果发现心跳状态不正常，可参考如下步骤进

行诊断：

Linux

检查Logtail进程是否存在。请执行如下命令确认进程是否存在。如果不存在请重新安装Logtail，否则进行下一步诊断操作。安装logtail文档。

sudo /etc/init.d/ilogtaild status

如果正常的话，打开文件 /usr/local/ilogtail/ilogtail.LOG。

找到config

[2016-12-12 12:07:10.968201] [INFO] [3726] [build/release64/sls/ilogtail/AppConfig.cpp:212] config server:http://logtail.cn-hangzhou-intranet.log.aliyuncs.com https config server:https://logtail.cn-hangzhou-intranet.log.aliyuncs.com

看看您的project是那个域的，是不是能跟这个对上，示例是杭州的project，走内网。服务入口。

地域能对上的话，检查一下网络连通性

经典网络

telnet logtail.cn-<region>-intranet.log.aliyuncs.com 80

VPC网络

telnet logtail.cn-<region>-vpc.log.aliyuncs.com 80

公网

telnet logtail.cn-<region>.log.aliyuncs.com 80

如果域没有问题，网络也是通的，找到UUID开头的这一段

[2016-12-12 12:07:10.970278] [INFO] [3726] [build/release64/sls/ilogtail/elogtail.cpp:113] Logtail started, appInfo:{“UUID” : “92B9E907-6D2F-4F1E-B856-32A8C06F2BF6”，“hostname” : “machinename”，“instance_id” : “A1841794-B793-11E6-B8FF-00163E2EBA3C”，“ip” : “XXX.XXX.XXX.XXX”，“logtail_version” : “0.11.6”，“os” : “Linux; 2.6.32-573.22.1.el6.x86_64; #1 SMP Wed Mar 23 03:35:39 UTC 2016; x86_64”，“update_time” : “2016-12-12 12:07:10”}

找到下面的这个”ip” : “XXX.XXX.XXX.XXX” ，控制台必须配置这个ip，否则无心跳。

如果发现文件中这个ip为空，说明您的机器没有第一块网卡（ifconfig eth0），遇到这种情况，请手工绑定一下hosts

vi /etc/hosts

第一行添加，保存

XXX.XXX.XXX.XXX machinename

XXX.XXX.XXX.XXX可以填其他网卡ip，就是拿来做个心跳标签，跟控制台配置的ip对上就行。machinename用hostname命令取一下。

控制台配置的ip必须跟这个文件里面的对上，这个ip是个标签，不影响走哪个网络，取这个ip的方法：先取/etc/hosts 里面绑定的ip，如果没有绑定，取第一块网卡的ip（ifconfig eth0），最后会生成在/usr/local/ilogtail/ilogtail.LOG 这个文件里面，所以直接看这个文件最简单。

以上都没有问题，查下/usr/local/ilogtail/ilogtail.LOG这个文件，是否有这个错误 Unauthorized ErrorMessage:no authority, denied by ACL。

如果有的话，是您的主账号没有access key造成的，现在主账号没有access key，logtail不能正常运行。去ak-console.aliyun.com建个access key，就会有心跳了。

上述都检查过，还没有心跳，再开工单解决。

windows

排查方法跟linux类似，logtail的日志在这个文件里：C:\Program Files (x86)\Alibaba\Logtail\logtail_0.log。

日志查询

日志服务提供哪些渠道查询收集的日志？

日志服务提供了三种方式查询日志

1. 通过 日志服务控制台查询。
2. 通过sdk(c++,java,php,dotnet,python）查询。详见SDK。
3. 通过restful API查询，详见API。

日志服务提供什么样的查询能力？

1. 提供组合条件过滤查询，查询语法参见日志查询语法。

2. 能够提供单次查询1000万日志的能力。用户可以根据一定的条件筛选出需要的日志，读取命中日志在时间维度上的分布，或者拿到原始日志。
3. 查询提供了cache的功能，第二次查询相同的条件获得更加完整的查询结果。

