日志服务

产品简介

产品简介

什么是日志服务

日志服务（Log Service，简称 Log）是针对日志类数据的一站式服务，在阿里巴巴集团经历大量大数据场景锤炼而成。您无需开发就能快捷完成日志数据采集、消费、投递以及查询分析等功能，提升运维、运营效率，建立 DT 时代海量日志处理能力。

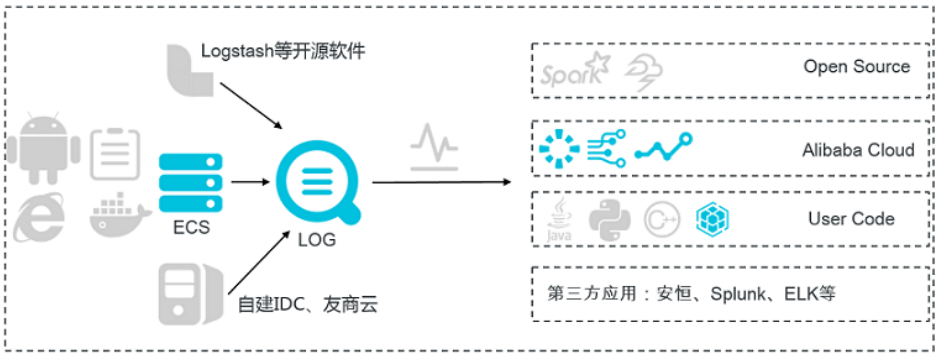
核心功能如下：

实时采集与消费（LogHub）

功能：

- 通过ECS、容器、移动端，开源软件，JS等接入实时日志数据（例如Metric、Event、BinLog、TextLog、Click等）
- 提供实时消费接口，与实时计算及服务对接

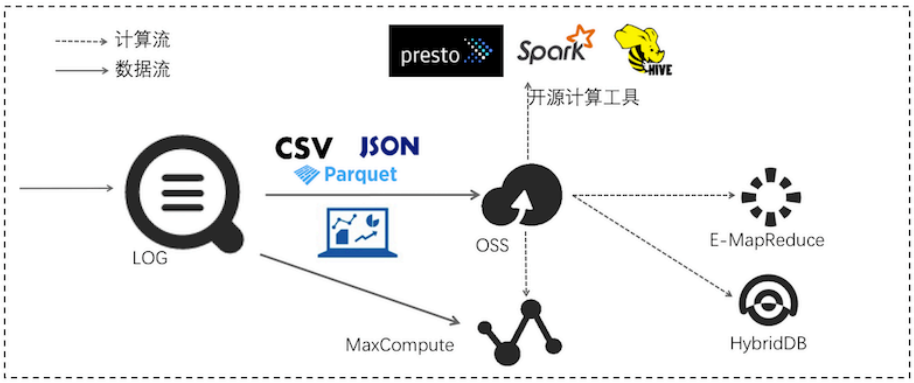
用途：数据清洗（ETL），流计算（Stream Compute），监控与报警， 机器学习与迭代计算



投递数仓（LogShipper）

稳定可靠的日志投递。将日志中枢数据投递至存储类服务进行存储与大数据分析。支持压缩、自定义Partition、以及行列等各种存储方式。

用途：数据仓库 + 数据分析、审计、推荐系统与用户画像

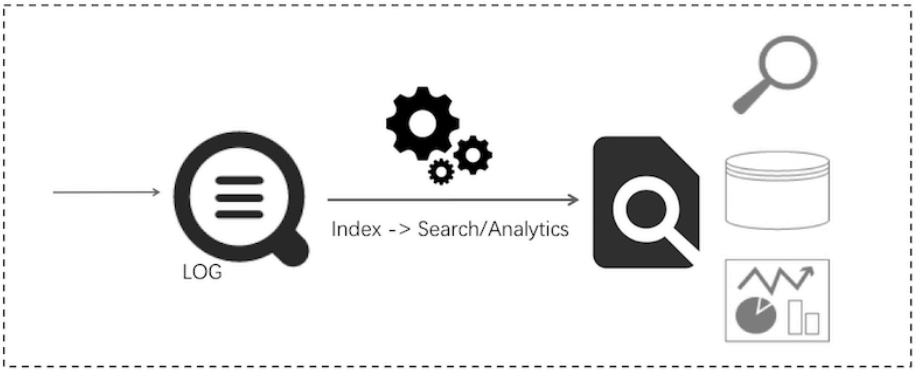


查询与实时分析（Search/Analytics）

实时索引、查询分析数据。

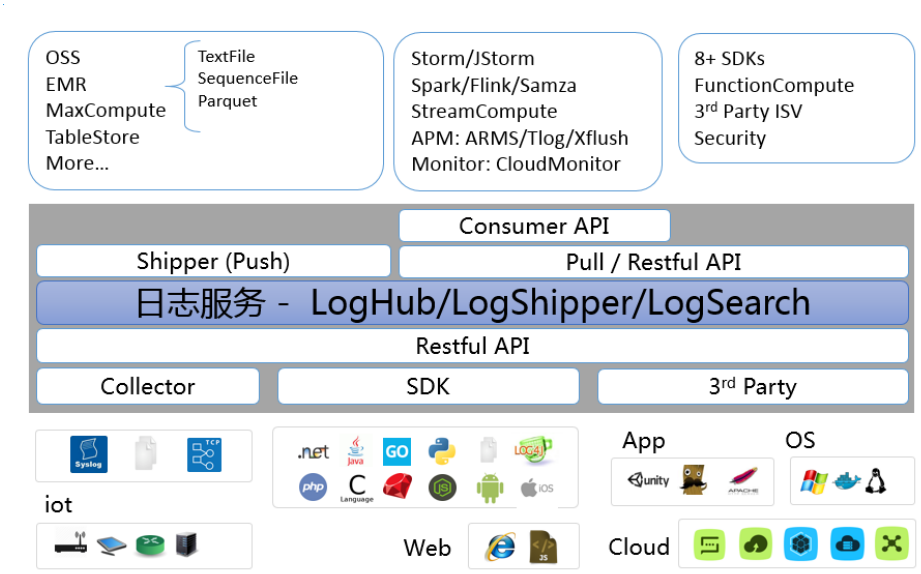
- 查询：关键词、模糊、上下文、范围
- 统计：SQL聚合等丰富查询手段
- 可视化：Dashboard + 报表功能
- 对接：Grafana，JDBC/SQL92

