文本复制检测报告单(全文标明引文)

№:ADBD2018R_2018053015312720180530154825440174024184　　检测时间:2018-05-30 15:48:25

检测文献:　53140204_关鸿睿_计算机科学与技术_基于Apache Flume的数据收集与分析系统的设计与实现

作者:　关鸿睿

检测范围:　中国学术期刊网络出版总库

中国博士学位论文全文数据库/中国优秀硕士学位论文全文数据库

中国重要会议论文全文数据库

中国重要报纸全文数据库

中国专利全文数据库

图书资源

优先出版文献库

大学生论文联合比对库

互联网资源(包含贴吧等论坛资源)

英文数据库(涵盖期刊、博硕、会议的英文数据以及德国Springer、英国Taylor&Francis 期刊数据库等)

港澳台学术文献库

互联网文档资源

CNKI大成编客-原创作品库

个人比对库

时间范围:　1900-01-01至2018-05-30

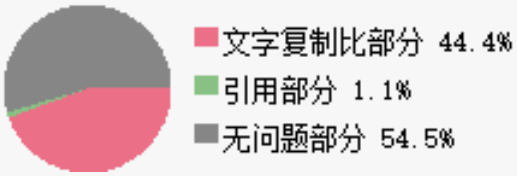
检测结果

总文字复制比：45.5%　　跨语言检测结果：0%

去除引用文献复制比：44.4%　　去除本人已发表文献复制比：45.5%

单篇最大文字复制比：14%（求助hadoop2.X分布式搭建两个NameNode均无法正常启动 - 大数据之美 - CSDN博客）

重复字数：	[28075]	总段落数：	[10]
总字数：	[61678]	疑似段落数：	[7]
单篇最大重复字数：	[8658]	前部重合字数：	[1106]
疑似段落最大重合字数：	[12521]	后部重合字数：	[26969]
疑似段落最小重合字数：	[313]		



指　标：☐疑似剽窃观点　☑疑似剽窃文字表述　☐疑似自我剽窃　☐疑似整体剽窃　☐过度引用

表格：0　　公式：0　　疑似文字的图片：0　　脚注与尾注：0

（注释：无问题部分 文字复制比部分 引用部分）

1. 53140204_关鸿睿_计算机科学与技术_基于Apache Flume的数据收集与分析系统的设计与实现_第1部分

总字数：1371

相似文献列表 文字复制比：0%(0) 疑似剽窃观点：(0)

原文内容 红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

吉林大学学士学位论文（设计）承诺书

本人郑重承诺：所呈交的学士学位毕业论文（设计），是本人在指导教师的指导下，独立进行实验、设计、调研等工作基础上取得的成果。除文中已经注明引用的内容外，本论文（设计）不包含任何其他个人或集体已经发表或撰写的作品成果。对本人实验或设计中做出重要贡献的个人或集体，均已在文中以明确的方式注明。本人完全意识到本承诺书的法律结果由本人承担。

学士学位论文（设计）作者签名：

2018年5月20日

摘要

基于Apache Flume的数据采集和分析系统的设计与实现

如果说没能真切感受到世纪交接时那历史瞬间是一种遗憾的话，那么信息时代的鼎盛将为我们弥补这一切。

当古老简易的传呼机变成现代精致的智能手机；当折叠封装的亲笔信变成方方正正的聊天气泡；当千里之外的思念变成近在眼前的喜悦，我们正在享受信息时代为我们带来的无限便利。

随着信息时代的到来，数据的井喷式增长也引起了人们的关注。数据的传输从原来的寥寥几字，演变成大批量传输，到现在的TB乃至PB量级，数据的指数型增长速度令人咂舌。

Hadoop的出现为我们提供了一种新的数据处置方式——分布式。HDFS（分布式文件系统）的广泛应用更是帮助我们更加从容的面对大数据时代带来的挑战。

作为一个近些年出现的全新分布式日志收集系统，Apache Flume表现出了独特的活力。从数据的采集到分析，基于Apache Flume的系统更加高效、精确，得到更广泛的使用。

关键字：信息时代，大数据，Apache Flume

Abstract

If it is a pity that there is no sense of truth in the passing of the century, the peak of the information age will make up for it.

When the old, simple pager became a modern, sophisticated smartphone; When folded and encapsulated letters become square and positive chat bubbles; We are enjoying the infinite convenience brought by the information age when

the yearning of thousands of miles away becomes the joy that is close at hand.

The advent of Hadoop provides a whole new way of data processing - distributed processing. The wide application of HDFS (distributed file system) has helped us to face the challenges of the big data era more easily.

As a new distributed logging collection system in recent years, Apache Flume has shown unique vitality. From data collection to analysis, the system based on Apache Flume is more efficient, accurate and widely used.

关键词：信息时代，大数据，apache flume。

目录

2. 53140204_关鸿睿_计算机科学与技术_基于Apache Flume的数据收集与分析系统的设计与实现_第2部分

总字数：2132

相似文献列表　文字复制比：33.6%(717)　疑似剽窃观点：(0)

1	大数据研究综述 卿勇; -《软件导刊》- 2016-12-29 1	32.8%（699） 是否引证：是
2	11650050_张欢欢_链家网大数据整合与应用分析 张欢欢 -《大学生论文联合比对库》- 2017-06-05	23.8%（508） 是否引证：否
3	大数据技术研究综述 张锋军; -《通信技术》- 2014-11-10	12.9%（274） 是否引证：否
4	2013447017-倪凯丽-智能终端用户的大数据分析方法及应用 倪凯丽 -《大学生论文联合比对库》- 2017-05-16	2.4%（52） 是否引证：否
5	连锁零售品类分析模型设计与实现 刘藤伟泰 -《大学生论文联合比对库》- 2016-04-24	1.7%（36） 是否引证：否
6	连锁零售品类分析模型设计与实现 刘藤伟泰 -《大学生论文联合比对库》- 2016-05-04	1.7%（36） 是否引证：否

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

第一章绪论

1.1研究背景

随着信息时代的到来，互联网技术得到了迅猛的发展和普及。从经济发展到军事建设，从大国外交到人际交往，从工作生产到日常生活，互联网算扮演的角色和发挥的作用愈加地举足轻重。

互联网带来的好处与便利显而易见，但是，瑕瑜互见，互联网的弊端也日益凸显，“斯诺登”事件使得全世界开始注意数据的安全性，网络平台的个人信息泄露事件也屡见不鲜。大数据时代个人私密信息应该如何保护？如何精准地传输海量数据？如何提高效率？如何保证传输过程中数据的完整性和安全性？所有这些都在考验时代，考验互联网工作者。

全球的网站数不胜数，在纷乱复杂的网页中寻找有价值的信息和数据成为了互联网用户乃至互联网巨头们的共同目标。因此，建立一个高效的数据采集和分析系统的重要程度便凸显出来。

2003，谷歌发布了谷歌文件系统，引入了谷歌海量数据处理中使用的文件系统，这使得互联网时代的数据存储发生了革命性的变化。

Doug Cutting等人将GFS的MapReduce逻辑应用到NUCH项目中，并发展成Hadoop项目。经过多年的发展，已经形成了包含多个相关项目的软件生态系统，并创造了海量数据处理的新局面。

Hadoop是为解决Internet时代海量数据的存储和处理而设计和开发的。简洁地说，Hadoop是一种分布式计量平台，可以与大规模数据并行地开发和处理。其主要特点是：可扩展性高、效率高、成本低。目前，Hadoop用户已经从传统的互联网公司扩展到科学计算、电信、电力、生物和金融公司，并得到了越来越广泛的应用。

1.2*国内外研究现状*

信息与网络的飞速发展，信息量大量增长；计算机硬件成本逐渐降低，使得昂贵的数据存储和处理变得经济。谷歌的

MapReduce、GFS和BigTable等核心技术引起了雅虎、Facebook等互联网公司的注意，为目前应用最广泛的开源大数据框架Apache Hadoop的诞生奠定了基础。联合国发布的《大数据促进发展：挑战与机遇》大数据政务白皮书指出，大数据对人类而言是一个历史性的挑战和机遇。[1]

1.2.1 总体概况

互联网最受欢迎的特点是社交媒体，如国内微博、微博客圈、博客和论坛，以及脸谱网、Twitter和Instagram等。

进入信息时代后，数据挖掘面临着新的挑战。因为大多数数据挖掘都停留在传统的固定模式中。如今的Web模式多样性增加，数据量级急剧增长，分布范围扩大到整个因特网。传统的数据挖掘模式已不能适应新的挑战。因此，我们需要研究新的算法和模型，以适应大数据时代。同时，在研究新的数据挖掘模式时，数据量级的提升，结构的复杂化，需要更高的系统性能，因而带来的开销也随之增加。开发高性能、低成本的数据挖掘系统已成为当务之急。

1.2.2国外研究现状

美国政府耗费巨资投入大数据技术研究，颁布了《大数据研究和发展计划》，目标是通过大数据技术实现感知、认知和预测支持的结合，增强信息提取分析、情报获取和对目标的洞察能力，培养该领域的技术人才[2].投入155个项目涉及国家多个重要领域，如国防部、能源部以及国家安全及未来发展战略等。主要项目包括：多尺度异常检测项目（ADAMS）、网络内部威胁计划（CINDER）、加密数据的编程计算项目（PROCEED项目）、视频与图像检索分析工具项目（VIRAT项目）等[2].为实现决策优化，美国还进行了数据可视化、信息安全与大数据结合等方面的综合研究，建立大数据中心，对各类大数据进行整合、分析，并向相关领域提供大数据分析产品[2]。

1.2.3国内研究现状

我国大数据应用还处于起步阶段，但已有国际知名项目投入使用，如Facebook开发的社交图谱数据、NSA棱镜计划、IBM Waston等项目。2013年，我国开始进行大数据专项研究，2014年，国内主要互联网公司已将大数据应用于相关业务中，取得了巨大的经济和社会效益[3]。同年，清华大学开设了大数据相关课程，正式开启了培养大数据领域专业人才的序幕[3]。（引用）

“数据存储与管理云服务环境”子项目建立了一个科学的数据存储基础设施，具有1个主要的1 + 12子中心50PB容量，为科学数据领域提供云存储和云计算服务，并提供稳定和易于使用的数据库结构。科研工具和自组织的科研社群建设工具。已经形成了支持科学的数据库性能评估和科学数据发布的功能，支持了空间科学试点、卫星遥感、微生物等领域的数据存储，部署了17个单元的私有云，分别放在5个领域。

1.3论文的主要工作和组织结构

本文主要介绍大数据时代潮流下，数据的收集、分析等处理过程。介绍Apache Flume的主要组成部分及功能。并基于Apache Flume设计实现日志收集和分析系统，对系统性能进行测试。

1.4本章小结

本章主要介绍了当前大数据时代下，数据在人们的生活、生产和工作中所扮演的重要角色。大数据为我们带了无数的机遇，更是带来了挑战。在时代的洪流中把握前进的方向。

同时也介绍了为了更好地适应大数据时代，更充分地享受大数据带来便利，也更好的避免随之而来的威胁，互联网工作者们所作的努力。以及近些年取得卓越的成果。

第二章 Apache Flume的介绍及数据采集和分析系统的工作

2.1 Apache Flume 是什么？

Apache Flume是一种高性能的分布式日志系统。现在是Apache的TOP项目。类似于水槽的日志采集系统，有脸谱网的ScRebe，Apache Cukka，Apache Kafka（LinkedIn），是一颗冉冉升起的星。

Apache Flume是一种分布式的、可靠的、易于使用的系统，它可以有效地收集、收集或传输大量来自不同的同源系统的日志数据到数据中心存储器。

Apache水槽的作用不限于日志摘要，因为数据源是可定制的，并且水槽可以用来传输大量的事件数据，包括但不限于网络流量数据、社交媒体生成的数据、电子邮件和几乎所有可能的数据。

flume提供了从控制台（控制台）、RPC（节俭RPC）、文本（文件）、尾部（UNIX尾）、Syslog（Syslog日志系统、TCP和UDP）等2种模式收集数据的能力，以及其他数据源，如（命令执行）[4]。（《消息存储转发类系统向云特性架构演进的设计与实现》

）同时，水槽的数据接收器可以是控制台、文本、DFS、RPC和SyScript。

作为Cloudera平台的一部分，Flume可以轻松与其他组件（例如Apache Kafka和Spark Streaming）协同工作，在单一平台内构建完整的流式工作负载[5]。它还受益于统一资源管理（通过YARN），简单的部署和管理（通过Cloudera Manager）以及共享的合规性安全和治理（通过Apache Sentry和Cloudera Navigator） - 这些都是生产运行的关键[6].

Flume的主要特点有：

一、有效流数据

轻松收集，汇总和移动来自多个源的流日志或事件数据到Hadoop中。作为构建完整的流处理流水线的关键部分，Flume设计用于在为近实时分析生成数据时进行采集 - 使其成为传感器数据聚合或“物联网”用例的理想选择[7].