日志查询有什么限制？

1. 最多能够查询10个词组成的组合条件。
2. 单次查询结果最多获取100行原始数据。
3. 单次查询最多处理1000万行数据。

日志查询无任何数据返回如何处理？

1. 回到“LogStore列表”页面，找到“日志消费模式”中的“日志消费”列，单击“预览”选择ShardID查看是否有数据。
    - 如果有数据，回到“日志索引”列“查询”功能页面，单击“关键词”搜索框边的“topic”输入框，确认是否有topic
        - 如果有topic，单击选择其中一个topic，点击“查询”按钮查看是否有数据
        - 如果没有topic，并且点击“查询”按钮查看仍然无数据，请提交工单
    - 如果没有数据，请确认数据是否收集全

日志服务监控指标

指标含义

监控数据入口请参考LogHub监控章节。

1. 写入/读取流量
    - 含义：每个日志库（logstore）写入、以及读取实时情况
    - 单位：Bytes/min
    - 含义：统计该logStore通过ilogtail和SDK、API等读写实时流量，大小为传输大小（压缩情况下为压缩后），每分钟统计一个点，单位为字节/分钟。
2. 原始数据大小
    - 含义：每个Logstore写入数据原始大小（压缩前）
    - 单位：Byte/min
3. 总体QPS
    - 含义：所有操作QPS，每分钟统计一个点
    - 单位：Count/Min
4. 操作次数
    - 含义：统计用户的各种操作对应的QPS，每分钟统计一个点
    - 单位：次/分钟（Count/Min)
    - 所有的操作包括：

        - 写入操作：
            - PostLogStoreLogs ：0.5API以后版本接口。
            - PutData ： 0.4 API以前版本接口。
        - 根据关键字查询：
            - GetLogStoreHistogram: 查询关键字分布情况，0.5API以后版本接口。
            - GetLogStoreLogs: 查询关键字命中日志，0.5API以后版本接口。
            - GetDataMeta ： 同GetLogStoreHistogram，为0.4API以前版本接口。
            - GetData : 同GetLogStoreLogs，为0.4API以前版本接口。
        - 批量获取数据：
            - GetCursorOrData：该操作包含了获取Cursor和批量获取数据两种方法。
            - ListShards：获取一个LogStore下所有的Shard。
        - List操作：
            - ListLogStoreLogs:遍历一个project下所有的LogStore。
            - ListCategory：同ListLogStoreLogs，为0.4API以前版本接口。
            - ListLogStoreTopics：遍历一个Logstore下所有的Topic。

5. 服务状态
    - 含义：该视图统计用户的各种操作返回的HTTP 状态码对应的QPS，方便用户根据错误的返回码来判断操作异常，及时调整程序。
    - 各状态码:
        - 200：为正常的返回码，表示操作成功。
        - 400：错误的参数，包括Host,Content-length,APIVersion,RequestTimeExpired,查询时间范围，Reverse，AcceptEncoding,AcceptContentType,Shard,Cursor，PostBody,Paramter,ContentType等方面的错误。
        - 401：鉴权失败，包括AccessKeyId不存在、签名不匹配、或者签名账户没有操作权限，请到SLSweb上查看project权限列表，是否包含了该AK。
        - 403: 超过预定Quota，包括能够创建的LogStore个数、Shard总数、以及读写操作的每分钟限额，请根据返回的Message判断发生了哪种错误。
        - 404：请求的资源不存在，包括project、 LogStore、Topic 、User等资源。
        - 405：错误的操作方法，请检查请求的URL路径。
        - 500：服务端错误，请重试。
        - 502：服务端错误，请重试。
6. 客户端解析成功流量
    - 含义：Logtail收集成功的日志大小，为原始数据大小
    - 单位：字节
7. 客户端（Logtail）解析成功行数
    - 含义：Logtail收集成功的日志的行数
    - 单位: 行
8. 客户端解析失败行数
    - 含义：Logtail收集日志过程中，采集出错的行数大小，如果该视图有数据，表示有错误发