用途：DevOps/线上运维，日志实时数据分析，安全诊断与分析，运营与客服系统



产品架构

日志服务的系统架构如下图所示：



Logtail

帮助您快速收集日志的代理。其特点如下所示：

- 基于日志文件、无侵入式的收集日志
  - 仅读取文件
  - 读取过程无侵入
- 安全、可靠
  - 支持文件轮转不丢失数据
  - 支持本地缓存
  - 网络异常重试
- 方便管理
  - Web端管理
  - 支持可视化配置
- 完善的自我保护
  - 实时监控进程 CPU、内存消耗
  - 限制内存使用上限

前端服务器

采用LVS+Nginx构建的前端机器。其特点如下所示：

- HTTP、REST协议
- 水平扩展
  - 流量上涨时支持水平扩展。
  - 可快速通过增加前端机来提高处理能力。
- 高吞吐、低延时
  - 纯异步处理，单个请求异常不会影响其他请求。

- 内部采用专门针对日志的Lz4压缩，提高单机处理能力，降低网络带宽。

后端服务器

后端是分布式的进程，部署在多个机器上，完成实时对Logstore数据的持久化，索引，查询，以及投递至MaxCompute。整体后端服务的特点如下所示：

- 数据高安全性：
  - 您写入的每条日志，都会被保存3份。
  - 任意磁盘损坏、机器硬件或软件系统错误的情况下，数据自动复制修复。
- 稳定服务：
  - 进程崩溃和机器长时间无响应时，Logstore会自动迁移。
  - 自动负载均衡，确保无单机热点。
  - 严格的Quota限制，防止单个用户行为异常对其他用户产生影响。
- 水平扩展：
  - 以分区（Shard）为单位进行水平扩展。
  - 用户可以按需动态增加分区来增加吞吐量。

产品优势

全托管服务

- 易用性强，5分钟即可接入服务进行使用，Agent支持任意网络下数据采集。
- LogHub覆盖Kafka 100%功能，并提供完整监控、报警等功能数据，弹性伸缩等（可支持PB/Day规模），使用成本为自建50%以下。
- LogSearch/Analytics 提供保存查询、仪表盘和报警功能、使用成本为自建 20%以下。
- 30+ 接入方式，与云产品（OSS/E-MapReduce/MaxCompute/Table Store/MNS/CDN/ARMS等）、开源软件（Storm、Spark）无缝对接。

生态丰富

- LogHub 支持30+采集端，包括Logstash、Fluent等，无论是从嵌入式设备，网页，服务器，程序等都能轻松接入。在消费端，支持与Spark Streaming、Storm、云监控、ARMS等对接。
- LogShipper 支持丰富数据格式（TextFile、SequenceFile、Parquet等），支持自定义Partition，数据可以直接对接Presto、Hive、Spark、Hadoop、E-MapReduce、MaxCompute、HybridDB等存储引擎。
- LogSearch/Analytics 查询分析语法完整，兼容SQL92，支持通过JDBC协议与Grafana对接。

实时性强

- LogHub：写入即可消费；Logtail（采集Agent）实时采集传输，1秒内到服务端（99.9%情况）。
- LogSearch/Analytics：写入即可查询分析，在多个查询条件下1秒可查询10亿级数据，多个聚合条件下1秒可分析1亿级数据。

完整API/SDK

- 轻松支持自定义管理及二次开发。
- 所有功能均可通过API/SDK实现，提供多种语言SDK，可轻松管理服务和百万级设备。
- 查询分析语法简单便捷（兼容SQL92），接口友好适合与生态软件对接（支持Grafana对接方案）。

应用场景

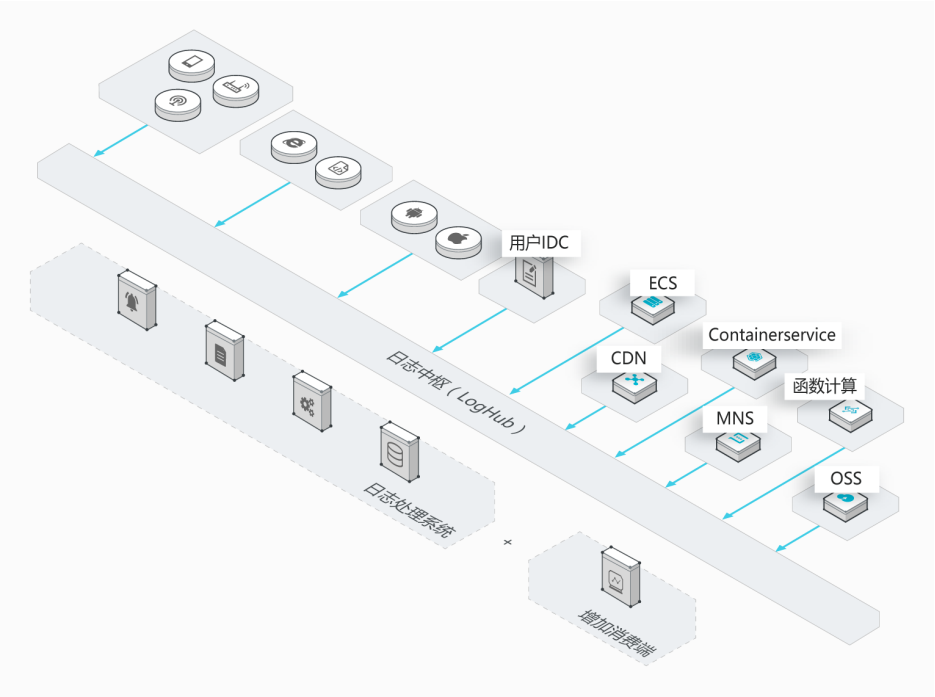
日志服务的典型应用场景包括：数据采集、实时计算、数仓与离线分析、产品运营与分析、运维与管理等场合。典型应用场景如下，更多应用场景参见最佳实践。