二、为Hadoop规模而构建：

随着流数据的增长，您可以简单地进行水平缩放以处理增加的负载 [8]。您还可以扩展到许多数据源，以高效地从多个系统或传感器收集日志，并且可以使用连接器将数据流式传输到多个系统[8]。

三、始终在线的可靠性：

防止数据丢失，并确保流式数据即使在发生故障时也能继续提供，并且内核具有容错能力并具有可调的可靠性以最大限度地满足您的需求[9]。

作为将日志和事件数据流式传输到Hadoop的标准工具，Flume是构建端到端流式工作负载的关键组件，其典型用例包括：

·欺诈识别

·物联网应用程序

·传感器和机器数据的汇总

·警报/ SIEM[10]

AgentAgent

HDFSHDFSWebserverWebserver

2.2 Apache Flume的具体组件

水槽采用分层结构，分别为代理、收集器和存储。其中，Agent是用来收集数据的，Agent是水槽中生成数据流的地方。同时，代理将生成的数据流发送到收集器。相应地，采集器被用来聚集数据，通常导致更大的流量。水槽传输数据的基本单元是事件，它包含字节负载和一些可选的字符属性。事件也可以是文本文件，通常是记录，它是事务的基本单元。事件本身是一个字节数组，并承载头信息，这是数据流的最小单位。

2.2.1 Agent

flume作业的核心是Agent。代理包括三个核心组件，即源、信道和接收器。在Flume，代理对应于收集器，而源对应于接收器。源和汇强调发送和接收方的特性，如数据格式、编码等，而代理和收集器则涉及功能。

图 Agent架构原理图

AgentAgent

ChannelChannelSinkSinkSourceSource

图 Agent架构原理图

SinkSinkChannelChannel

SinkSinkChannelChannel

SourceSource

SinkSink

ChannelChannel

一、Source

信源负责收集数据并将数据划分为事务和事件。

Source访问客户终端操作的消耗数据源，水槽支持AVRO、Log4J、SysLog和HTTP POST（正文是JSON格式）。允许应用程序直接与现有的源代码交互，如AVRESURCE，SyslogTcpSource。您还可以编写一个以IPC或RPC的方式访问自己的应用程序的源码，AVRO和节俭都可以（NETTyavrRPCOclipse和TraceFrPC客户端分别实现RPC客户端接口），而AVro则是默认的RPC协议。特定代码级客户端数据访问

Flume内置了大量的Sourece，其中Avro Source、Thrift Source、Spooling Directory Source、Kafka Source具有较好的性能和较广泛的使用场景

Avro Source

AVRO源监听AVRO端口，接收外部AVRO客户端发送的AVRO事件数据。在多级流中，AVRO源可以与以前的水槽代理的AVRO接收器相匹配，从而建立一个分层的收集拓扑。

Avro是Hadoop中的子项目，是Apache中的独立项目。AVRO是一种基于二进制数据传输的高性能中间件。该工具还用于Hadoop的其他项目，如HBASE（REF）和HIVE（REF）。AVRO是一种数据系统，系统的特性表现在具有串行化性质。

AVRO可以将数据结构转换格式，一般转换后的格式便于保存和传输。在最初的设计阶段， Avro的功能主要是用于密集的数据应用，适合于远距离或本地大规模数据保存和转换。AVRO具有多种多样的数据类型。快速、可压缩二进制数据格式在数据二进制串行化后，可以节省数据存储空间和网络传输带宽。用于存储长期使用数据的文件器皿；可以实现远距离过程调用RPC。

Avro支持跨编程语言的实现（C，C++，C＃，java，Python，Ruby，PHP），但有个显著的特点：Avro依赖模式，动态加载相关的数据模式，数据读写操作的Avro频繁，而这些操作NS都是减少对每个数据文件的写入的模式。序列化的成本是快速和轻的。这种数据及其模式的自我描述有助于使用动态脚本语言。AVRO数据存储在一个文件中，与此同时被保存记录的还有他的模式，这种做法的目的在于是的其它所有程序均可对文件进行操作处理。如果在读取数据时使用的模式不同于写入数据时使用的模式，则也容易解决，因为已知的读写模式。

Avro定义了八种简单的数据类型；

·null： no value

·boolean： a binary value

·int： 32-bit signed integer

·long： 64-bit signed integer

·float： single precision (32-bit) IEEE 754 floating-point number

·double： double precision (64-bit) IEEE 754 floating-point number

·bytes： sequence of 8-bit unsigned bytes

·string： unicode character sequence

Thrift Source

thrift源接收由外部thrift客户发送的thrift事件数据。在多级流中，thrift源可以与先前的水槽代理的thrift汇配对，以建立分级收集拓扑。

AVRO和thrift是跨语言、基于二进制的高性能的通信中间组件。它们都提供RPC服务和数据序列化机制。总体功能是相似的。thrift来自于脸谱网在各种服务之间的交流背景。thrift的设计强调统一编程接口的多程序通信框架。

然而，thrift支持更多的语言。在现阶段，thrift支持C++、C＃，可可，Erlang，Haskell，java，Ocami，Perl，PHP，PHP，Perl，等等。

在数据结构方面，Thrift与Avro十分接近。

二、Channel

通道：类似于工作栈，临时存储数据源发送的数据，等待下沉的调用；

通道有多种方式：内存通道、JDBC通道、记忆恢复通道、文件通道。

Memory Channel

由于内存运行，实现了高速吞吐。但是，这也是内存通道最大的缺点。它不能持久。一旦出现电源故障、停机等紧急情况，无法保证数据再重新启动之后不发生丢失的现象，因而无法保证数据的完整。内存通道最重要的部分是三个接口：channel（s）通道、Transaction事物和configuration配置。

1、通道接口主要声明了三种方法：PUT、PATH和GETPACTION。

2、事务接口主要声明了四种事务处理机制：事务启动、提交、故障回滚和关闭。

具体代码实现如下：

```
enum TransactionState { Started, Committed, RolledBack, Closed }
void begin();
void commit();
void rollback();
void close();
```

这四种方法指定了事务的四种状态。

3、Configurable接口只声明了一个context方法，作用于获取配置信息。可配置接口与水槽的主要配置部件相关。需要从水槽配置系统获取配置信息的任何组件都必须实现接口。

Configurable接口的参数如下：

·容量能力：默认情况下，MMRROY信道为100。

·跨容量：每笔交易的最大容量，即每个交易所能获取的最大事件数。默认也是100。

· 参考文献

卿勇．大数据研究综述[EB/OL]http://www.fx361.com/page/2017/0121/622976.shtml 2017-01-21

卿勇．大数据研究综述[EB/OL]http://www.fx361.com/page/2017/0121/622976.shtml 2017-01-21

卿勇．大数据研究综述[EB/OL]http://www.fx361.com/page/2017/0121/622976.shtml 2017-01-21

袁野《消息存储转发类系统向云特性架构演进的设计与实现》[D]大连海事大学2016-01-01

维基百科[EB/OL] https://zh.wikipedia.org/wiki/Apache_Hadoop

维基百科[EB/OL] https://zh.wikipedia.org/wiki/Apache_Hadoop
维基百科[EB/OL] https://zh.wikipedia.org/wiki/Apache_Hadoop
维基百科[EB/OL] https://zh.wikipedia.org/wiki/Apache_MapReduce
维基百科[EB/OL] https://zh.wikipedia.org/wiki/Apache_MapReduce
致谢
至此，毕业设计即将完成，四年的大学生活也要画上句点。激动和喜悦、期待与梦想、努力与汗水，都不及一声感谢。
首先，要感谢我的毕业设计导师王康平老师。对于我来说，大数据是一个全新的领域，也是毕业设计的一次艰难挑战。所以，能够顺利完成本次毕业设计王康平老师的帮助与支持无可替代。从论文选题的细心讲解，到开题报告的严谨指点；从中期检测的悉心教诲，到最后定稿的无微不至。王老师的教导与鼓励一直帮助我前行。在此，特向王老师致以崇高的敬意与感谢。
其次，要感谢计算机学院的各位尊敬的老师。感谢老师四年以来孜孜不倦的教导。感谢老师们带我走进计算机的世界，四年的知识灌溉才有我的今天。没有各位老师的领路指向，求知的路途将多有坎坷。感恩老师们的关心与呵护，未来的道路将一马平川。
还要特别感谢一下许兰东师兄，在王老师工作繁重辛劳的时候，为我们答疑解惑。毕业设计能够顺利完成，许师兄也功不可没。
同时，要感谢身边的朋友和一直站在身后的家人。不论是学习还是生活，感谢他们的不顾一切的支持与毫无怨言的付出。他们是我直面挑战坚实不倒的后盾，更是我前进路上一往无前的勇气。
最后，衷心感谢研究室的各位老师和为审查论文辛勤工作的各位老师。
字节容量缓冲百分比：定义通道中的事件百分比，我们需要考虑页眉中的数据。
·Byte Capacity：byte Capacity等于设置的byte Capacity值或堆的80%乘以1减去byte Capacity Buffer Percentage的百分比，然后除以100
·keep-alive：增加和删除一个Event的超时时间（单位：秒）
·初始化具有容量大小的Link键块DeGK对象。以及各种信号量对象。
·最后初始化计数器。
具体代码实现如下：
public class MemoryChannel extends BasicChannelSemantics {
私有静态记录器Logger-= LoggEngPrime.getLogger(MemoryChannel.class);
私有静态最终整数默认容量 = 100；
私有静态最终整数Debug TracabaPosiv= 100；
私有静态最终双字节码大小 = 100；
私有静态最终长Debug TytCePaCuth=（long）（运行时）。GetRuntime（ ）maxMemory() * .
私有静态最终整数Debug TytEcPaCaseBuffel% = 20；
私有静态最终整数Debug TekEng活着 = 3；
4、Memory Channel的工作流程主要分为三个部分：
第一部分
定义一些默认配置信息、通道、事务处理大小等信息；在同一代码中定义了特定代码和可配置接口的参数。

指　标
疑似剽窃文字表述
---
1. 程序均可对文件进行操作处理。如果在读取数据时使用的模式不同于写入数据时使用的模式

4. 53140204_关鸿睿_计算机科学与技术_基于Apache Flume的数据收集与分析系统的设计与实现_第4部分 总字数：449
相似文献列表　文字复制比：0%(0)　疑似剽窃观点：(0)
原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容
第二部分
实现Channel的put、get方法和事务的commit、rollback方法，这些方法是通过内部类MemoryTransaction实现
基于事务容量TracaPosipe创建了两个阻塞的双端队列PotListand TakelTIST，主要用于事务处理。当事件事件从源到通道被放置时，事件首先被放置在PATLIST队列中（相当于临时缓冲队列），然后放置在PUPIST队列中的事件。进入内存通道队列；当数据从通道发送到接收器时，首先将事件放入TaelKIST队列，然后从TaelKIST队列发送事件到接收器。不管是发生还是发生，都调用回滚方法来回滚事务，当回滚时，首先防止通道锁有线程访问，如果TAKLIST不是空的，那么将TAKELIST中的事件再次放入通道，然后删除所有TH。在PUTLIST中的事件（即，删除写入PIDLIST临时队列的事件）。 从

上边代码发现这里只是具体方法的实现，实际的的调用是发生在 Source 端写事件和 Sink 读事件时，也就是事务发生时。

5. 53140204_关鸿睿_计算机科学与技术_基于Apache Flume的数据收集与分析系统的设计与实现_第5部分

总字数：5797

相似文献列表　文字复制比：5.4%(313)　疑似剽窃观点：(0)

1	Hadoop：从初出茅庐的小象变身行业巨人--相关文章 -《互联网文档资源（http://www.360doc.co）》- 2012	5.0%（288） 是否引证：否
2	阿里巴巴数据交换平台——集大成于一身--相关文章 -《互联网文档资源（http://www.360doc.co）》- 2012	5.0%（288） 是否引证：否
3	Hadoop版本选择探讨--相关文章 -《互联网文档资源（http://www.360doc.co）》- 2014	5.0%（288） 是否引证：否
4	RPC, Serialization and Schema--相关文章 -《互联网文档资源（http://www.360doc.co）》- 2010	5.0%（288） 是否引证：否
5	windows/linux服务器程序支持库的开发（2）--相关文章 -《互联网文档资源（http://www.360doc.co）》- 2012	5.0%（288） 是否引证：否
6	[爆料]英特尔：将Hadoop“固化”到Xeon中--相关文章 -《互联网文档资源（http://www.360doc.co）》- 2013	4.8%（279） 是否引证：否
7	Hadoop，有所为而有所不为--相关文章 -《互联网文档资源（http://www.360doc.co）》- 2012	4.8%（279） 是否引证：否
8	美国大数据工程师面试攻略--相关文章 -《互联网文档资源（http://www.360doc.co）》- 2014	4.8%（279） 是否引证：否
9	HBase核心贡献者Ted Yu：参与开源比收入更重要--相关文章 -《互联网文档资源（http://www.360doc.co）》- 2012	4.8%（279） 是否引证：否
10	hadoop优化二--相关文章 -《互联网文档资源（http://www.360doc.co）》- 2014	4.8%（279） 是否引证：否
11	Java跨语言调用实现方案	淘宝JAVA中间件团队博客--相关文章 -《互联网文档资源（http://www.360doc.co）》- 2010	4.8%（279） 是否引证：否
12	hadoop公共模块RPC实现机理 - sxf--相关文章 -《互联网文档资源（http://www.360doc.co）》- 2010	4.8%（279） 是否引证：否
13	关于thrift--相关文章 -《互联网文档资源（http://www.360doc.co）》- 2011	4.8%（279） 是否引证：否
14	对象序列化 - h345210sun的专栏 - CSDN博客--相关文章 -《互联网文档资源（http://www.360doc.co）》- 2009	4.8%（279） 是否引证：否
15	Hadoop的Python语言封装--相关文章 -《互联网文档资源（http://www.360doc.co）》- 2010	4.8%（279） 是否引证：否
16	Eliot - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》- 2013	2.2%（125） 是否引证：否
17	2013年05月存档 - Eliot - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》- 2013	2.2%（125） 是否引证：否
18	2013年05月存档 - Eliot - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	2.2%（125） 是否引证：否
19	2016年09月存档 - leo - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	2.2%（125） 是否引证：否
20	2015年08月存档 - AlvinTech14的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	2.2%（125） 是否引证：否
21	2014年08月存档 - twsghxs的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	2.2%（125） 是否引证：否
22	架构 - 光线技术博客 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	2.2%（125） 是否引证：否
23	基于云计算技术的化合物相似性分析系统 李杰辉(导师：张亮) -《复旦大学博士论文》- 2012	2.0%（117） 是否引证：否
24	云队列：一个基于Hadoop的大规模消息基础平台 史冬冬(导师：施霞萍) -《东华大学博士论文》- 2012	1.0%（56） 是否引证：否