生
- 单位：行

9. 客户端错误次数
   - 含义：Logtail收集日志过程中，出现所有收集错误的IP总数
   - 单位：次
10. 发生客户端错误机器数
   - 含义：Logtail收集日志过程中，出现收集错误的告警次数
   - 单位：个
11. 错误IP统计（Count/5min)
   - 含义：分类别展示各种采集错误发生的IP数，各种错误包括：
     - LOGFILE_PERMINSSION_ALARM:没有权限打开日志文件。
     - SENDER_BUFFER_FULL_ALARM：数据采集速度超过了网络发送速度，数据被丢弃。
     - INOTIFY_DIR_NUM_LIMIT_ALARM（INOTIFY_DIR_QUOTA_ALARM）：监控的目录个数超过了3000个，请把监控的根目录设置成更低层目录。
     - DISCARD_DATA_ALARM：数据丢失，因为数据时间在系统时间之前15分钟，请保证新写入日志文件的数据是在15分钟之内的。
     - MULTI_CONFIG_MATCH_ALARM：有多个配置在收集同一个文件，logtail会随机选择一个配置进行收集，另一个配置则收集不到数据。
     - REGISTER_INOTIFY_FAIL_ALARM：注册inotify事件失败，具体原因请查看logtail日志。
     - LOGDIR_PERMINSSION_ALARM：没有权限打开监控目录。
     - REGEX_MATCH_ALARM：正则式匹配错误，请调整正则式。
     - ENCODING_CONVERT_ALARM：转换日志编码格式时出现错误，具体原因请查看logtail日志。
     - PARSE_LOG_FAIL_ALARM：解析日志错误，一般是行首正则表达式错误或单条日志超过512KB导致的日志分行错误，请查看Logtail日志确定原因，如行首正则表达式错误请调整配置。
     - DISCARD_DATA_ALARM：丢弃数据，Logtail发送数据到日志服务失败且写本地缓存文件失败导致，可能的原因是日志文件产生较快但写磁盘缓存文件较慢。
     - SEND_DATA_FAIL_ALARM：解析完成的日志数据发送日志服务失败，请查看Logtail日志发送数据失败相关ErrorCode和ErrorMessage，常见的错误有服务端Quota超限、客户端网络异常等。
     - PARSE_TIME_FAIL_ALARM：解析日志time字段出错，Logtail根据正则表达式解析出来的time字段按照时间格式配置无法解析成功，请修改配置。
     - OUTDATED_LOG_ALARM：Logtail丢弃历史数据，请保证当前写入日志数据的时间与系统时间相差在5分钟以内。
   - 请根据具体错误请找到出错IP，登录机器查看/usr/logtail/ilogtail.LOG查看错误原因

使用监控+报警

Logtail日志是否完整收集？

客户端（Logtail)在运行过程中，可能会因设置不正确产生错误：例如某些日志格式不匹配，一个日志文件被重复收集等（Logtail场景问题）。为了及时发现这种情况，我们可以对客户端解析失败行数、客户端错误次数等指标进行监控，以及时发现这类问题。步骤如下：

i. 打开LogHub云监控页面，参考LogHub监控章节。
ii. 选择关心的Logstore，新建报警规则：
    i. 选择”客户端解析错误”
    ii. 设置规则，当出错误后报警
    iii. 设置报警短信接收人进行报警

iii. 除此之外，还可以根据Logtail其他错误项进行报警，第一时间发现各类日志收集过程中发现的问题

Logstore下Shard资源是否足够

Logstore下每个Shard提供5MB/S (1000次/S) 写入能力，这个数值对于大部分用户而言都是足够的，在超过时日志服务会尽可能去服务（非拒绝）你的请求，但在高峰期间不保证超出部分可用性。如果你的日志量非常大，需要添加更多Shard，可以在控制台（日志消费/修改）中进行调整。在之前可以设置logstore出入流量报警以检测该情况。可以设置Logstore流量报警以检测该情况，步骤如下：