数据采集与消费

通过日志服务LogHub功能，可以大规模低成本接入各种实时日志数据（包括Metric、Event、BinLog、TextLog、Click等）。

方案优势：

- 使用便捷：提供30+实时数据采集方式，让您快速搭建平台；强大配置管理能力，减轻运维负担；节点遍布全国与全球
- 弹性伸缩：无论是流量高峰还是业务增长都能轻松应对



数据清洗与流计算 (ETL/Stream Processing)

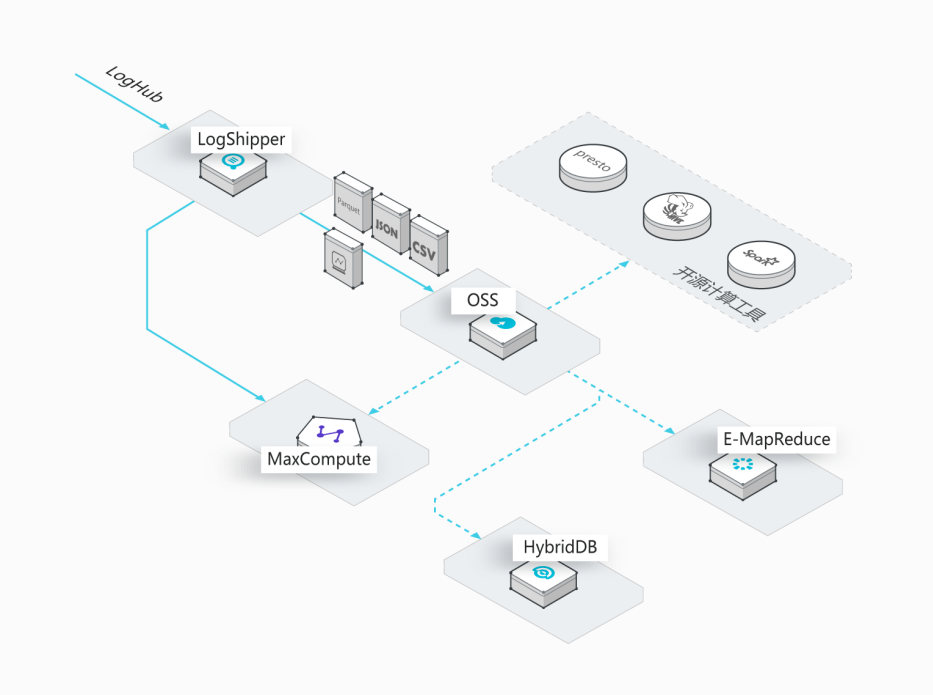
日志中枢（LogHub）支持与各种实时计算及服务对接，并提供完整的进度监控，报警等功能，并可以根据SDK/API实现自定义消费。

- 操作便捷：提供丰富SDK以及编程框架，与各流计算引擎无缝对接
- 功能完善：提供丰富监控数据，以及延迟报警机制
- 弹性伸缩：PB级弹性能力，0延迟

数据仓库对接(Data Warehouse)

日志投递（LogShipper）功能可以将日志中枢（LogHub）中数据投递至存储类服务，过程支持压缩、自定义Partition、以及行列等各种存储格式。

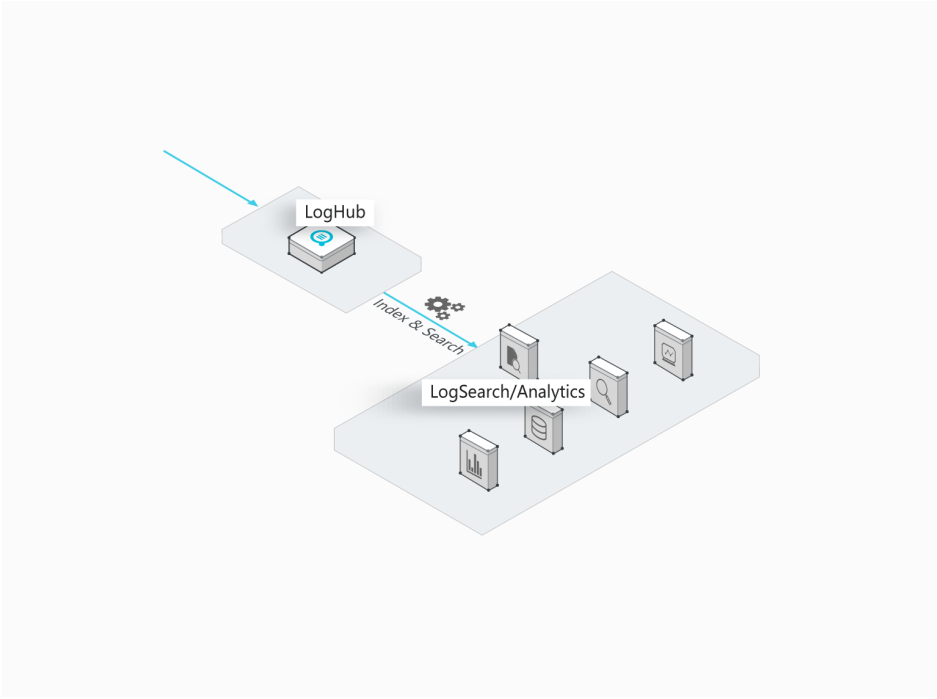
- 海量数据：对数据量不设上限
- 种类丰富：支持行、列、TextFile等各种存储格式
- 配置灵活：支持用户自定义Partition等配置