第三部分

通过配置和初始化内存通道获取配置文件系统。配置文件的获取有两种方式。在启动时只读一次或动态加载配置文件。

这里特别介绍一下动态加载配置文件的方法：

动态加载配置文件需要动态改变Channel的容量大小，此时会调resizeQueue 方法进行调整。

显然，动态容量调整有三种情况：

·新老容量相同，直接返回

·老容量大于新容量，则进行缩容操作，需先给未被占用的空间加锁，防止在缩容时有线程再写入数据，然后创建新的容量队列，将原本队列中所有的 event 添加至新队列中

·老容量小于新容量，则进行扩容操作，然后创建新容量的队列，将原本队列加入中所有的 event 添加至新队列中

File Channel

同样作为通道，File Channel（文件通道）弥补了Memory Channel最大的缺点。File Channel将所有数据写入磁盘，因而当发生断电或宕机等紧急状况时不会丢失进程或数据，所以文件通道是Flume的持久通道。

同时，只有当接收到事件并提交事务时，文件通道才能从通道中移除事务，即使机器或代理崩溃或重新启动，文件通道也可以确保数据的完整性不会受到损坏。因此，在处理多个源和汇时，文件通道被设计为高度并发。此外，由于文件通道将数据写入磁盘，因此大大提高了信道容量。特别是，与存储通道相比。

此外，只要磁盘空间足够，文件通道的最大容量就可以达到数亿个事件。比如当预期从渠道获取的汇款将无法跟上有限的高峰期，并且大量积压。此时，File Channel便可发挥其超大容量的作用，临时存储数据。与此同时，正确的配置也可以解决理更长时间的下游停机问题。由于在事件提交后，信道不将内存存储在内存中，因此它占用的空间比等效容量存储信道少得多。

文件通道保证通过代理和机器故障或重新启动将写入的每个事件都可用。它是通过编写将信道放入磁盘的每个事件来实现的。一旦提交事务，该事务中的事件就可用于执行。这些事件是从磁盘读取的，当从信道获取时，它们被传递给接收器，并且被完全取消。在提交交易后，他们有资格被删除。

文件通道允许用户通过在不同的安装点安装多个磁盘来配置多个磁盘。当配置为使用多个磁盘时，信道在磁盘之间循环，允许信道在更多磁盘可用时性能更好。建议（但不是必需的）使用单独的磁盘来进行文件通道检查点。检查点反映了检查点通道的确切状态。文件通道使用检查点快速重启而不必读取所有的数据文件。它在运行时将检查点写入磁盘。当重新启动时，信道加载最后一个写入的检查点，只重放放样，并且在检查点之后执行，并且允许通道快速启动并准备好进行正常操作。通常情况下，即便它是可配置的，任意两个连续的检查点相隔的时间间隔设置为30秒。

三、Sink

从通道中删除数据信息，并将数据引入到其他存储文件系统、数据库或远程服务器中，如HDFS、记录器、AVRO、节俭、IPC、文件、NULL、HBASE、SOLR等。

1.Logger Sink

记录信息级别，通常用于调试。源中使用的接收器是这种类型的接收器。

必须配置的属性：

属性说明：

!channel –

!类型-组件类型名称，需要记录器

maxBytesToLog 16 Maximum number of bytes of the Event body to log

要求必须在 --conf 参数指定的目录下有 log4j的配置文件

它可以通过-dFLUME。logger=INFO,console在命令启动时手动指定log4j参数

2.File Roll Sink

在本地文件系统中存储事件。每次它被指定，它就成长为一个文件来存储在此期间收集的日志信息。

属性说明：

!channel –

!类型，必须是“文件卷”

!目录——存储在文件中的目录

sink.RollInterval每隔30秒滚动30个文件（它应该是30秒的单独数据到一个文件的含义）。如果设置为0，则不允许滚动，从而将所有数据写入文件。

sink.序列化文本其他可能的选项包括AVROCYPER事件或FQCN的FQCNBuilder接口。

batchSize 100

2.2.2拦截器、通道选择器与选择处理器

拦截器是数据流中的一个point，存在于事件的内部，具体位置在Source与Channel之间。用于对事件的修改和检查。拦截器可以不用设置，也可以设置多个拦截器。（在2.7中继续介绍）

通道选择器的作用顾名思义，在数据传输过程中，通道选择器负责将数据传个某个或某些通道。通道选择器的选择非常灵活，可以使用flume自带的通道选择器，也可以使用个人定义的通道选择器。同时，多路通道选择器与拦截器的配合，使得数据可以输入到不同的通道之中。（在2.8中继续介绍）

选择处理器是一种恢复机制，通过创建恢复路径来应对故障事件。

2.3 HDFS接收器

HDFS接收器的工作方式是不断地打开HDFS中的文件，将数据流持续写入。可以根据需要，在特定的事件关闭当前文件并打开下一个新的文件，以达到存储的目的。

使用HDFS接收器的前提要求配置sink，将接收器类型设置为hdfs。

然后，还要指定路径，也就是配置path，指定数据写入的位置。路径有三种选择，分别是相对路径、绝对路径、带有Server名称的绝对路径。

最后一步是配置Channel信息。

2.3.1 path与文件名

当flume向HDFS传输数据后，HDFS需要将数据存入文件中。此时，新开启一个文件，文件名的创建，需要hdfs.filePrefix、点符号、文件打开时间戳以及一个指定文件后缀组成。

当使用时间的不断增加，hdfs.path的目录将会写满，因此，需要根据写入文件的时间元素创建子目录。Flume中含有基于时间的转义方法，可以解决创建子目录的问题。

为了提高效率，flume支持同时写入若干文件，但是出于开销的考虑，每次同时写入的上限为5000。

当写入操作完成时，会在文件后增加拓展名.tmp，关闭文件时将自动删除。

2.3.2文件转储

一般情况下，flume可以间隔30秒、间隔10个事件或者1024字节将写人的文件转储，甚至也可以禁止转储，只要将对应的属性值设为0即可。

有时输出的文件包含头数据，那么对应的HDFS会比预期略大，因为HDFS的转储只考虑事件本体的长度。因此，如果禁用转储会导致HDFS只含有一个目录且文件愈来愈大。

最后介绍hdfs.batchsize参数。此参数标识接收的事件数目，如果Channel中数据过多，可将参数设置成大于100，可以降低代价，提升性能。

2.4压缩编解码器1

编码器用于通过各种压缩算法来压缩和解压缩数据。Flume支持gzip、bzip2、lzo和snappy，不过你需要自己安装lzo，特别是在使用了CDH分发版本的情况下，这是由于许可问题造成的。

如果希望HDFS接收器写入的是压缩文件，你需要指定数据的压缩方式，这是通过设置hdfs.codeC属性实现的。该属性还用作写入到HDFS的文件的前缀。比如，如果像下面这样指定了编解码器，那么所有写入的文件都会带有一个.gizp扩展，因此在这种情况下就无需指定hdfs.fileSuffix属性了。

选择使用哪个编解码器需要做些调研。对于使用gzip还是bzip2存在一些争议，因为他们拥有更高的压缩率，但压缩时间却更长，特别是数据写一次而要读成百上千次的情况。另一方面，使用snappy或lzo的压缩性能会更高，不过压缩率却降低了。请记住，文件的分割性（特别是使用纯文本文件）会极大地影响MapReduce job的性能。

2.5事件序列化器

作为Flume的一种机制，序列化器将事件转换为其他格式再输出，类似于LayOut类。在一般情况下，text序列化器仅仅输出事件体，不会输出头部信息，输出头信息的任务由另一个序列化器完成，即header and text，同时它也会输出事件体信息。

2.5.1文本输出

文本输出即指默认的text序列化器，它只输出文本信息而舍弃头信息。

2.5.2 输出文本带有头信息

Text_with_headers可以将头信息与事件体信息一起输出而不是将其丢弃。其输出信息的格式为：头信息+空格+体信息。在输出序列最后需要用禁用字符终止。

2.5.3 Apache Avro（引用http://aoyouzi.iteye.com/blog/2292417）

Avro是Hadoop中的一个子项目，也是Apache中一个独立的项目，Avro是一个基于二进制数据传输高性能的中间件。在Hadoop的其他项目中例如HBase(Ref)和Hive(Ref)的Client端与服务端的数据传输也采用了这个工具，Avro可以做到将数据进行序列化，适用于远程或本地大批量数据交互。在传输的过程中Avro对数据二进制序列化后节约数据存储空间和网络传输带宽。做个比方：有一个100平方的房子，本来能放100件东西，现在期望借助某种手段能让原有面积的房子能存放比原来多150件以上或者更多的东西，就好比数据存放在缓存中，缓存是精贵的，需要充分的利用缓存有限的空间，存放更多的数据。再例如网络带宽的资源是有限的，希望原有的带宽范围能传输比原来高大的数据量流量，特别是针对结构化的数据传输和存储，这就是Avro存在的意义和价值。Avro还可以做到在同一系统中支持多种不同语言，也有点类似Apache的另一个产品：Thrift(Ref)，对于Thrift不同的是Avro更加具有灵活性，Avro可以支持对定义的数据结构(Schema)动态加载，利于系统扩展。

2.5.4 文件类型

大多数情况下，HDFS写入数据的方式是通过HDFS接收器以Hadoop SequenceFiles的方式写入。Hadoop SequenceFiles由一个键位和一个二进制字段与记录分隔符分隔的值域组成。一般来说，计算机的文本认为每个换行符确定一条记录，为了避免数据中存在换行符，人们研究出了序列化文件的办法，通过使用无法打印字符进行分隔。

常用的文件类型有以下几种：

1、 序列文件

2、 数据流

3、 压缩流

2.6 接收器组

出于可靠性的考虑，避免管道中的单点失败，接收器组的出现，使得flume具有均衡负载或故障恢复能力。接收器组可以创立逻辑接收器组，由接受处理器控制，决定事件的路由方式。

单个接收器是不需要配置的，但是接收器组需要声明。Sinkgroups是由flume提供的全新顶层属性，可以用于定义接收器组。

2.7拦截器

在之前的部分简单介绍了拦截器的功能。这里介绍拦截器的种类。

拦截器主要负责Flume中事件的传递。一般情况下，传入的是事件，返回的值也是事件，拦截器可以对事件进行修改，也可以将其原封不动的传递下去。如果返回值为空，则说明事件被舍弃。添加拦截器的方法就是添加Interceptor属性，先定义拦截器的名字，再定义拦截器的属性。

Flume中常用的拦截器有以下几种：

·Timestamp拦截器

·Host拦截器

·Static拦截器

·正则表达式过滤

·正则表达式抽取

2.8 通道选择器

如图所示，agent可以有单通道模式（channel）和多通道模式（channels），一个Source可以只对应一个通道，一个源也可以对应多个Channel。因此，对应的结构也有相应的处理方式。事件可以在进入所有通道也可以写入指定的一个通道。写入指定通道的方法就是通过Flume的header信息确定的。这种机制被称为通道选择器。

2.9 ETL

ETL指数据流分层。数据流分层的目的在于限制直连到Hadoop的Agent数量，并以此来限定并行请求的个数；还可以解决维护Hadoop是应用Server上磁盘空间不足的问题。

2.10监控FLUME

顾名思义，Flume的监控选项执行着监听Flume的工作的任务。对于单一的工作服务器，我们可能实时监控工作状况，发生故障可以技术处理。但是，在实际应用过程中，会有成千上万台服务器同时工作，人工监控无法面面俱到，所以Flume需要监控选项，将所有服务器的信息汇总，发现异常及时解决。