- 方案1：对流量预警
    i. 打开LogHub云监控页面，参考LogHub监控章节。
    ii. 选择关心的Logstore，新建报警规则：
        - 选择”原始数据大小”
        - 设置规则，超过25GB/Min后进行报警
        - 设置报警短信接收人进行报警
- 方案2：设置状态码报警
    i. 打开LogHub云监控页面，参考LogHub监控章节。
    ii. 选择关心的Logstore，新建报警规则：
        - 选择”服务状态”，status填写403（流量超过阈值）
        - 设置规则，超过25GB/Min后进行报警
        - 设置报警短信接收人进行报警

1. **Project Quota是否足够** 每个Project默认写入限制为30GB/分钟（原始数据大小），这个数值主要目的是为了保护用户因程序错误产生大量日志设计，在一般场景中对于大部分用户都是足够的。如果你的日志量非常大，可能会超过限制，可以通过工单联系我们调整大这个数值。Logstore流量报警与预警步骤如上。

日志服务Loghub功能 和 Kafka简介

Kafka是分布式消息系统，由于其高吞吐和水平扩展，被广泛使用于消息的发布和订阅。以开源软件的方式提供，各用户可以根据需要搭建Kafka集群。

日志服务(Log Service):是基于飞天Pangu构建的针对日志平台化服务。服务提供各种类型日志的实时收集，存储，分发及实时查询能力。通过标准话的Restful API对外提供服务。

Log Service Loghub提供公共的日志收集、分发通道，用户如果不想自己搭建、运维kafka集群，可以使用Log Service LogHub功能。

Log Service Loghub & Kafka 概念映射

概念	Kafka	Loghub
存储对象	topic	logstore
水平分区	partition	shard
数据消费位置	offset	cursor

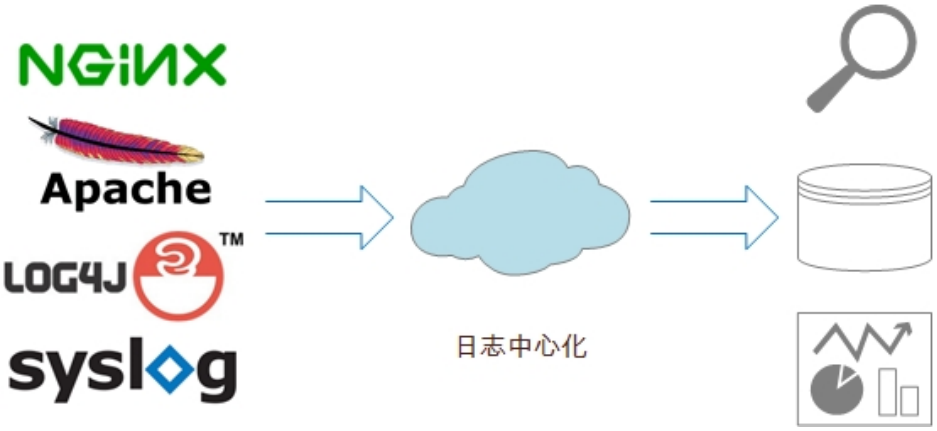
Loghub & Kafka 功能比较

功能	Kafka	LogHub
使用依赖	自建或共享Kafka集群	Log Service服务
通信协议	TCP 打通网络	Http (restful API)，80端口
访问控制	无	基于云账号的签名认证+访问控

		制
动态扩容	暂无	支持动态shard个数弹性伸缩(Merge/Split)，对用户无影响
多租户Qos	暂无	基于shard的标准化流控
数据拷贝数	用户自定义	暂不开放，默认3份拷贝
failover/replication	调用工具完成	自动，用户无感知
扩容/升级	调用工具完成，影响服务	用户无感知
写入模式	round robin/key hash	暂只支持round robin/key hash
当前消费位置	保存在kafka集群的zookeeper	服务端维护、无需关心
保存时间	配置确定	根据需求动态调整

日志收集场景下客户端测评

DT时代，数以亿万计的服务器、移动终端、网络设备每天产生海量的日志。中心化的日志处理方案有效地解决了在完整生命周期内对日志的消费需求，而日志从设备采集上云是始于足下的第一步。