日志实时查询与分析

实时查询分析（LogAnalytics）可以实时索引LogHub中数据，提供关键词、模糊、上下文、范围、SQL聚合等丰富查询手段。

- 实时性强：写入后即可查询
- 海量低成本：支持PB/Day索引能力，成本为自建方案15%
- 分析能力强：支持多种查询手段，及SQL进行聚合分析，并提供可视化及报警功能



基本概念

简介

日志

日志（Log）是系统在运行过程中变化的一种抽象，其内容为指定对象的某些操作和其操作结果按时间的有序集合。文件日志（LogFile）、事件（Event）、数据库日志（BinLog）、度量（Metric）数据都是日志的不同载体。在文件日志中，每个日志文件由一条或多条日志组成，每条日志描述了一次单独的系统事件，是日志服务中处理的最小数据单元。

日志组

日志组即一组日志的集合，是写入与读取的基本单位。

日志主题

一个日志库内的日志可以通过日志主题（Topic）来划分。用户可以在写入时指定日志主题，并在查询时指定查

询的日志主题。

项目

项目（Project）是日志服务中的资源管理单元，用于资源隔离和控制。您可以通过项目来管理某一个应用的所有日志及相关的日志源。它管理着用户的所有日志库（Logstore），采集日志的机器配置等信息，同时它也是用户访问日志服务资源的入口。

日志库

日志库（Logstore）是日志服务中日志数据的采集、存储和查询单元。每个日志库隶属于一个项目，且每个项目可以创建多个日志库。

分区

每个日志库分若干个分区（Shard），每个分区由 MD5 左闭右开区间组成，每个区间范围不会相互覆盖，并且所有的区间的范围是 MD5 整个取值范围。

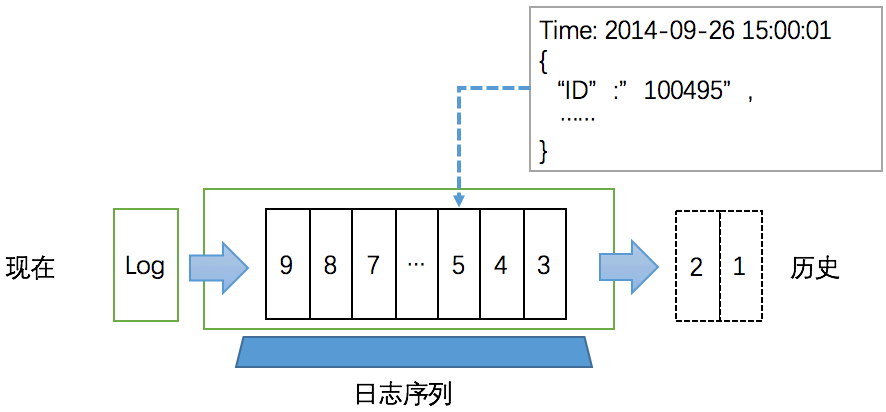
日志

半世纪前说起日志，想到的是船长、操作员手里厚厚的笔记。如今计算机诞生使得日志产生与消费无处不在：服务器、路由器、传感器、GPS、订单、及各种IoT设备通过不同角度描述着我们生活的世界。借助于计算力量，通过采集、处理、使用日志，我们不断更新对整个世界以及体系的认知。

日志是什么?

从船长日志中我们可以发现，日志除了带一个记录的时间戳外，可以包含几乎任意的内容，例如：一段记录文字、一张图片、天气状况、船行方向等。几个世纪过去了，“船长日志”的方式已经扩展到一笔订单、一项付款记录、一次用户访问、一次数据库操作等多样的领域。

日志这种广泛使用模式之所以经久不衰，在于“日志是一种简单的不能再简单的存储抽象”。它是一个只能增加的，完全按照时间排序的一系列记录。日志（时间序列数据）看起来如下：



我们可以给日志末尾添加记录，并且可以从左到右读取日志记录。每一条记录都指定了一个唯一的有一定顺序的日志记录编号。

日志顺序由“时间”来确定，从图上可以看到日志从右到左的时间顺序，新产生的事件被记录，过去的事件渐渐远去，但它记录了什么时间发生了什么事情，这无论对于计算机、人类、还是整个世界而言，是认知与推理的基础。

日志服务中的日志（Log）

日志（Log）是系统在运行过程中变化的一种抽象，其内容为指定对象的某些操作和其操作结果按时间的有序集合。文件日志（LogFile）、事件（Event）、数据库日志（BinLog）、度量（Metric）数据都是日志的不同载体。在文件日志中，每个日志文件由一条或多条日志组成，每条日志描述了一次单独的系统事件，是日志服务中处理的最小数据单元。

日志服务采用半结构数据模式定义一条日志。该模式中包含主题（Topic）、时间（Time）、内容（Content）和来源（Source）四个数据域。

与此同时，日志服务对日志各字段的格式有不同要求，具体如下表所示：

数据域	含义	格式
主题（Topic）	用户自定义字段，用以标记一批日志。例如访问日志可根据不同站点进行标记。	包括空字符串在内的任意字符串，长度不超过128字节。默认情况下，该字段为空字符串。
时间（Time）	日志中的保留字段（必选），用以表示日志产生的时间，一般由日志中的时间信息直接提取生成。	整型，Unix标准时间格式。单位为秒，表示从1970-1-1 00:00:00 UTC计算起的秒数。
内容（Content）	用以记录日志的具体内容。内容部分由一个或多个内容项组成，每一个内容项为一个Key-Value对。	Key为UTF-8编码字符串，包含字母、下划线和数字，且不以数字开头。长度不超过128字节。不可以使用如下关键字：__time__、__source__、__topic__、__partition_time__、_extract_others_、__extract_others__。Value 为任意字符串，长度不超过