2.10.1 监控Agent

作为Flume的核心组件，当然需要被当作监控的首选目标。因为只有代理一直保持正常运转状态，Flume才能正常工作，是整个数据收集和分析系统的基石。监控系统的选择非常多，不论怎么样，监控系统最终就是一个小屏幕。

介绍两种常用的监控系统：Monit和Nagios。

Monit有一个免费的使用版本，可以实时监控Flume Agent是否运行，如果不是运行状态，就重启，而且会以电子邮件的形式通知，可以根据邮件信息分析进程死掉的原因。除此之外，还可以自主添加Monit的功能。

Nagios与Monitor相似，可以通过邮件或SNMP发出异常信息，监控代理的工作状态。但是Nagios无法像Monit一样有重启功能。

2.10.2 监控数据分析

虽然监控系统的使用非常简单，但是，想要明确知道代理的工作情况的具体问题出在哪里，还需要一些严谨的参数数据来反映。所以，监控系统要有明确的监控内容。比如下面三种参数：

·数据是否按照预设的速度传进源中

·数据与通道数量是否合适

·数据是否按照预设的速度传出接收器

2.11 本章小结

本章介绍了本论文最核心的研究部分——Apache Flume。作为现在Apache基金协会的顶尖项目，flume被对大数据有需求公司广泛地使用。

Flume的核心是代理，也就是agent。在Flume运行前对Agent进行配置，有三个主要参数，分别是Source、Channel和Sink。

除此之外，Flume还包含其它组件，其中事件、拦截器、接收器、通道选择器、接收器和接收器组，以及监控系统等等。下一章将会介绍flume所依赖的运行环境Hadoop以及分布式文件系统。

6. 53140204_关鸿睿_计算机科学与技术_基于Apache Flume的数据收集与分析系统的设计与实现_第6部分

总字数：4813

相似文献列表　文字复制比：30.2%(1452)　疑似剽窃观点：(0)

1	卢坪-20115376-基于Hadoop平台的桥梁健康监测系统设计与实现 卢坪 -《大学生论文联合比对库》- 2015-05-27	16.4%（789） 是否引证：否
2	分布式日志解析系统的设计与实现 李宁 -《大学生论文联合比对库》- 2015-06-02	11.6%（560） 是否引证：否
3	基于Spark内存计算的弹性大数据处理工具设计与实现 汤飞 -《大学生论文联合比对库》- 2016-05-15	11.5%（552） 是否引证：否
4	基于内容的社交网络社群发现和自动标注 施昊 -《大学生论文联合比对库》- 2014-05-28	11.1%（536） 是否引证：否
5	刘家财-毕业论文 刘家财 -《大学生论文联合比对库》- 2014-06-13	9.7%（468） 是否引证：否
6	云计算环境中海量数据的压缩存储研究 唐尚文 -《大学生论文联合比对库》- 2014-04-02	9.5%（459） 是否引证：否
7	大规模跨语言知识图谱中实体的重要性分析 张东升 -《大学生论文联合比对库》- 2014-05-27	9.4%（454） 是否引证：否
8	分布式网络爬虫研究 辛伟 -《大学生论文联合比对库》- 2015-05-28	9.4%（454） 是否引证：否
9	加密文件破解技术的研究 吉竹庆 -《大学生论文联合比对库》- 2016-05-26	9.1%（440） 是否引证：否
10	BX0902-30-叶扬-毕业设计 叶扬 -《大学生论文联合比对库》- 2013-05-28	8.8%（422） 是否引证：否
11	基于Hadoop的海量日志分析处理系统的设计与实现 刘旭 -《大学生论文联合比对库》- 2015-05-31	7.4%（358） 是否引证：否
12	基于Storm的短信诈骗拦截提示系统的设计与实现 李治梦 -《大学生论文联合比对库》- 2016-05-17	7.4%（356） 是否引证：否
13	基于Storm的短信诈骗拦截提示系统的设计与实现 李治梦 -《大学生论文联合比对库》- 2016-05-20	7.4%（356） 是否引证：否
14	基于大数据的多媒体benchmark研究与实现 叶海男 -《大学生论文联合比对库》- 2015-06-04	7.3%（351） 是否引证：否
15	hadoop家族介绍 - 永不言弃！ - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	7.1%（341） 是否引证：否
16	计算机与通信学院_杜红刚_11408300102 -《大学生论文联合比对库》- 2015-05-27	6.5%（315） 是否引证：否
17	计算机与通信学院_杜红刚_11408300102 -《大学生论文联合比对库》- 2015-06-09	6.5%（315） 是否引证：否
18	基于Hadoop的日志收集、存储、分析系统 陈峰 -《大学生论文联合比对库》- 2015-04-14	6.2%（300） 是否引证：否
19	HADOOP框架的理解 - wjlwangluo的博客 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	5.7%（276） 是否引证：否
20	1210512_代广泰_一种PageRank增量式算法的并行化实现 代广泰 -《大学生论文联合比对库》- 2016-05-10	5.3%（255） 是否引证：否
21	软件学院-软件工程-2010118082-刘文达 软件工程 -《大学生论文联合比对库》- 2014-05-12	5.2%（248） 是否引证：否
22	栾英英_2010055030_基于Hadoop YARN的MapReduce调度器分析（陈妍） 栾英英 -《大学生论文联合比对库》- 2014-06-13	4.9%（235） 是否引证：否
23	基于Hadoop大数据的单词计数计算 顺博尔 -《大学生论文联合比对库》- 2017-06-13	4.5%（215） 是否引证：否
24	9_顺博尔_基于Hadoop大数据的单词计数计算	4.5%（215）

第三章 Hadoop简介

3.1 Hadoop简史

在Hadoop的历史中，有两个人占据主导地位。分别是Lucene和Nutch。最初的时候Hadoop是Nutch的子项目，Nutch是Lucene的子项目。它们都是由Doug Cutting创始，然后不断演化到今天的系统

Lucene是引擎开发包，为用户提供基于Java的索引。Lucene可以嵌入各种应用中实现检索或索引。Nutch始于2002年，是一个搜索引擎应用。

但是随之而来的是大数据的快速发展，人们逐渐发现现有的架构无法适应日趋庞大的互联网用户群体和上亿的数据网络。Google公司最先发布了分布式文件系统的论文，称分布式系统（简称GFS）可以解决面临的问题。

2004年，Google在一次会议上向世人介绍了MapReduce。

Doung Cutting等人加入雅虎之后，经过一段时间的研究与雅虎的团队资源投入，在2006年，Hadoop系统正式成为一个可在网络上运行的系统[11]。

2008年，Hadoop被各大公司，如脸谱网、纽约时报等应用。就此成为Apache的顶端项目。特别是纽约时报，它使用运行在亚马逊的EC2云计算上的Hadoop，将4TB的报纸文档压缩，转换为用于Web的PDF文档，这个过程历时不到24小时，使用100台机器运行，这成为Hadoop一个良好的宣传范例[12]。

2008年，Hadoop正式走上舞台中央，大数据也开始进入Hadoop时代，大量的有关项目逐渐问世，例如HBase、ZooKeeper以及本篇文章重点介绍的Flume。

3.2 Hadoop 技术介绍

Apache Hadoop是一款支持数据密集型分布式应用程序并以Apache 2.0许可协议发布的开源软件框架。它支持在商品硬件构建的大型集群上运行的应用程序。Hadoop是根据谷歌公司发表的MapReduce和Google文件系统的论文自行实现而成[13]。所有的Hadoop模块都有一个基本假设，即硬件故障是常见情况，应该由框架自动处理[]13。

Hadoop框架透明地为应用提供可靠性和数据移动[14]。它实现了MapReduce的编程范式：应用程序被分区成许多小部分，而每个部分都能在集群中的任意节点上运行或重新运行,此外，Hadoop还提供了分布式文件系统，用以存储所有计算节点的数据，这为整个集群带来了非常高的带宽,MapReduce和分布式文件系统的设计，使得整个框架能够自动处理节点故障,它使应用程序与成千上万的独立计算的电脑和PB级的数据连接起来[14]。现在普遍认为整个Apache Hadoop“平台”包括Hadoop内核、MapReduce、Hadoop分布式文件系统（HDFS）以及一些相关项目，有Apache Hive和Apache HBase等等[15]。（维基百科https://zh.wikipedia.org/wiki/Apache_Hadoop）

3.3 MapReduce

3.3.1 MapReduce简介

MapReduce是Hadoop中的一个架构。显而易见，MapReduce由两部分组成，分别是Map（归纳）和Reduce（映射）。当前的软件实现是指定一个Map（映射）函数，用来把一组键值对映射成一组新的键值对，指定并发的Reduce（归纳）函数，用来保证所有映射的键值对中的每一个共享相同的键组[16]。（https://zh.wikipedia.org/wiki/MapReduce）

在设计分布式程序的时候，研发人员只需编写两个函数：map()和reduce()。

Map函数的功能是输入，输入的内容为键值，同时在这个过程中还会有一系列键值写入本地存储的磁盘空间之中，这部分输入将被用作中间输出。下一步就是聚集，这部分功能MapReduce的框架会自动实现，将键值聚集且key值相同。将数据传输给reduce函数处理。

Reduce函数接收来自map函数的数据，将key相同的值合并之后，输出新的key和value，最后写入HDFS。

3.3.2 MapReduce架构

1、Client []17

作为MapReduce的客户端，所有的程序将会通过此处传到Jobtracker端；此外，在Client端用户可以查看作业（Job）运行状态，因为Client端为用户提供了接口。在hadoop的系统中，MapReduce程序被称为“作业”（Job），同时，一个程序有相应的多个作业，而作业则对应着多个Task（任务）。

2、JobTracker[18]

JobTracker 接收来自Client端的程序。Jobtracker的主要工作是监控taskTracker和job的工作状态，如果发生意外情况，立刻转移对应的任务到其他工作节点。除此之外，JobTracker还承担着监控任务执行进度、资源使用情况的的任务，JobTracker会将检测到的信息传给任务调度器，而调度器会在需要的时候，也就是资源空闲状态，合理地分配这些资源。值得注意的是，调度器在hadoop中可以自定义，所以用户可根据需要定制自己的调度器。

4、 TaskTracker[19]

如MapReduce的架构图所示， TaskTracker用过Heartbeat与JobTracker进行交互。具体的交互内容为TaskTracker向jobTracker发送资源使用状况和任务的执行进度，JobTracker向TaskTracker发送相应操作的执行命令，比如开始新的任务、

杀掉任务等等。而且这一切的交互工作都是具有周期性的。

TaskTracker划分节点的资源量。这项工作有“Slot”完成。“slot”代表计算资源（CPU、内存等），一个 Task 获取到一个 slot 后才有机会运行，而 Hadoop 调度器的作用就是将各个 TaskTracker 上的空闲 slot 分配Task 使用[20]。slot 分为 Map slot 和 Reduce slot 两种，分别供 Map Task 和 Reduce Task 使用。TaskTracker 通过 slot 数目（可配置参数）限定 Task 的并发度[20]。

5、 Task [21]

在MapReduce中，Task任务有两种形式，一种是Map Task，另一种是Reduce Task。虽然种类不同，但是二者全部由TaskTracker启动。HDFS存储数据是以block为单位，且Block的大小固定。但是在MapReduce中，数据的单位是split。

作为逻辑概念，Split中只含有数据信息，例如数据的开始位置、数据大小以及数据的存储节点等。Split由用户自定义。唯一需要注意的是，Split与MapTask聚义一一对应的关系，因此Split决定MapTask的数量，每个MaptTask只处理对应的Split。

图 Hadoop MapReduce 架构图

ClientClient

JobTrakerJobTraker

ClientClientTaskScheduleTaskSchedule

ClientClient

HeartBeat HeartBeat HeartBeat

TaskTrackerTaskTracker

MapTaskMapTaskMapTaskkMapTaskk

ReducetaskkReducetaskkTaskTrackerTaskTrackerTaskTrackerTaskTracker

MapTaskMapTaskMapTaskMapTaskMapTaskMapTaskMapTaskMapTask

ReducetaskReducetaskReducetaskReducetask

3.4 HDFS

3.4.1 HDFS概括

HDFS全称Hadoop Distributed File System，是Hadoop框架内一个高效、容错率高的分布式文件系统。提供高吞吐量的数据流通，广泛的应用在各种大数据应用上。

3.4.2 HDFS架构

1、Client[21]

Client指代表用户，Client可以与NameNode和DataNode进行交互，以此访问HDFS中存储的文件。Client还提供了接口，接口类型类似于POSIX，供用户使用。

2、NameNode[21]

每个Hadoop系统只含有一个NameNode。它“统领”整个系统，执行HDFS的目录管理工作和有关文件信息的管理。这些信息一部分以HDFS源数据镜像文件的形式存在，另一部分则是HDFS文件改动日志，二者均存储于本地的磁盘空间，每次HDF重新启动，这两个文件也将会重新构建。同时，DataNode的健康工作状态也有NameNode负责监控，如果发现某个或多个DataNode发生宕机的状况，立刻将故障部分数据备份。

3、Secondary NameNode[21]