三款日志收集工具

logstash

- 开源界鼎鼎大名ELK stack中的”L”，社区活跃，生态圈提供大量插件支持
- logstash基于JRuby实现，可以跨平台运行在JVM上
- 模块化设计，有很强的扩展性和互操作性。

fluentd

- 开源社区中流行的日志收集工具，td-agent是其商业化版本，由Treasure Data公司维护，是本文选用的评测版本。
- fluentd基于CRuby实现，并对性能表现关键的一些组件用C语言重新实现，整体性能不错。
- fluentd设计简洁，pipeline内数据传递可靠性高
- 相较于logstash，其插件支持相对少一些。

- logtail
  - 阿里云日志服务的生产者，经过3年多阿里集团大数据场景考验
  - 采用C++语言实现，对稳定性、资源控制、管理等下过很大的功夫，性能良好
  - 相比于logstash、fluentd的社区支持，logtail功能较为单一，专注日志收集功能。

功能对比

功能项	logstash	fluentd	logtail
日志读取	轮询	轮询	事件触发
文件轮转	支持	支持	支持
Failover处理 (本地checkpoint)	支持	支持	支持
通用日志解析	支持grok（基于正则表达式）解析	支持正则表达式解析	支持正则表达式解析
特定日志类型	支持delimiter、key-value、json等主流格式	支持delimiter、key-value、json等主流格式	支持delimiter、key-value、json等主流格式
数据发送压缩	插件支持	插件支持	LZ4
数据过滤	支持	支持	支持
数据buffer发送	插件支持	插件支持	支持
发送异常处理	插件支持	插件支持	支持
运行环境	JRuby实现，依赖JVM环境	CRuby、C实现，依赖Ruby环境	C++实现，无特殊要求
线程支持	支持多线程	多线程受GIL限制	支持多线程
热升级	不支持	不支持	支持
中心化配置管理	不支持	不支持	支持
运行状态自检	不支持	不支持	支持cpu/内存阈值保护

日志文件收集场景 - 性能对比

日志样例:以Nginx的access log为样例，如下一条日志365字节，结构化成14个字段：

在接下来的测试中，将模拟不同的压力将该日志重复写入文件，每条日志的time字段取当前系统时间，其它13个字段相同。

相比于实际场景，模拟场景在日志解析上并无差异，有一点区别是：较高的数据压缩率会减少网络写出流量。

logstash

logstash-2.0.0版本，通过grok解析日志并写出到kafka（内置插件，开启gzip压缩）。

日志解析配置：

```
grok {
patterns_dir=>"/home/admin/workspace/survey/logstash/patterns"
match=>{ "message"=>"%{IPORHOST:ip} %{USERNAME:rt} - \[%{HTTPDATE:time}\] \"%{WORD:method}
%{DATA:url}\" %{NUMBER:status} %{NUMBER:size} \"%{DATA:ref}\" \"%{DATA:agent}\" \"%{DATA:cookie_unb}\"
\"%{DATA:cookie_cookie2}\" \"%{DATA:monitor_traceid}\" %{WORD:cell} %{WORD:ups}
%{BASE10NUM:remote_port}" }
remove_field=>["message"]
}
```

测试结果：

写入TPS	写入流量 (KB/s)	CPU使用率 (%)	内存使用 (MB)
500	178.22	22.4	427
1000	356.45	46.6	431
5000	1782.23	221.1	440
10000	3564.45	483.7	450

fluentd

td-agent-2.2.1版本，通过正则表达式解析日志并写入kafka（第三方插件fluent-plugin-kafka，开启gzip压缩）。

日志解析配置：

```
<source>
type tail
format /^(?<ip>\S+)\s(?<rt>\d+)\s-
\s\[(?<time>[^\]]*)\]\s"(?<url>[^\"]+)"\s(?<status>\d+)\s(?<size>\d+)\s"(?<ref>[^\"]+)"\s"(?<agent>[^\"]+)"\s"(?<
cookie_unb>\d+)"\s"(?<cookie_cookie2>\w+)"\s"(?
<monitor_traceid>\w+)"\s(?<cell>\w+)\s(?<ups>\w+)\s(?<remote_port>\d+).*$/
```

```
time_format %d/%b/%Y:%H:%M:%S %z
path /home/admin/workspace/temp/mock_log/access.log
pos_file /home/admin/workspace/temp/mock_log/nginx_access.pos
tag nginx.access
</source>
```