		1024*1024字节。
来源（Source）	日志的来源地，例如产生该日志机器的IP地址。	任意字符串，长度不超过128字节。默认情况下该字段为空。

实际使用场景中，日志的格式多样。为了帮助理解，以下以一条nginx原始访问日志如何映射到日志服务日志数据模型为例说明。假设用户nginx服务器的IP地址为10.249.201.117，以下为该服务器的一条原始日志：

```
10.1.168.193 - - [01/Mar/2012:16:12:07 +0800] "GET /Send?AccessKeyId=8225105404 HTTP/1.1" 200 5 "-"
"Mozilla/5.0 (X11; Linux i686 on x86_64; rv:10.0.2) Gecko/20100101 Firefox/10.0.2"
```

把该条原始日志映射到日志服务日志数据模型，如下：

数据域	内容	说明
Topic	“”	沿用默认值，即空字符串。
Time	1330589527	日志产生的精确时间,表示从1970-1-1 00:00:00 UTC计算起的秒数。从原始日志中的时间戳转换而来。
Content	Key-Value对	日志具体内容。
Source	“10.249.201.117”	使用服务器IP地址作为日志源。

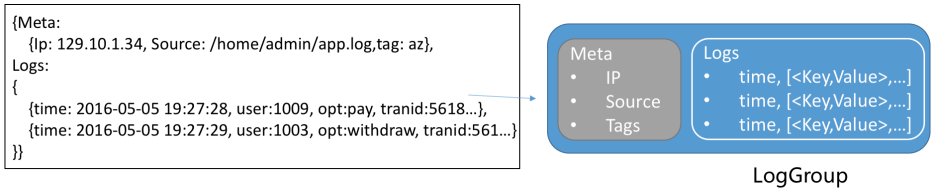
用户可以自己决定如何提取日志原始内容并组合成Key-Value对，例如下表：

Key	Value
ip	“10.1.168.193”
method	“GET”
status	“200”
length	“5”
ref_url	“-”
browser	“Mozilla/5.0 (X11; Linux i686 on x86_64; rv:10.0.2) Gecko/20100101 Firefox/10.0.2”

日志组

一组日志的集合，写入与读取的基本单位。

日志组的限制为：最大 4096 条日志，或 10MB 空间。



项目

项目（Project）是日志服务中的资源管理单元，用于资源隔离和控制。您可以通过项目来管理某一个应用的所有日志及相关的日志源。它管理着用户的所有日志库（Logstore），采集日志的机器配置等信息，同时它也是用户访问日志服务资源的入口。

具体来说，项目可以提供如下功能：

帮助您组织、管理不同的日志库。在实际使用中，您可能需要使用日志服务集中采集、存储不同项目、产品或者环境的日志。您可以把不同项目、产品或者环境的日志分类管理在不同的项目中，方便后续的日志消费、导出或者索引。同时，项目还是日志访问权限管理的载体。

为您提供日志服务资源的访问入口。每创建一个项目，日志服务会为该项目分配一个独有的访问入口。该访问入口支持通过网络写入、读取及管理日志。

日志库

日志库（Logstore）是日志服务中日志数据的采集、存储和查询单元。每个日志库隶属于一个项目，且每个项目可以创建多个日志库。您可以根据实际需求为某一个项目生成多个日志库，其中常见的做法是为一个应用中的每类日志创建一个独立的日志库。例如，用户有一个“big-game”游戏应用，服务器上有三种日志：操作日志（operation_log）、应用程序日志（application_log）以及访问日志（access_log），用户可以首先创建名为“big-game”的项目，然后在该项目下面为这三种日志创建三个日志库，分别用于它们的采集、存储和查询。

无论是写入或者查询日志，您都需要指定操作的 Logstore。如果您希望投递日志数据到 MaxCompute 做离线分析，其数据投递也是以 Logstore 为单元进行数据同步，即一个 Logstore 内的日志数据投递到一张 MaxCompute 的 Table。

具体来说，日志库提供如下功能：

- 采集日志，支持实时日志写入

- 存储日志，支持实时消费
- 建立索引，支持日志实时查询
- 提供投递到 MaxCompute 的数据通道

分区

Logstore读写日志必定保存在某一个分区（Shard）上。每个日志库（Logstore）分若干个分区，每个分区由MD5左闭右开区间组成，每个区间范围不会相互覆盖，并且所有的区间的范围是MD5整个取值范围。

分区范围

创建Logstore时，指定分区个数，会自动平均划分整个MD5的范围。每个分区均有范围，可用MD5方式来表示，且必定包含于以下范围中：[00000000000000000000000000000000,ffffffffffffffffffffffffffffffff）。

分区的范围均为左闭右开区间，由以下Key组成：

- BeginKey：分区起始的Key值，分区范围中包含该Key值
- EndKey：分区结束的Key值，分区范围中不包含该Key值

分区的范围用于支持指定Hash Key的模式写入，以及分区的分裂和合并操作。在向分区读写数据过程中，读必须指定对应的分区，而写的过程中可以使用负载均衡模式或者指定Hash Key的模式。负载模式下，每个数据包随机写入某一个当前可用的分区中，在指定Hash Key模式下，数据写入分区范围包含指定Key值的分区。

例如，某Logstore共有4个分区，且该Logstore的MD5取值范围是[00,FF)。各个分区范围如下表所示。

分区号	范围
Shard0	[00,40)
Shard1	[40,80)
Shard2	[80,C0)
Shard3	[C0,FF)

当写入日志时，通过指定Hash Key模式指定一个MD5的Key值是5F，日志数据会写入包含5F的Shard1分区上；如果指定一个MD5的Key值是8C，日志数据会写入包含8C的Shard2分区上。

分区的读写能力

每个分区可提供一定的服务能力：

- 写入：5MB/s，2000次/s

- 读取：10MB/s，100次/s

建议您根据实际数据流量规划分区个数，流量超出读写能力时，及时分裂分区以增加分区个数，从而达到更大的读写能力；如您的流量远远达不到分区的最大读写能力时，建议您合并分区以减少分区个数，从而节约分区租赁费用。

例如，如果您有两个readwrite状态的分区，最大可以提供10MB/s的数据写入服务，但如果您实时写入的数据达到14MB/s，建议分裂其中一个分区，使readwrite分区数量达到3个。如果您实时写入的数据仅为3MB/s，那么一个分区即可满足需要，建议您合并两个分区。