Secondary NameNode的任务经常被误认为是为NameNode的数据备份，实则不然，Secondary NameNode最关键的任务是定期将HDFS源数据镜像文件和HDFS文件改动日志合并，并将合并后的文件数据传给NameNode。需要特别注意的是，出于压力问题的考虑，NameNode不会自己将HDFS源数据镜像文件和HDFS文件改动日志合并，然后将其写入本地磁盘空间。这些任务都由Secondary NameNode

承担。

4、DataNode[21]

如图所示，一个slave节点一定安装一个DataNode，它的任务是实际数据的存储，然后将数据按时定期传输到NameNode。上面介绍过，HDFS中文件的基本单位是Block，因此，DataNode也以大小固定的Block为单位，在没有任何要求的情况下，每个Block大小为64MB。因此，当用户上传到HDFS上的文件过大时，文件便会被分割成多个Block，然后写入不同的DataNode；同时，为了确保数据的可靠性，同一个Block会写入3个不同的DataNode上。而且这一系列的分隔以及存储过程对于用户来说都是透明的。

图 HDFS架构图

3.5 本章小结

本章介绍了Hadoop的前世今生，从最初的传统数据收集模型，为处理海量数据而生的分布式文件系统，Hadoop的出现可谓千呼万唤始出来。

Hadoop集群的主要组件是HDFS、MapReduce还有Hadoop的核心。

HDFS是Hadoop分布式文件系统。HDFS的主要组成架构包括：Client、NameNode、Secondary NameNode和

DataNode

与其它系统相比，具有很明显的优势，这些优势体现在高容错率，高吞吐量以及高效性。在Flume数据系统中，HDFS负责接收来自flume的文件信息，并对文件进行处理。

MapReduce的主要工作是数据的分析工作。MapReduce的主要架构组成包括：Client、JobTracker、TaskTracker和Task。

MapReduce主要包含两个关键函数Map()和Reduce()，两个函数之间信息交互完成相应的功能。

MapReduce包含很多编译模型，通过对数据进行处理分析，获得数据下隐藏的信息。MapReduce的主要功能有数据划分和计算任务调度、数据/代码互定位、系统优化、出错检测和恢复[22]。

https://baike.baidu.com/item/MapReduce/133425?fr=aladdin

指　标
疑似剽窃文字表述

1. 2 Hadoop 技术介绍
   Apache Hadoop是一款支持数据密集型分布式应用程序并以Apache 2.0许可协议发布的开源软件框架。它支持在商品硬件构建的大型集群上运行的应用程序。
2. 转移对应的任务到其他工作节点。除此之外，JobTracker还承担着监控任务执行进度、资源
3. 任务的执行进度，JobTracker向TaskTracker发送相应操作的执行命令，比如开始新的任务、
4. slot 分为 Map slot 和 Reduce slot 两种，分别供 Map Task 和 Reduce Task 使用。
5. 数据信息，例如数据的开始位置、数据大小以及数据的存储节点等。Split由用户自定义。唯一需要注意的是，Sp
6. DataNode也以大小固定的Block为单位，在没有任何要求的情况下，每个Block大小为64MB。因此，当用户
7. 不同的DataNode上。而且这一系列的分隔以及存储过程对于用户来说都是透明的。
   图 HDFS架构图
   3.
8. 组成架构包括：Client、NameNode、Secondary NameNode和DataNo

7. 53140204_关鸿睿_计算机科学与技术_基于Apache Flume的数据收集与分析系统的设计与实现_第7部分

总字数：6711

相似文献列表　文字复制比：58.9%(3952)　疑似剽窃观点：(0)

1	南京第五十五所技术开发有限公司 版本XD-BigData-v1.0 发布日期2015.10... -《互联网文档资源（http://www.worlduc.c）》- 2016	48.8%（3273） 是否引证：否
2	hadoop中exmaple运行参数分析 - 记事本 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	48.7%（3266） 是否引证：否
3	1214490035_王鹏_基于Hadoop云平台构建及应用 王鹏 -《大学生论文联合比对库》- 2016	48.3%（3240） 是否引证：否
4	1214490035_王鹏_基于Hadoop云平台构建及应用 王鹏 -《大学生论文联合比对库》- 2016	48.3%（3240） 是否引证：否
5	1214490035_王鹏_基于Hadoop云平台构建及应用 王鹏 -《大学生论文联合比对库》- 2016	48.3%（3240） 是否引证：否
6	Ubuntu系统下使用eclipse搭建Hadoop2.7.1运行环境 - cuihaolong的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	46.4%（3111） 是否引证：否
7	2013001862_王彦杰_基于大数据的高校学生成绩聚类分析研究 王彦杰 -《大学生论文联合比对库》- 2017	39.7%（2667） 是否引证：否
8	Intellij IDEA远程向hadoop集群提交mapreduce作业 - u011654631的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	38.2%（2562） 是否引证：否
9	计信_1015080113_李光敏 计信 -《大学生论文联合比对库》- 2014	33.5%（2251） 是否引证：否
10	MapReduce的工作机理及其应用研究 陈坚钊(导师：谢维波) -《华侨大学博士论文》- 2013	33.3%（2233） 是否引证：否
11	hadoop相关(以期为单位） - anhuidelinger的专栏 - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》- 2013	32.8%（2199） 是否引证：否
12	20110321034_杨雪娇_基于物联网与大数据技术的教学楼外来人员监控系统 杨雪娇 -《大学生论文联合比对库》- 2015	32.7%（2197） 是否引证：否
13	Hadoop框架研究	31.7%（2127）

	毕赢之 - 《大学生论文联合比对库》- 2014	是否引证：否
14	Myeclipse 的hadoop环境搭建 - okman1214的专栏 - 博客频道 - CSDN.NET	31.6%（2122）
	- 《网络（http://blog.csdn.net）》- 2013	是否引证：否
15	使用Eclipse编译运行MapReduce程序 Hadoop2.7.1/Ubuntu - fluery - CSDN博客	31.6%（2122）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
16	hadoop学习笔记：mapreduce框架详解 - 走在学习的路上 - CSDN博客	31.6%（2122）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
17	MapReduce剖析笔记之一：从WordCount理解MapReduce的几个阶段 - Mirage520的专栏 - CSDN博客	31.6%（2122）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
18	Hadoop集群	31.6%（2119）
	- 《网络（http://blog.csdn.net）》- 2013	是否引证：否
19	Hadoop集群Eclipse开发环境的配置_conan	31.6%（2119）
	- 《网络（http://blog.sina.com）》- 2013	是否引证：否
20	HADOOP的学习笔记（第四期）eclipse 执行 wordcount . - kandy的专栏 - CSDN博客	31.6%（2119）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
21	我的Hadoop之路——Win7 Eclipse开发环境搭建 - maojunjiemomo的专栏 - CSDN博客	31.6%（2119）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
22	windows 32 eclipse 远程hadoop开发环境搭建 - 风在身后的专栏 - CSDN博客	31.5%（2117）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
23	MapReduce例子1--wordcount - Tear的专栏 - 博客频道 - CSDN.NET	30.9%（2076）
	- 《网络（http://blog.csdn.net）》- 2013	是否引证：否
24	Hadoop阅读笔记（一）——强大的MapReduce - codepython的专栏 - CSDN博客	30.9%（2074）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
25	Hadoop DistributedCache使用及原理 - leishenop的专栏 - CSDN博客	29.1%（1955）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
26	基于大数据的日志管理系统的设计与实现	1.0%（69）
	牟肖蓬(导师：卢朝霞) - 《东北大学博士论文》- 2014	是否引证：否
27	适用于地震数据处理的分布式执行控制系统的设计与实现	0.7%（49）
	陈曦(导师：胡光岷) - 《电子科技大学博士论文》- 2013	是否引证：否

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

第四章数据采集及分析系统的主要框架和构建

整个系统主要由日志发送模块、日志收集模块、日志存储统计模块（HDFS，MapReduce）三个模块组成。

4.1系统总体设计目标

4.1.1总体要求

在典型的互联网应用中，有必要收集尽可能多的数据来描述用户行为和身份。这些数据基本上是日志类数据。一旦写入，它们将不再被修改。但是写作的数量和速度都很高。Flume是一个分布式日志采集系统。本课题的目标就是搭建一个分布式日志采集系统，并测试它的性能，构建几个典型的数据分析类型来展示此系统的效果。

该系统由两部分组成，即日志系统的建立和测试。在分布式环境下基于HDFS搭建一个日志采集系统，并构建压力测试模型。测试系统的性能

数据分析的显示和应用。用几个典型的数据分析的实例，来展示Apache Flume使用的典型场景和方法。同时，学习如何使用数据显示工具。

4.1.2 系统特性

(1) 可靠性

当节点失败时，日志可以传输到其他节点而不丢失它。渡槽提供了三个级别的可靠性选项，分别从强到弱：端到端（接收到的数据代理首先写入事件到磁盘，当数据传输成功时，然后删除；如果数据失败，则可以再次发送。存储在故障（这也是抄写器采用的策略，当数据接收器崩溃，数据写入本地，恢复，继续发送），尽最大努力（在数据发送到接收器，没有确认）。其中，端到端采用磁盘记录和接收终端ACK，保证水槽接收到的数据最终达到目的。当存储失败时，数据将保留在本地硬盘上。不同于端到端，如果出现问题，存储在失败时可能会丢失一些数据。最大努力不做任何QoS保证。

(2) 可扩展性

flume采用三层结构，即Agent、集合体和存储，每一级都可以水平扩展。其中，所有代理和收集器均由主机统一管理，这使得系统易于监视和维护，并且主机允许多个（动物园管理员管理和负载平衡）避免单一故障点。

为了保证可扩展性，水槽采用多主模式。为了保证配置数据的一致性，水槽引入了动物园管理员来保存配置数据，动物园管理员本身保证了配置数据的一致性和高可用性，并且动物园管理员可以在配置数据改变时通知水槽主节点。

(3) 可管理性

所有的代理和COLLATR都由主机管理，这使得系统易于维护。在多主机的情况下，水槽使用动物园管理员和八卦来保证动态配置数据的一致性和高可用性。用户可以查看主数据源上的每个数据源或数据流的执行，并可以配置和动态加载每个数据源。渡槽提供Web和Shell脚本命令两种形式的数据流管理。

(4) 功能可扩展性

基于java，用户可以添加各种新功能的水槽。例如，通过继承源，用户可以实现自己的数据访问模式，实现Sink的子类，用户可以将数据写入特定的目标，同时，用户可以通过SinkDecorator预处理数据。用户可以根据需要添加自己的代理、收集器或存储。此外，水槽有许多组件，包括各种代理（文件、系统日志等）、收集器和存储（文件、HDFS等）。

4.2环境搭建

·操作系统版本：ubuntu

·Hadoop版本：2.7.5

·Jdk版本：jdk1.8.0

·安装flume版本：FLUMED-DISPLATION -0。4-bin

4.3 Flume Client

在实践中，数据生产是非常多样化的。人们希望选择一种合适而有效的方式。虽然，Flume有许多内置的机制来收集数据。但是，用户更倾向于将Flume于应用程序直接相连来发送数据。RPCEclipse的出现解决了这个问题。

渡槽的RPCEclipse实现了水槽的RPC机制。用户的应用程序可以简单地调用FLUME客户端SDK的附加（事件）或AppEntFrand（列表<事件>）方法来发送数据，而不必担心底层信息交换的细节。用户可以通过直接实现事件接口来提供所需的事件，例如使用EngEngIdEnter简单、方便地实现SimeEngEnter类或RealeBooT（ ）（ ）静态助理方法。

那么，什么是RPC呢？

RPC被称为远程过程调用，它是由Biell和Nelson在1984分布式计算领域引入的。这是解决上述访问透明度的一套极好的解决方案。随着RPC技术的发展，出现了大量的相关技术，如XML- RPC、JSON-RPC（JavaScript对象NoCTRPC）、CORBA（公共对象请求代理体系结构、公共对象请求代理体系结构）等。

简单地说，RPC是允许程序调用其他计算机（或同一台计算机的不同进程）。当机器A上的进程调用机器B上的进程时，A上的调用进程被暂停，并且在B上执行调用进程。调用方使用参数向调用方发送信息，然后通过返回的结果获取信息。在这个过程中，A是RPC客户端和B是RPC服务器。同时，程序员不能看到任何消息传递，它们的行为就像是一个进程调用另一个进程。

传统的调用过程中，主程序将参数压进栈内并调用过程，这时主程序处于停止执行当前过程并执行相应的调用过程。调用过程从堆栈中获取参数，然后执行进程函数。执行后，参数将返回到堆栈或存储在寄存器中，并且控制将返回给调用方。调用方获取返回参数并继续执行。

RPC跨越不同的过程。当客户端需要调用远程进程时，参数通过CyLeNT打包成消息，并附加调用进程的名称，指示在服务器上调用哪个进程，然后将消息发送到服务器。客户端发送消息后，必须等待服务器回复。此时，执行流是空闲的。