测试结果：

写入TPS	写入流量 (KB/s)	CPU使用率 (%)	内存使用 (MB)
500	178.22	13.5	61
1000	356.45	23.4	61
5000	1782.23	94.3	103

注：受GIL限制，fluentd单进程最多使用1个cpu核心，可以使用插件multiprocess以多进程的形式支持更大的日志吞吐。

logtail

logtail 0.9.4版本，设置正则表达式进行日志结构化，数据LZ4压缩后以HTTP协议写到阿里云日志服务，设置batch_size为4000条。

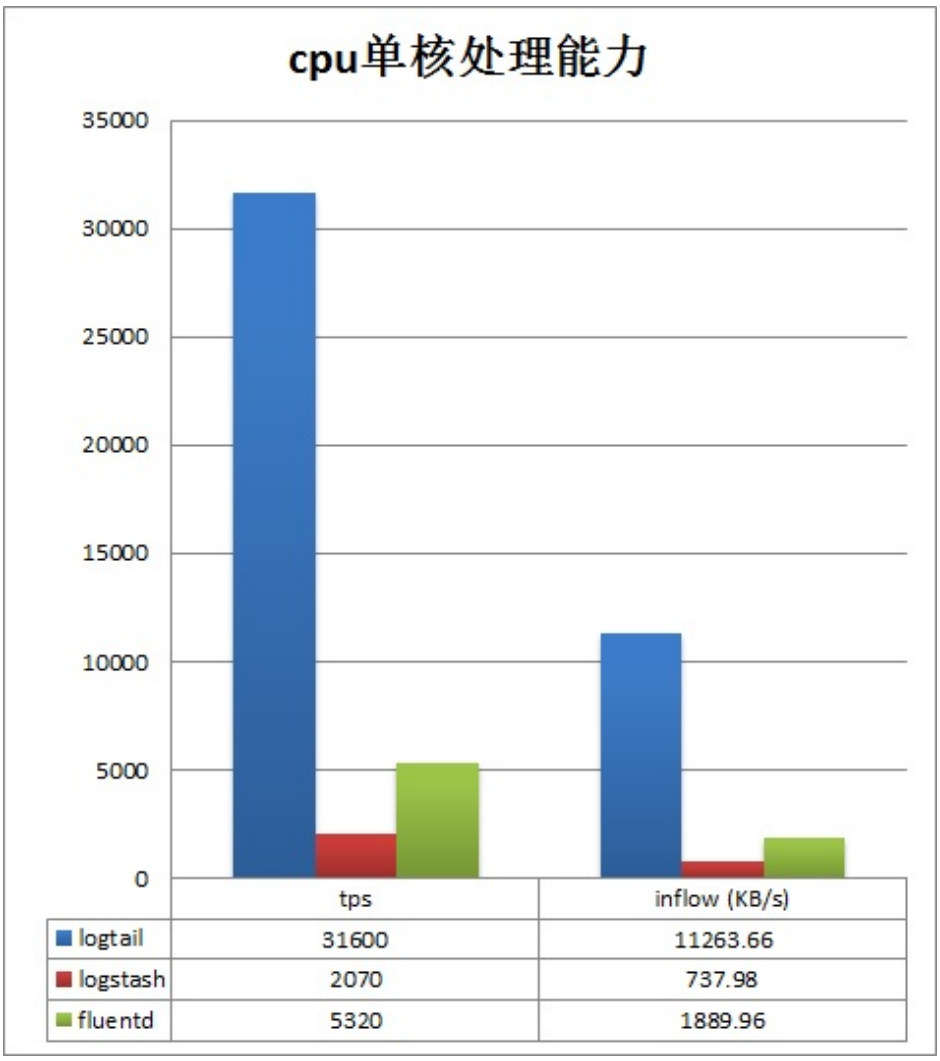
日志解析配置：

```
logRegex : (\S+)\s(\d+)\s-
\s\[([^]]+)]\s"([^"]+)"\s(\d+)\s(\d+)\s"([^"]+)"\s"([^"]+)"\s"(\d+)"\s"(\w+)"\s"(\w+)"\s(\w+)\s(\w+)\s(\d+).*
keys : ip,rt,time,url,status,size,ref,agent,cookie_unb,cookie_cookie2,monitor_traceid,cell,ups,remote_port
timeformat : %d/%b/%Y:%H:%M:%S
```