注意：

- 当写入的API持续报告403或者500错误时，通过 Logstore云监控查看流量和状态码 判断是否需要增加分区。
- 对超过分区服务能力的读写，系统会尽可能服务，但不保证服务质量。

分区的状态

分区的状态包括：

- readwrite：可以读写
- readonly：只读数据

创建分区时，所有分区状态均为readwrite状态，**分裂**或**合并**操作会改变分区状态为readonly，并生成新的readwrite分区。分区状态不影响其数据读取的性能，同时，readwrite分区保持正常的数据写入性能，readonly状态分区不提供数据写入服务。

在**分裂**分区时，需要指定一个处于readwrite状态的ShardId和一个MD5。MD5要求必须大于分区的BeginKey并且小于EndKey。分裂操作可以从一个分区中分裂出另外两个分区，即分裂后分区数量增加2。在分裂完成后，被指定分裂的原分区状态由readwrite变为readonly，数据仍然可以被消费，但不可写入新数据。两个新生成的分区状态为readwrite，排列在原有分区之后，且两个分区的MD5范围覆盖了原来分区的范围。

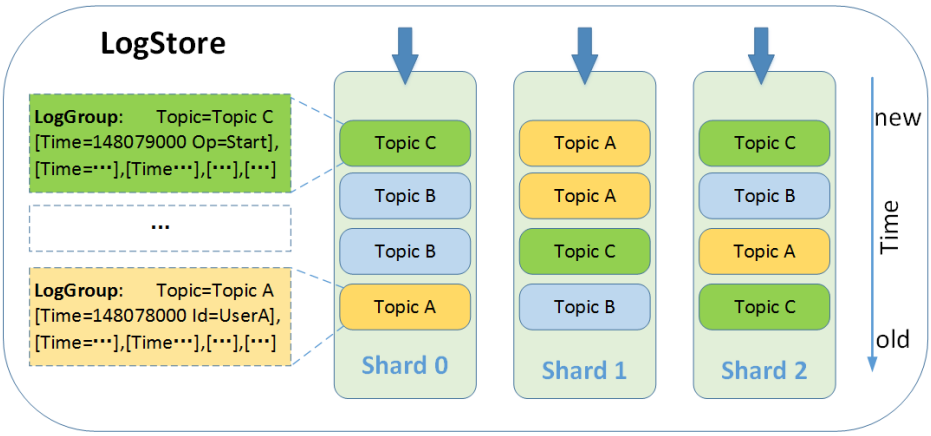
在**合并**操作时，必须指定一个处于readwrite状态的分区，指定的分区不能是最后一个readwrite分区。服务端会自动找到所指定分区的右侧相邻分区，并将两个分区范围合并。在合并完成后，所指定的分区和其右侧相邻分区变成只读（readonly）状态，数据仍然可以被消费，但不能写入新数据。同时新生成一个 readwrite 状态的分区，新分区的MD5范围覆盖了原来两个分区的范围。

日志主题

一个日志库内的日志可以通过日志主题（Topic）来划分。用户可以在写入时指定日志主题，并在查询时指定查询的日志主题。例如，一个平台用户可以使用用户编号作为日志主题写入日志。这样在查询时可利用日志主题让不同用户仅看到自己的日志。如果不需要划分一个日志库内的日志，让所有日志使用相同的日志主题即可。

注意：空字符串是一个有效的日志主题（Topic），且无论是写入还是查询日志时，默认的日志主题都是空字符串。所以，如果不需要使用日志主题，最简单的方式就是在写入和查询日志时都使用默认日志主题，即空字符串。

下图描述了日志库、日志主题和日志之间的关系：



发布历史

2018/01 详情

新地域

亚太南（孟买）访问入口

新功能

1. Log4J/LogBack/Consumer Library发布 使用案例
2. [新功能] IP来源分析函数+功能 文档
3. Logtail：MySQL Binlog采集 文档，使用案例
4. Logtail：JDBC查询结果采集 文档
5. Logtail：HTTP请求结果采集 文档
6. Logtail：Nginx监控数据采集 文档
7. [试用] OSS访问日志分析 介绍

功能优化

1. Go SDK支持Windows原生编译 Github

2. 命令行工具（CLI）升级 工具地址, 文档
3. 控制台新增数据源/图形/优化仪表盘性能。

2017/12 详情

新功能

1. 嵌入式/物联网IoT日志客户端（C-Producer）：面向嵌入式、物联网（IoT）客户端C-Producer Library定位为一个“轻量级Logtail”，虽没有实时配置管理机制，但具备除此之外70%功能，包括：多租户支持、分级处理、上下文查询、并发发送/断点续传。还有一些专门为IoT准备功能，大大降低数据采集门槛：本地调试、细粒度资源控制、日志压缩缓存。文档, 使用案例
2. Flink Connector：可以方便、快捷Flink开发环境中进行日志服务LogHub中数据开发，包括消费者(Consumer)和生产者(Producer)两个部分。 文档, 使用案例
3. 命令行工具（CLI）发布：跨平台命令行工具，支持常见所有API操作，并提供日志查询结果导出、批量创建、批量复制等功能。 工具地址, 文档
4. Nginx日志向导：在控制台上根据Nginx配置文件一键生成索引、视图、导入（OSS/MaxCompute）等选项，方便用户进行一站式配置。 使用案例
5. API网关日志向导：推出API网关日志一键打通功能，方便开发者进行日志分析。文档，使用案例
6. 告警功能升级：新增新增钉钉，自定义WebHook两种新通知渠道；对告警内容进行优化。 文档
7. 控制台优化：
    i. 查询页面新增快速分析功能：提供命中结果1W条数据Top10统计与分布
    ii. 新增数据接入向导（Wizard）：数据接向导（Wizard）功能，快速完成数据的采集、存储、分析、离线投递， 降低户使志服务槛
    iii. 自动刷新功能：持定时刷新功能，设置时定期刷新所有图表时间范围
    iv. 过滤功能：可以指定过滤条件进行视图生成逻辑