在接收到的消息的调用中，RPC的服务器运行时间被阻塞，并且当它接收到来自客户端的请求时，它被解包以获得请求参数，该请求参数类似于传统的过程调用，该调用由调用函数调用以从堆栈A接收参数。然后确定调用过程的名称（指示调用服务器的哪个进程）并调用该阶段。过程。调用完成后，将返回的值打包并通过主程序发送回客户端，并通知客户端调用的结束。

4.4 Flume 服务器

作为整个系统的核心，水槽服务器负责收集数据。正如之前所介绍的一样，日志收集模块由Source、Channel和Sink组成。

根据Flume Client的Sink接口类型配置Source接口，可选接口类型有AvroSource和ThriftSource等。

Flume服务器的核心为agent，将Flume的安装包解压到指定文件夹，配置环境，根据系统需求，对Flume进行重构。

4.5日志存储模块

日志存储模块建立在HDFS的基础上，负责接收来自日志收集模块的数据并存储。

这一部分系统的关键毫无疑问就是HDFS接收器。

HDFS接收器的工作方式是不断地打开HDFS中的文件，将数据流持续写入。可以根据需要，在特定的事件关闭当前文件并打开下一个新的文件，以达到存储的目的。

使用HDFS接收器的前提要求配置sink，将接收器类型设置为hdfs。

然后，还要指定路径，也就是配置path，指定数据写入的位置。路径有三种选择，分别是相对路径、绝对路径、带有Server名称的绝对路径。

最后一步是配置Channel信息。

配置HDFS:

<?xml version="1.0" encoding="UTF-8"?>

<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

```
<!--
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at
http://www.apache.org/licenses/LICENSE-2.0
Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file.
-->
<!-- Put site-specific property overrides in this file. -->
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
</configuration>
```

配置MapReduce

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at
http://www.apache.org/licenses/LICENSE-2.0
Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file.
-->
<!-- Put site-specific property overrides in this file. -->
<configuration>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
</configuration>
```

MapReduce的词频统计测试程序

```
import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
public class WordCount {
```