测试结果：

写入TPS	写入流量 (KB/s)	CPU使用率 (%)	内存使用 (MB)
500	178.22	1.7	13
1000	356.45	3	15
5000	1782.23	15.3	23
10000	3564.45	31.6	25

单核处理能力对比



总结

可以看到三款日志工具各有特点：

- logstash支持所有主流日志类型，插件支持最丰富，可以灵活DIY，但性能较差，JVM容易导致内存使用量高。
- fluentd支持所有主流日志类型，插件支持较多，性能表现较好。
- logtail占用机器CPU/内存资源最少，性能吞吐量较好，针对常用日志场景支持全面，但缺少插件等机制，灵活性和可扩展性不如以上两个客户端。

RAM 子帐号访问控制台

前提条件

使用RAM子帐号访问日志服务控制台，子帐号本身需要满足以下两个条件：

- 需要为该子用户创建AccessKey。
- 必须启用该子帐号的控制台登录功能，为该子帐号设置登录用户名、密码。

设置方法请参考文档：《子帐号控制台登录》

授权子帐号访问日志服务控制台

满足登录前提条件的子帐号，要使用日志服务控制台还需要授权。控制台使用日志服务提供的OpenAPI访问后台的日志服务，因此这一步本质上就是授予子帐号调用日志服务某些OpenAPI的权限，下面针对一些典型的子用户访问场景，对授权方法进行说明。

授予子帐号对父帐号日志服务资源只读权限

1. 登录访问控制服务控制台，到子用户管理界面，选择要授权的子用户。操作方法参考访问控制文档《给用户授权》
2. 在授权策略页面选择AliyunLogReadOnlyAccess。

授予子帐号对父帐号日志服务资源完全访问权限

和场景《授予子帐号对父帐号日志服务资源只读权限》的唯一区别在于，授权策略选择AliyunLogFullAccess。

更细粒度的子用户权限控制

在访问控制服务控制台可以自定义授权策略，创建方法参考创建自定义授权策略，日志服务提供的权限策略参考日志服务RAM授权策略。

授权策略例子：

1. 子帐号可以看到父帐号有哪些日志服务Project。
2. 子帐号对父帐号的某个日志服务Project有只读的访问权限。

同时满足1、2的授权策略如下：

```
{
"Version": "1",
"Statement": [
{
"Action": ["log:ListProject"],
"Resource": ["acs:log:*:*:project/*"],
"Effect": "Allow"
} ,
{
"Action": [
"log:Get*",
"log:List*"
],
"Resource": "acs:log:*:*:project/<指定的Proejct名称>/*",
"Effect": "Allow"
}
```

```
]
}
```

日志服务目前提供基于用户日志内容报警功能，为了读取日志数据，需要用户显式授权日志服务服务账号访问用户数据。如果已经阅读过该文档完成授权，可以略过以下内容直接创建报警规则，具体授权步骤如下具体说明。

1. 创建访问控制（RAM）角色

用户登录访问控制控制台，打开“角色管理”功能，单击右上角“新建角色”按钮，第一步角色类型选择“服务角色”。

第二步类型信息选择“LOG日志服务”

第三步填写角色名称为“aliyunlogreadrole”（默认会扮演该角色访问数据，请不要修改名称），点击创建完成。

2. 授权角色访问日志数据权限

角色创建完成后，在“角色管理”列表中，找到“aliyunlogreadrole”角色名称，点击“授权”，搜索“AliyunLogReadOnlyAccess”权限，直接应用。

完成如上步骤后，日志服务即有权限定期读取指定日志库数据进行报警检查。