新地域

1. 美东（弗吉尼亚）访问入口

2017/11

免费额度（FreeTier)上线：每月 500MB 读写流量/索引流量/存储空间，31 个*天 活跃Shard，可满足大部分普通用户需求。

新功能

1. 支持自定义ETL处理（通过函数服务）：通过日志服务触发器与函数服务互通；提供常用ETL模板供用户直接使用。参见：功能介绍，最佳实践
2. 支持DataV：可以在DataV中配置阿里云日志服务数据源，通过查询与分析语法配置各种大屏视图，参见：使用文档
3. Python SDK 发布：支持完整的配置和日志操作接口，并支持实时可靠的消费组接口。参见：使用文

档

4. Python 版 Consumer Library：支持通过Python进行LogHub实时数据处理开发，过程中无需担心负载均衡，Failover以及弹性扩展等问题。参见：使用文档
5. 告警支持邮件及多人：告警功能与通知中心打通，可通过短信+邮件进行多人的告警通知管理。参见：使用文档
6. 查询分析能力加强：
    - 时间序列函数
    - 支持子查询
    - 支持IP统计: 内网流量/IP来源地（国家、省市、城市）/统计运营商

新地域

日本站上线

2017/10

新功能

1. 支持JDBC协议进行日志分析：提供JDBC协议与标准SQL92查询语法支持，能够通过程序、工具以及各种成熟的可视化、数据库连接产品等进行实时日志分析。参考文档
2. 支持通过Grafana进行日志分析：发布Grafana插件，用户可以在Grafana中配置阿里云日志服务数据源，通过查询与分析语法配置各种分析视图，支持交互式实时分析与展示。参考文档
3. 优化Android SDK体验：对数据采集接口进行优化，方便开发者通过模板快速上手
    i. Android
    ii. IOS

新地域

马来西亚，内蒙节点上线，参见地域列表。

2017/9

新功能

- 支持JDBC协议：通过SQL92标准语法对日志进行查询分析

2017/8

性能优化

- 对底层存储进行深度优化，分析性能提升1000倍，做到真正实时日志分析。参见场景：Nginx访问日

志分析，行车轨迹日志分析，销售营业日志统计
- 同样性能，更强计算能力+存储量，成本为自建ELK 10-13%，对比报告

新功能

- 仪表盘（Dashboard)功能：将查询分析另存为实时图表/报警，参见演示视频

Logtail

- Logtail新增本地状态查询功能:实时掌握Agent运行历史与现状

2017/6

新功能

- 日志服务OSS Shipper支持CSV格式投递：使用说明
    - 支持投递CSV格式数据，可以选择额外压缩。
    - 支持丰富的CSV格式选项，如delimiter、quote、escape、header、null等。
    - 存储于OSS的CSV数据可以通过HybridDB、MaxCompute、EMR或开源工具进行消费。
- 消费进度告警功能：使用说明
    - 在云监控中查看日志库（Logstore）下每个消费组（ConsumerGroup）消费点位与最新数据之间时间差，以衡量当前计算的实时性。
    - 支持对Storm、Spark Streaming、Flink等计算系统消费延迟监控与告警。
- Logtail ：
    - 采集目录支持通配符模式（参见文档）：日志采集目录支持通配符模式，便于用户以更细的粒度、更加灵活的方式指定日志采集目录。
    - 格式新增多字符分隔符模式（参见文档）：日志格式新增多字符分割符模式，相对复杂的单字符分隔符日志，配置方式更加简洁、对日志格式要求更低。
    - 上传支持绑定网卡（参见文档）：日志上传增加绑定指定网卡功能，适用于多网卡服务器中业务数据与日志数据分离、日志采集网络QoS控制等场景。

功能优化

优化上下文查询功能体验。

- 日志查询过程中还原任意一条日志精确上下文。
- 提供线程号字段过滤筛选功能。

2017/5 详情

新渠道

- 支持中信云售卖

新功能

- 支持SQL进行日志实时分析：使用例子、查询语法
    - 聚合查询支持自适应展示图表或原始日志。
    - 新增表格，饼图可视化结果展示。
- 完善Go SDK：增加Example、案例以及说明
- Logtail ：新增环境检查工具

限制调整

- 单用户：可创建Project上限调整为30个。
- 单Project：可创建Shard调整为200个。
- 日志保存天数：上限从90天调整为365天。

2017/4 详情

新地域

新增8个Region（完整Region入口列表），向国内用户提供完整基础设施出海，海外用户也可以直接使用阿里云全球节点进行服务

- 增加香港、迪拜、悉尼、法兰克福、东京、新加坡、美西硅谷
- 新增国内张家口节点
- 国际站/国内站均已对外开放

新功能

OSS投递支持Parquet存储（使用方法）

- 控制台直接配置即可生成开源Parquet格式文件，方便与开源社区对接
- Parquet存储对接E-MapReduce/MaxCompute消费方式（使用方法）

2017/3 详情

新功能

支持数值类（Double/Long）索引与查询，使用说明

- 对于日志中延时、位置和精度等数据提供多维度过滤
- 支持与文本类数据进行多维数据查询（最大30维）

计费调整

通过技术大幅优化降低了成本，启用新计费模式（3月20日生效，费用下降10%-85%），与其他方案成本对比：

- 与开源搭建的日志查询方案对比：是Elastic Search（ELK）成本 20%，Hive成本 50%
- 自定义TTL时长，具备100 PB 级长时间存储能力

2017/1

功能优化

- LogShipper新增投递OSS自定义Partition功能，根据计算需求定义存储格式，参考投递OSS章节
- 控制台新增Logtail错误诊断功能：快速诊断日志采集错误与原因，参考logtail采集错误查询
- 简化报警功能：无需通过RAM授权，具体请参考报警设置
- 优化日志查询页面体验：保存每页条数、排序方式、是否换行等参数后会持续生效