```
public static class TokenizerMapper
extends Mapper<Object, Text, Text, IntWritable>{
private final static IntWritable one = new IntWritable(1);
private Text word = new Text();
public void map(Object key, Text value, Context context
) throws IOException, InterruptedException {
StringTokenizer itr = new StringTokenizer(value.toString());
while (itr.hasMoreTokens()) {
word.set(itr.nextToken());
context.write(word, one);
}
}
}
public static class IntSumReducer
extends Reducer<Text,IntWritable,Text,IntWritable> {
private IntWritable result = new IntWritable();
public void reduce(Text key, Iterable<IntWritable> values,
Context context
) throws IOException, InterruptedException {
int sum = 0;
for (IntWritable val : values) {
sum += val.get();
}
result.set(sum);
context.write(key, result);
}
}
public static void main(String[] args) throws Exception {
Configuration conf = new Configuration();
String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();
if (otherArgs.length != 2) {
System.err.println("Usage: wordcount <in> <out>");
System.exit(2);
}
Job job = new Job(conf,"word count");
job.setJarByClass(WordCount.class);
job.setMapperClass(TokenizerMapper.class);
job.setCombinerClass(IntSumReducer.class);
job.setReducerClass(IntSumReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);
FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));
System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}
```

指　标
疑似剽窃文字表述

1. 环境搭建

   ·操作系统版本：ubuntu

·Hadoop版本：2.7.5

·Jdk版本：jdk1.8.0

·安装flume版本：FL

2. 当机器A上的进程调用机器B上的进程时，A上的调用进程被暂停，并且在B上执行调用进程。调用方

8. 53140204_关鸿睿_计算机科学与技术_基于Apache Flume的数据收集与分析系统的设计与实现_第8部分

总字数：16317

相似文献列表　文字复制比：76.7%(12521)　疑似剽窃观点：(0)

1	hadoop之MapReduce调用R的一次失败的总结~(续五) - yibei8811的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	46.1%（7526） 是否引证：否
2	hadoop入门系列之三【hadoop的安装与配置】 - u014548782的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	44.1%（7194） 是否引证：否
3	Hadoop-2.6.0集群HA搭建 - 鲍礼彬的CSDN博客 ~ -《网络（http://blog.csdn.net）》- 2017	42.7%（6963） 是否引证：否
4	求助hadoop2.X分布式搭建两个NameNode均无法正常启动 - 大数据之美 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	42.7%（6960） 是否引证：否
5	伪分布模式下HBase的安装 - wuruiaoxue的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	40.4%（6596） 是否引证：否
6	hadoop64位编译及开发环境搭建（二） - qyl445的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	39.3%（6416） 是否引证：否
7	Hadoop-2.5.1集群安装配置笔记 - greensurfer的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	38.6%（6297） 是否引证：否
8	hadoop namenode -format Couldn'tload main class "-Djava.library.path=.home.hadoop.hadoop-2.5.2.lib" - hzdxw的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	38.2%（6233） 是否引证：否
9	hadoop - php -《网络（http://blog.csdn.net）》- 2017	37.7%（6149） 是否引证：否
10	Hadoop2.3.0详细安装过程 - MySQL - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	37.4%（6101） 是否引证：否
11	hadoop 2.2.0安装开发环境（单机伪分布模式） - 我勒个去啊 -《网络（http://blog.csdn.net）》- 2017	36.3%（5927） 是否引证：否
12	ubuntu12.04+hadoop2.2.0+zookeeper3.4.5+hbase0.96.2+hive0.13.1伪分布式环境部署 - 迦壹的博墅 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	33.7%（5497） 是否引证：否
13	Centos6.5 x64 安装hadoop2.2.0 - 千里足迹 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	31.1%（5082） 是否引证：否
14	hadoop大数据平台手动搭建(八)HDFS High Availability Using the Quorum Journal Manager - feilong2483的博客 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	21.3%（3475） 是否引证：否
15	???? ??? :: -《网络（http://proits.tistor）》- 2015	20.9%（3406） 是否引证：否
16	hadoop jps 进程显示不全 - hzdxw的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	20.7%（3385） 是否引证：否
17	Hadoop系列二：启动HDFS和YARN过程日志 - 魔方泥瓦匠 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	20.3%（3306） 是否引证：否
18	hadoop dfsadmin -report failed - hzdxw的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	20.1%（3281） 是否引证：否
19	hadoop+zookeeper+hbase+hive+mahout整合配置 - leekwen的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	17.5%（2858） 是否引证：否
20	hadoop2.2 搭建QJM模式HA遇阻，求助贴 - 雲竹小师傅的破庙 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	6.4%（1037） 是否引证：否
21	Hadoop2.2.0 HA高可用分布式集群搭建(hbase,hive,sqoop,spark) - edent-ON THE WAY - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	4.3%（704） 是否引证：否
22	hive.ql.exec.DDLTask. MetaException(message:java.io.IOException: Attempt to start meta tracker	3.7%（606）

	faile - 风行天下 - CSDN博客	
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
23	Apache Hadoop安装 - 魔方泥瓦匠 - CSDN博客	0.8%（123）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
24	Hadoop学习---- Mac OSX下Hadoop 2.3.0安装及配置 - 博客频道 - CSDN.NET	0.4%（62）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
25	hadoop2.2.0编译与安装 - Spark MLlib 机器学习 - CSDN博客	0.4%（62）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

第五章系统测试

5.1 测试内容

集成几个典型的数据展示和分析的应用。

构建压力测试模型；测试系统的性能，在3-4台主流配置的服务器上，实现100000/S的日志请求。典型的日志数据大小1K左右。

5.2 系统各部分测试

5.2.1 Flume客户端测试

对程序采取了黑盒测试与白盒测试

5.2.2 Flume服务器测试

收集DataNode的工作日志

STARTUP_MSG: Starting DataNode
STARTUP_MSG: host = duquan-VirtualBox/127.0.1.1
STARTUP_MSG: args = []
STARTUP_MSG: version = 2.7.5
STARTUP_MSG: classpath = /opt/hadoop/etc/hadoop:/opt/hadoop/share/hadoop/common/lib/mockito-all-1.8.5.jar:/opt/hadoop/share/hadoop/common/lib/hamcrest-core-1.3.jar:/opt/hadoop/share/hadoop/common/lib/commons-math3-3.1.1.jar:/opt/hadoop/share/hadoop/common/lib/commons-net-3.1.jar:/opt/hadoop/share/hadoop/common/lib/xz-1.0.jar:/opt/hadoop/share/hadoop/common/lib/commons-beanutils-core-1.8.0.jar:/opt/hadoop/share/hadoop/common/lib/httpclient-4.2.5.jar:/opt/hadoop/share/hadoop/common/lib/apacheds-i18n-2.0.0-M15.jar:/opt/hadoop/share/hadoop/common/lib/commons-io-2.4.jar:/opt/hadoop/share/hadoop/common/lib/commons-configuration-1.6.jar:/opt/hadoop/share/hadoop/common/lib/api-util-1.0.0-M20.jar:/opt/hadoop/share/hadoop/common/lib/curator-client-2.7.1.jar:/opt/hadoop/share/hadoop/common/lib/protobuf-java-2.5.0.jar:/opt/hadoop/share/hadoop/common/lib/jetty-sslengine-6.1.26.jar:/opt/hadoop/share/hadoop/common/lib/jetty-6.1.26.jar:/opt/hadoop/share/hadoop/common/lib/jersey-core-1.9.jar:/opt/hadoop/share/hadoop/common/lib/jaxb-impl-2.2.3-1.jar:/opt/hadoop/share/hadoop/common/lib/gson-2.2.4.jar:/opt/hadoop/share/hadoop/common/lib/servlet-api-2.5.jar:/opt/hadoop/share/hadoop/common/lib/hadoop-auth-2.7.5.jar:/opt/hadoop/share/hadoop/common/lib/netty-3.6.2.Final.jar:/opt/hadoop/share/hadoop/common/lib/curator-framework-2.7.1.jar:/opt/hadoop/share/hadoop/common/lib/jackson-core-asl-1.9.13.jar:/opt/hadoop/share/hadoop/common/lib/jetty-util-6.1.26.jar:/opt/hadoop/share/hadoop/common/lib/commons-lang-2.6.jar:/opt/hadoop/share/hadoop/common/lib/junit-4.11.jar:/opt/hadoop/share/hadoop/common/lib/paranamer-2.3.jar:/opt/hadoop/share/hadoop/common/lib/slf4j-api-1.7.10.jar:/opt/hadoop/share/hadoop/common/lib/xmlenc-0.52.jar:/opt/hadoop/share/hadoop/common/lib/asm-3.2.jar:/opt/hadoop/share/hadoop/common/lib/jackson-xc-1.9.13.jar:/opt/hadoop/share/hadoop/common/lib/htrace-core-3.1.0-incubating.jar:/opt/hadoop/share/hadoop/common/lib/java-xmlbuilder-0.4.jar:/opt/hadoop/share/hadoop/common/lib/activation-1.1.jar:/opt/hadoop/share/hadoop/common/lib/jets3t-0.9.0.jar:/opt/hadoop/share/hadoop/common/lib/commons-httpclient-3.1.jar:/opt/hadoop/share/hadoop/common/lib/hadoop-annotations-2.7.5.jar:/opt/hadoop/share/hadoop/common/lib/commons-beanutils-1.7.0.jar:/opt/hadoop/share/hadoop/common/lib/commons-cli-1.2.jar:/opt/hadoop/share/hadoop/common/lib/guava-11.0.2.jar:/opt/hadoop/share/hadoop/common/lib/apacheds-kerberos-codec-2.0.0-M15.jar:/opt/hadoop/share/hadoop/common/lib/zookeeper-3.4.6.jar:/opt/hadoop/share/hadoop/common/lib/jackson-jaxrs-1.9.13.jar:/opt/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar:/opt/hadoop/share/hadoop/common/lib/jettison-1.1.jar:/opt/hadoop/share/hadoop/common/lib/avro-1.7.4.jar:/opt/hadoop/share/hadoop/common/lib/snappy-java-1.0.4.1.jar:/opt/hadoop/share/hadoop/common/lib/jackson-mapper-asl-1.9.13.jar:/opt/hadoop/share/hadoop/common/lib/commons-codec-1.4.jar:/opt/hadoop/share/hadoop/common/lib/commons-collections-3.2.2.jar:/opt/hadoop/share/hadoop/common/lib/jersey-json-

1.9.jar:/opt/hadoop/share/hadoop/common/lib/curator-recipes-2.7.1.jar:/opt/hadoop/share/hadoop/common/lib/httpcore-4.2.5.jar:/opt/hadoop/share/hadoop/common/lib/log4j-1.2.17.jar:/opt/hadoop/share/hadoop/common/lib/commons-logging-1.1.3.jar:/opt/hadoop/share/hadoop/common/lib/jaxb-api-2.2.2.jar:/opt/hadoop/share/hadoop/common/lib/jersey-server-1.9.jar:/opt/hadoop/share/hadoop/common/lib/jsp-api-2.1.jar:/opt/hadoop/share/hadoop/common/lib/commons-digester-1.8.jar:/opt/hadoop/share/hadoop/common/lib/jsch-0.1.54.jar:/opt/hadoop/share/hadoop/common/lib/commons-compress-1.4.1.jar:/opt/hadoop/share/hadoop/common/lib/stax-api-1.0-2.jar:/opt/hadoop/share/hadoop/common/lib/api-asn1-api-1.0.0-M20.jar:/opt/hadoop/share/hadoop/common/lib/jsr305-3.0.0.jar:/opt/hadoop/share/hadoop/common/hadoop-nfs-2.7.5.jar:/opt/hadoop/share/hadoop/common/hadoop-common-2.7.5.jar:/opt/hadoop/share/hadoop/common/hadoop-common-2.7.5-tests.jar:/opt/hadoop/share/hadoop/hdfs:/opt/hadoop/share/hadoop/hdfs/lib/netty-all-4.0.23.Final.jar:/opt/hadoop/share/hadoop/hdfs/lib/commons-io-2.4.jar:/opt/hadoop/share/hadoop/hdfs/lib/protobuf-java-2.5.0.jar:/opt/hadoop/share/hadoop/hdfs/lib/jetty-6.1.26.jar:/opt/hadoop/share/hadoop/hdfs/lib/jersey-core-1.9.jar:/opt/hadoop/share/hadoop/hdfs/lib/servlet-api-2.5.jar:/opt/hadoop/share/hadoop/hdfs/lib/netty-3.6.2.Final.jar:/opt/hadoop/share/hadoop/hdfs/lib/jackson-core-asl-1.9.13.jar:/opt/hadoop/share/hadoop/hdfs/lib/jetty-util-6.1.26.jar:/opt/hadoop/share/hadoop/hdfs/lib/commons-lang-2.6.jar:/opt/hadoop/share/hadoop/hdfs/lib/xmlenc-0.52.jar:/opt/hadoop/share/hadoop/hdfs/lib/asm-3.2.jar:/opt/hadoop/share/hadoop/hdfs/lib/htrace-core-3.1.0-incubating.jar:/opt/hadoop/share/hadoop/hdfs/lib/commons-cli-1.2.jar:/opt/hadoop/share/hadoop/hdfs/lib/guava-11.0.2.jar:/opt/hadoop/share/hadoop/hdfs/lib/leveldbjni-all-1.8.jar:/opt/hadoop/share/hadoop/hdfs/lib/jackson-mapper-asl-1.9.13.jar:/opt/hadoop/share/hadoop/hdfs/lib/commons-codec-1.4.jar:/opt/hadoop/share/hadoop/hdfs/lib/xercesImpl-2.9.1.jar:/opt/hadoop/share/hadoop/hdfs/lib/commons-daemon-1.0.13.jar:/opt/hadoop/share/hadoop/hdfs/lib/log4j-1.2.17.jar:/opt/hadoop/share/hadoop/hdfs/lib/commons-logging-1.1.3.jar:/opt/hadoop/share/hadoop/hdfs/lib/jersey-server-1.9.jar:/opt/hadoop/share/hadoop/hdfs/lib/xml-apis-1.3.04.jar:/opt/hadoop/share/hadoop/hdfs/lib/jsr305-3.0.0.jar:/opt/hadoop/share/hadoop/hdfs/hadoop-hdfs-nfs-2.7.5.jar:/opt/hadoop/share/hadoop/hdfs/hadoop-hdfs-2.7.5-tests.jar:/opt/hadoop/share/hadoop/hdfs/hadoop-hdfs-2.7.5.jar:/opt/hadoop/share/hadoop/yarn/lib/xz-1.0.jar:/opt/hadoop/share/hadoop/yarn/lib/commons-io-2.4.jar:/opt/hadoop/share/hadoop/yarn/lib/jersey-client-1.9.jar:/opt/hadoop/share/hadoop/yarn/lib/protobuf-java-2.5.0.jar:/opt/hadoop/share/hadoop/yarn/lib/zookeeper-3.4.6-tests.jar:/opt/hadoop/share/hadoop/yarn/lib/jetty-6.1.26.jar:/opt/hadoop/share/hadoop/yarn/lib/jersey-core-1.9.jar:/opt/hadoop/share/hadoop/yarn/lib/jaxb-impl-2.2.3-1.jar:/opt/hadoop/share/hadoop/yarn/lib/servlet-api-2.5.jar:/opt/hadoop/share/hadoop/yarn/lib/netty-3.6.2.Final.jar:/opt/hadoop/share/hadoop/yarn/lib/guice-3.0.jar:/opt/hadoop/share/hadoop/yarn/lib/jackson-core-asl-1.9.13.jar:/opt/hadoop/share/hadoop/yarn/lib/jetty-util-6.1.26.jar:/opt/hadoop/share/hadoop/yarn/lib/commons-lang-2.6.jar:/opt/hadoop/share/hadoop/yarn/lib/asm-3.2.jar:/opt/hadoop/share/hadoop/yarn/lib/guice-servlet-3.0.jar:/opt/hadoop/share/hadoop/yarn/lib/jackson-xc-1.9.13.jar:/opt/hadoop/share/hadoop/yarn/lib/jersey-guice-1.9.jar:/opt/hadoop/share/hadoop/yarn/lib/activation-1.1.jar:/opt/hadoop/share/hadoop/yarn/lib/javax.inject-1.jar:/opt/hadoop/share/hadoop/yarn/lib/commons-cli-1.2.jar:/opt/hadoop/share/hadoop/yarn/lib/guava-11.0.2.jar:/opt/hadoop/share/hadoop/yarn/lib/zookeeper-3.4.6.jar:/opt/hadoop/share/hadoop/yarn/lib/jackson-jaxrs-1.9.13.jar:/opt/hadoop/share/hadoop/yarn/lib/leveldbjni-all-1.8.jar:/opt/hadoop/share/hadoop/yarn/lib/jettison-1.1.jar:/opt/hadoop/share/hadoop/yarn/lib/jackson-mapper-asl-1.9.13.jar:/opt/hadoop/share/hadoop/yarn/lib/commons-codec-1.4.jar:/opt/hadoop/share/hadoop/yarn/lib/commons-collections-3.2.2.jar:/opt/hadoop/share/hadoop/yarn/lib/jersey-json-1.9.jar:/opt/hadoop/share/hadoop/yarn/lib/log4j-1.2.17.jar:/opt/hadoop/share/hadoop/yarn/lib/commons-logging-1.1.3.jar:/opt/hadoop/share/hadoop/yarn/lib/jaxb-api-2.2.2.jar:/opt/hadoop/share/hadoop/yarn/lib/jersey-server-1.9.jar:/opt/hadoop/share/hadoop/yarn/lib/commons-compress-1.4.1.jar:/opt/hadoop/share/hadoop/yarn/lib/stax-api-1.0-2.jar:/opt/hadoop/share/hadoop/yarn/lib/aopalliance-1.0.jar:/opt/hadoop/share/hadoop/yarn/lib/jsr305-3.0.0.jar:/opt/hadoop/share/hadoop/yarn/hadoop-yarn-client-2.7.5.jar:/opt/hadoop/share/hadoop/yarn/hadoop-yarn-registry-2.7.5.jar:/opt/hadoop/share/hadoop/yarn/hadoop-yarn-server-common-2.7.5.jar:/opt/hadoop/share/hadoop/yarn/hadoop-yarn-server-nodemanager-2.7.5.jar:/opt/hadoop/share/hadoop/yarn/hadoop-yarn-applications-distributedshell-2.7.5.jar:/opt/hadoop/share/hadoop/yarn/hadoop-yarn-server-sharedcachemanager-2.7.5.jar:/opt/hadoop/share/hadoop/yarn/hadoop-yarn-server-applicationhistoryservice-2.7.5.jar:/opt/hadoop/share/hadoop/yarn/hadoop-yarn-applications-unmanaged-am-launcher-2.7.5.jar:/opt/hadoop/share/hadoop/yarn/hadoop-yarn-api-2.7.5.jar:/opt/hadoop/share/hadoop/yarn/hadoop-yarn-common-2.7.5.jar:/opt/hadoop/share/hadoop/yarn/hadoop-yarn-server-resourcemanager-2.7.5.jar:/opt/hadoop/share/hadoop/yarn/hadoop-yarn-server-tests-2.7.5.jar:/opt/hadoop/share/hadoop/yarn/hadoop-yarn-server-web-proxy-2.7.5.jar:/opt/hadoop/share/hadoop/mapreduce/lib/hamcrest-core-1.3.jar:/opt/hadoop/share/hadoop/mapreduce/lib/xz-1.0.jar:/opt/hadoop/share/hadoop/mapreduce/lib/commons-io-