计费预告

计费模式即将调整：提供免费额度；索引费用下降（最大84%），参见计费模式预告

2016/12 详情

查询增强

- 性能提升10+倍：十亿级日志多关键词查询秒级返回（非Cache情况）
- 实时性提升（30S->1S)：99.9%场景写入1秒即可查询，最大3秒
- 新增上下文功能（业界首创）：定位关键日志时，提供严格精确上下文内容（无论是Docker、T4、ECS或物理机），参见上下文功能文档
- 新增模糊查询 (支持：*, ?等语法)，参见查询语法
- 新增多Topic、全Topic查询（空改为查询Logstore下所有Topic）,参见查询语法
- 新增根据时间段分布聚合功能,参见查询语法
- 成本优化：全文索引+键值索引只计算一份流量

存储增强

- 提供生命周期管理（TTL）：
    - 支持动态调整时长
    - LogHub生命周期与LogSearch统一
- 多拷贝技术/机架级容灾：
    - SLA > 99.95%
    - 数据安全性 > 99.99999999%

- 极为低廉成本（针对索引查询场景，参见对比）：
  - ELK（搜索方案）方案10%
  - Hadoop类方案30%

生态支持

- 对查询提供另存为功能，参见快速开始
- 对查询结果进行短信报警: 参见报警设置
- iOS(Swift/OC)/Android SDK升级：全面支持https

计费预告

计费模式即将调整：提供免费额度；索引费用下降（最大84%），参见计费模式预告

2016/09

产品发布

- 9.1 深圳金融云发布

新功能发布

- Windows版Logtail发布：稳定、高性能、行为与Linux版对齐，更多请参考

2016/07

产品发布

- 7.1 上海Region正式启用

新功能发布

- 发布Log4J Appender支持Java开发用户，使用手册
- 发布LogHub Producer Library支持客户端高并发写，使用手册
- 发布IOS/Android 客户端：iOS, Android
- Logtail支持根据IP等自定义字段，Hash映射Logstore Shard，实现保序功能，请工单联系我们开通
- LogHub与TS（表格存储）打通，使用手册

功能优化

- 控制台支持关闭全文索引、按需配置索引，参考步骤

2016/6

产品发布

- 6.12 日志服务正式商用

新功能发布

- 机器组支持设置用户自定义标示UserDefinedID，支持机器组自动扩容
- 与消息服务（MNS）/ CDN两个云产品日志打通，接入参考
- 发布Unity3D SDK，支持游戏开发用户采集日志
- Logtail支持日志转码/过滤功能

功能优化

- 与MaxCompute投递支持自动授权建表
- 控制台支持子账号操作Project

2016/05

新功能发布

- 支持Tracking Pixel方式接入日志：通过页面嵌入HTTP Tracking方式采集用户的访问行为数据，支持H5页面、以及各种移动端等
- logtail支持根据目录名生成Topic：支持Logtail从路径中提取内容作为Topic字段，区分同类型各种日志
- logtail支持JSON格式日志：Json格式输出日志可以直接被采集，无需进行结构化转化
- logtail支持分隔符(Delimiter)格式日志：支持直接解析分隔符标准的日志

2016/03

新功能发布

- oss shipper支持ram pass role检查，要求oss shipper配置者必须对所配置的role有访问权限，从而加强shipper安全。
- logtail支持接入syslog数据。
- loghub支持根据cursor反查数据时间。
- loghub支持拉指定时间窗口的数据。

2016/02

新功能发布

- 提供数据投递OSS功能，帮助用户在OSS做长期数据存储或通过其它系统（如E-MapReduce）消费OSS数据来发掘日志的更大价值
- 数据投递ODPS功能支持用户自定义ODPS表结构导入，实现日志服务到ODPS的字段级别的映射，用户使用ODPS加工数据更方便
- 支持Shard扩容、缩容、删除，Logstore资源具备弹性伸缩能力
- 发布基于新API的SDK，支持查看、重试错误ODPS/OSS投递任务，包含：Java, Python
- 对LogHub消费者提协同消费模式loghub consumer library，解决多个消费者同时消费logstore时自动分配shard、按序消费等问题，使用户只需专注在自己业务逻辑上，而无需关心shard分配、CheckPoint、Failover等事宜。

2016/01

行为变更

数据模型变更

- 数据模型变更，原API格式依然兼容，推荐用户使用新的API。

```
Category
```

变更为

```
LogStore
```

- SLS接口文档

离线投递行为变更

- 离线投递行为变更：由投递到ODPS公共表变更为直接导入用户指定表。在此之前，用户已经设定的投递到公共表配置依然在后台兼容，但不可再增加配置，推荐用户使用新的投递方式。
- 在ODPS中查看导入日志

预留写入吞吐容量变更

- 公测阶段，单个Project预留写入吞吐容量由10MB/s变更为1MB/Min。在此之前，用户已经建立的Project预留数值不变，即600MB/Min。

可创建Project数量变更

- 由允许用户创建1个Project，变更为允许用户创建10个Project。

新功能发布

- 发布基于新API的SDK，包含：Java, .NET, PHP, PythonSLS SDK
- 发布日志采集客户端Logtail的Windows版本Windows Logtail
- 支持key-value对的查询SLS 查询语法
- 提供离线投递任务管理页面，用于查询任务状态及失败重试。

2015/12

行为变更

新增LogHub接口

- 新增Shard（分区）概念，用以日志写入与读取消费，参见数据模型
- 新增数据订阅API，参考日志消费API

新功能发布

- 发布基于新API SDK，包含：Java, Python

客户端Logtail安装方式

- 安骑士远程安装功能已关闭，目前只支持自助安装Logtail客户端
- 具体安装方式请参考安装说明

2015/01 详情

新功能发布

发布基于新API的SDK，包含：Java, .NET, PHP, PythonSLS SDK

发布日志采集客户端Logtail的Windows版本Windows Logtail

支持key-value对的查询SLS 查询语法

提供离线投递任务管理页面，用于查询任务状态及失败重试。