2.4.jar:/opt/hadoop/share/hadoop/mapreduce/lib/protobuf-java-2.5.0.jar:/opt/hadoop/share/hadoop/mapreduce/lib/jersey-core-1.9.jar:/opt/hadoop/share/hadoop/mapreduce/lib/netty-3.6.2.Final.jar:/opt/hadoop/share/hadoop/mapreduce/lib/guice-3.0.jar:/opt/hadoop/share/hadoop/mapreduce/lib/jackson-core-asl-1.9.13.jar:/opt/hadoop/share/hadoop/mapreduce/lib/junit-4.11.jar:/opt/hadoop/share/hadoop/mapreduce/lib/paranamer-2.3.jar:/opt/hadoop/share/hadoop/mapreduce/lib/asm-3.2.jar:/opt/hadoop/share/hadoop/mapreduce/lib/guice-servlet-3.0.jar:/opt/hadoop/share/hadoop/mapreduce/lib/jersey-guice-1.9.jar:/opt/hadoop/share/hadoop/mapreduce/lib/javax.inject-1.jar:/opt/hadoop/share/hadoop/mapreduce/lib/hadoop-annotations-2.7.5.jar:/opt/hadoop/share/hadoop/mapreduce/lib/leveldbjni-all-1.8.jar:/opt/hadoop/share/hadoop/mapreduce/lib/avro-1.7.4.jar:/opt/hadoop/share/hadoop/mapreduce/lib/snappy-java-1.0.4.1.jar:/opt/hadoop/share/hadoop/mapreduce/lib/jackson-mapper-asl-1.9.13.jar:/opt/hadoop/share/hadoop/mapreduce/lib/log4j-1.2.17.jar:/opt/hadoop/share/hadoop/mapreduce/lib/jersey-server-1.9.jar:/opt/hadoop/share/hadoop/mapreduce/lib/commons-compress-1.4.1.jar:/opt/hadoop/share/hadoop/mapreduce/lib/aopalliance-1.0.jar:/opt/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-client-app-2.7.5.jar:/opt/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.5.jar:/opt/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-client-jobclient-2.7.5.jar:/opt/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-client-hs-plugins-2.7.5.jar:/opt/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-client-core-2.7.5.jar:/opt/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-client-common-2.7.5.jar:/opt/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-client-shuffle-2.7.5.jar:/opt/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-client-jobclient-2.7.5-tests.jar:/opt/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-client-hs-2.7.5.jar:/contrib/capacity-scheduler/*.jar:/contrib/capacity-scheduler/*.jar:/contrib/capacity-scheduler/*.jar

STARTUP_MSG: build = https://shv@git-wip-us.apache.org/repos/asf/hadoop.git -r 18065c2b6806ed4aa6a3187d77cbe21bb3dba075; compiled by 'kshvachk' on 2017-12-16T01:06Z

STARTUP_MSG: java = 1.8.0_151

**/

2018-05-16 20:40:48,367 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: registered UNIX signal handlers for [TERM, HUP, INT]

2018-05-16 20:40:54,555 INFO org.apache.hadoop.metrics2.impl.MetricsConfig: loaded properties from hadoop-metrics2.properties

2018-05-16 20:40:55,289 INFO org.apache.hadoop.metrics2.impl.MetricsSystemImpl: Scheduled snapshot period at 10 second(s).

2018-05-16 20:40:55,317 INFO org.apache.hadoop.metrics2.impl.MetricsSystemImpl: DataNode metrics system started

2018-05-16 20:40:55,367 INFO org.apache.hadoop.hdfs.server.datanode.BlockScanner: Initialized block scanner with targetBytesPerSec 1048576

2018-05-16 20:40:55,412 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Configured hostname is duquan-VirtualBox

2018-05-16 20:40:55,507 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Starting DataNode with maxLockedMemory = 0

2018-05-16 20:40:55,742 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Opened streaming server at /0.0.0.0:50010

2018-05-16 20:40:55,782 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Balancing bandwith is 1048576 bytes/s

2018-05-16 20:40:55,782 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Number threads for balancing is 5

2018-05-16 20:40:56,655 INFO org.mortbay.log: Logging to org.slf4j.impl.Log4jLoggerAdapter(org.mortbay.log) via org.mortbay.log.Slf4jLog

2018-05-16 20:40:56,712 INFO org.apache.hadoop.security.authentication.server.AuthenticationFilter: Unable to initialize FileSignerSecretProvider, falling back to use random secrets.

2018-05-16 20:40:56,838 INFO org.apache.hadoop.http.HttpRequestLog: Http request log for http.requests.datanode is not defined

2018-05-16 20:40:56,916 INFO org.apache.hadoop.http.HttpServer2: Added global filter 'safety' (class=org.apache.hadoop.http.HttpServer2$QuotingInputFilter)

2018-05-16 20:40:56,938 INFO org.apache.hadoop.http.HttpServer2: Added filter static_user_filter

(class=org.apache.hadoop.http.lib.StaticUserWebFilter$StaticUserFilter) to context datanode

2018-05-16 20:40:56,938 INFO org.apache.hadoop.http.HttpServer2: Added filter static_user_filter (class=org.apache.hadoop.http.lib.StaticUserWebFilter$StaticUserFilter) to context static

2018-05-16 20:40:56,938 INFO org.apache.hadoop.http.HttpServer2: Added filter static_user_filter (class=org.apache.hadoop.http.lib.StaticUserWebFilter$StaticUserFilter) to context logs

2018-05-16 20:40:57,077 INFO org.apache.hadoop.http.HttpServer2: Jetty bound to port 41629

2018-05-16 20:40:57,077 INFO org.mortbay.log: jetty-6.1.26

2018-05-16 20:40:58,548 INFO org.mortbay.log: Started HttpServer2$SelectChannelConnectorWithSafeStartup@localhost:41629

2018-05-16 20:40:59,839 INFO org.apache.hadoop.hdfs.server.datanode.web.DatanodeHttpServer: Listening HTTP traffic on /0.0.0.0:50075

2018-05-16 20:41:04,032 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: dnUserName = root

2018-05-16 20:41:04,032 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: supergroup = supergroup

2018-05-16 20:41:04,738 INFO org.apache.hadoop.ipc.CallQueueManager: Using callQueue: class java.util.concurrent.LinkedBlockingQueue queueCapacity: 1000

2018-05-16 20:41:04,891 INFO org.apache.hadoop.ipc.Server: Starting Socket Reader #1 for port 50020

2018-05-16 20:41:06,051 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Opened IPC server at /0.0.0.0:50020

2018-05-16 20:41:06,306 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Refresh request received for nameservices: null

2018-05-16 20:41:06,572 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Starting BPOfferServices for nameservices: <default>

2018-05-16 20:41:06,756 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Block pool <registering> (Datanode Uuid unassigned) service to singlenode/127.0.0.1:9000 starting to offer service

2018-05-16 20:41:06,841 INFO org.apache.hadoop.ipc.Server: IPC Server Responder: starting

2018-05-16 20:41:06,859 INFO org.apache.hadoop.ipc.Server: IPC Server listener on 50020: starting

2018-05-16 20:41:09,321 INFO org.apache.hadoop.hdfs.server.common.Storage: Using 1 threads to upgrade data directories (dfs.datanode.parallel.volumes.load.threads.num=1, dataDirs=1)

2018-05-16 20:41:09,375 INFO org.apache.hadoop.hdfs.server.common.Storage: Lock on /opt/hadoop/tmp/dfs/data/in_use.lock acquired by nodename 2729@duquan-VirtualBox

2018-05-16 20:41:09,747 INFO org.apache.hadoop.hdfs.server.common.Storage: Analyzing storage directories for bpid BP-448324278-127.0.1.1-1516004778052

2018-05-16 20:41:09,747 INFO org.apache.hadoop.hdfs.server.common.Storage: Locking is disabled for /opt/hadoop/tmp/dfs/data/current/BP-448324278-127.0.1.1-1516004778052

2018-05-16 20:41:09,836 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Setting up storage: nsid=2110769945;bpid=BP-448324278-127.0.1.1-1516004778052;lv=-56;nsInfo=lv=-63;cid=CID-955cbe9a-d625-459a-9d5c-ddf40ff96570;nsid=2110769945;c=0;bpid=BP-448324278-127.0.1.1-1516004778052;dnuuid=08342554-a11c-4138-8689-cee4c0e0617e

2018-05-16 20:41:10,

9. 53140204_关鸿睿_计算机科学与技术_基于Apache Flume的数据收集与分析系统的设计与实现_第9部分

总字数：17234

相似文献列表　文字复制比：50.7%(8731)　疑似剽窃观点：(0)

1	Hadoop系列二：启动HDFS和YARN过程日志 - 魔方泥瓦匠 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	16.1%（2781） 是否引证：否
2	Hadoop系列三：运行job过程日志 - 魔方泥瓦匠 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	10.8%（1856） 是否引证：否
3	关于DataNode更改IP地址后所可能引发HDFS集群状态变化的分析 - Scott的专栏 - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》- 2013	10.3%（1782） 是否引证：否
4	关于DataNode更改IP地址后所可能引发HDFS集群状态变化的分析 - Scott的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	10.3%（1782） 是否引证：否
5	求助hadoop2.X分布式搭建两个NameNode均无法正常启动 - 大数据之美 - CSDN博客	9.9%（1698）

	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
6	[Hive]那些年我们踩过的Hive坑 - Ying - CSDN博客	8.8%（1520）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
7	关于使用hadoop出各种错的一些积累 - 筑墙的博客 - CSDN博客	7.3%（1265）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
8	Hadoop系列四： HDFS相关的checkpoint交互 - 魔方泥瓦匠 - CSDN博客	7.2%（1246）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
9	Hadoop安装配置(伪分布模式) - mxfeng的专栏 - CSDN博客	7.1%（1222）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
10	hadoop基本知识 - xupeng874395012的博客 - CSDN博客	7.0%（1206）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
11	Spark学习笔记@第一个例子wordcount+Eclipse - 骑猪喝咖啡 - CSDN博客	6.7%（1148）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
12	CDH3b4_Installation	5.2%（896）
	- 《互联网文档资源（http://www.docin.com）》- 2012	是否引证：否
13	Hadoop基础教程-第9章 HA高可用（9.3 HDFS 高可用运行）（草稿） - 程裕强的专栏 - CSDN博客	5.0%（867）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
14	运行hadoop 出现的一些问题 及其解决方法	4.7%（816）
	- 《互联网文档资源（http://www.360doc.co）》- 2015	是否引证：否
15	运行hadoop 出现的一些问题 及其解决方法	4.7%（816）
	- 《互联网文档资源（http://www.360doc.co）》- 2017	是否引证：否
16	hadoop HA集群搭建(red hat) - shursulei的博客 - CSDN博客	4.2%（723）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
17	Ubuntu14.04下hadoop-2.6.0单机配置和伪分布式配置 - Reverse - CSDN博客	4.0%（690）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
18	Hadoop2.x实战：Eclipse本地开发环境搭建与本地运行wordcount实例 - 林炳文Evankaka的专栏 - CSDN博客	4.0%（690）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
19	hadoop配置常见问题 - Stereo - CSDN博客	3.4%（584）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
20	【转】hadoop集群搭建时的奇怪现象 Name node is in safe mode 解决方法【转】 _syn_001	3.1%（540）
	- 《网络（http://blog.sina.com）》- 2015	是否引证：否
21	HDFS常用的Java Api详解 - 码农崛起 - CSDN博客	1.9%（335）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
22	Hadoop配置机架感知（python 编写） - gaobooo的专栏 - CSDN博客	1.7%（286）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
23	hadoop安装配置 - i5secs的专栏 - CSDN博客	0.8%（137）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
24	Name node is in safe mode - Aidon博客 - CSDN博客	0.6%（108）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

083 INFO org.apache.hadoop.hdfs.server.datanode.fsdataset.impl.FsDatasetImpl: Added new volume: DS-27a6561f-f569-461c-816e-de1bae032bb8

2018-05-16 20:41:10,083 INFO org.apache.hadoop.hdfs.server.datanode.fsdataset.impl.FsDatasetImpl: Added volume - /opt/hadoop/tmp/dfs/data/current, StorageType: DISK

2018-05-16 20:41:10,400 INFO org.apache.hadoop.hdfs.server.datanode.fsdataset.impl.FsDatasetImpl: Registered FSDatasetState MBean

2018-05-16 20:41:10,427 INFO org.apache.hadoop.hdfs.server.datanode.fsdataset.impl.FsDatasetImpl: Adding block pool BP-448324278-127.0.1.1-1516004778052

2018-05-16 20:41:10,427 INFO org.apache.hadoop.hdfs.server.datanode.fsdataset.impl.FsDatasetImpl: Scanning block pool BP-448324278-127.0.1.1-1516004778052 on volume /opt/hadoop/tmp/dfs/data/current...

2018-05-16 20:41:10,619 INFO org.apache.hadoop.hdfs.server.datanode.fsdataset.impl.FsDatasetImpl: Time taken to

scan block pool BP-448324278-127.0.1.1-1516004778052 on /opt/hadoop/tmp/dfs/data/current: 191ms

2018-05-16 20:41:10,619 INFO org.apache.hadoop.hdfs.server.datanode.fsdataset.impl.FsDatasetImpl: Total time to scan all replicas for block pool BP-448324278-127.0.1.1-1516004778052: 192ms

2018-05-16 20:41:10,627 INFO org.apache.hadoop.hdfs.server.datanode.fsdataset.impl.FsDatasetImpl: Adding replicas to map for block pool BP-448324278-127.0.1.1-1516004778052 on volume /opt/hadoop/tmp/dfs/data/current...

2018-05-16 20:41:10,668 INFO org.apache.hadoop.hdfs.server.datanode.fsdataset.impl.FsDatasetImpl: Time to add replicas to map for block pool BP-448324278-127.0.1.1-1516004778052 on volume /opt/hadoop/tmp/dfs/data/current: 41ms

2018-05-16 20:41:10,668 INFO org.apache.hadoop.hdfs.server.datanode.fsdataset.impl.FsDatasetImpl: Total time to add all replicas to map: 48ms

2018-05-16 20:41:11,100 INFO org.apache.hadoop.hdfs.server.datanode.VolumeScanner: Now rescanning bpid BP-448324278-127.0.1.1-1516004778052 on volume /opt/hadoop/tmp/dfs/data, after more than 504 hour(s)

2018-05-16 20:41:11,135 ERROR org.apache.hadoop.hdfs.server.datanode.DirectoryScanner: dfs.datanode.directoryscan.throttle.limit.ms.per.sec set to value below 1 ms/sec. Assuming default value of 1000

2018-05-16 20:41:11,136 INFO org.apache.hadoop.hdfs.server.datanode.DirectoryScanner: Periodic Directory Tree Verification scan starting at 1526490964136ms with interval of 21600000ms

2018-05-16 20:41:11,164 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Block pool BP-448324278-127.0.1.1-1516004778052 (Datanode Uuid null) service to singlenode/127.0.0.1:9000 beginning handshake with NN

2018-05-16 20:41:11,482 INFO org.apache.hadoop.hdfs.server.datanode.VolumeScanner: VolumeScanner(/opt/hadoop/tmp/dfs/data, DS-27a6561f-f569-461c-816e-de1bae032bb8): finished scanning block pool BP-448324278-127.0.1.1-1516004778052

2018-05-16 20:41:11,489 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Block pool Block pool BP-448324278-127.0.1.1-1516004778052 (Datanode Uuid null) service to singlenode/127.0.0.1:9000 successfully registered with NN

2018-05-16 20:41:11,489 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: For namenode singlenode/127.0.0.1:9000 using BLOCKREPORT_INTERVAL of 21600000msec CACHEREPORT_INTERVAL of 10000msec Initial delay: 0msec; heartBeatInterval=3000

2018-05-16 20:41:11,578 INFO org.apache.hadoop.hdfs.server.datanode.VolumeScanner: VolumeScanner(/opt/hadoop/tmp/dfs/data, DS-27a6561f-f569-461c-816e-de1bae032bb8): no suitable block pools found to scan. Waiting 1814399521 ms.

2018-05-16 20:41:11,958 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Namenode Block pool BP-448324278-127.0.1.1-1516004778052 (Datanode Uuid 08342554-a11c-4138-8689-cee4c0e0617e) service to singlenode/127.0.0.1:9000 trying to claim ACTIVE state with txid=387

2018-05-16 20:41:11,959 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Acknowledging ACTIVE Namenode Block pool BP-448324278-127.0.1.1-1516004778052 (Datanode Uuid 08342554-a11c-4138-8689-cee4c0e0617e) service to singlenode/127.0.0.1:9000

2018-05-16 20:41:12,388 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Successfully sent block report 0x36d5d45d1b, containing 1 storage report(s), of which we sent 1. The reports had 12 total blocks and used 1 RPC(s). This took 2 msec to generate and 426 msecs for RPC and NN processing. Got back one command: FinalizeCommand/5.

2018-05-16 20:41:12,388 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: Got finalize command for block pool BP-448324278-127.0.1.1-1516004778052

2018-05-16 21:05:50,505 WARN org.apache.hadoop.hdfs.server.datanode.DataNode: IOException in offerService

java.io.EOFException: End of File Exception between local host is: "duquan-VirtualBox/127.0.1.1"; destination host is: "singlenode":9000; : java.io.EOFException; For more details see: http://wiki.apache.org/hadoop/EOFException

at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)

at sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:62)

at sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:45)

at java.lang.reflect.Constructor.newInstance(Constructor.java:423)

at org.apache.hadoop.net.NetUtils.wrapWithMessage(NetUtils.java:792)

at org.apache.hadoop.net.NetUtils.wrapException(NetUtils.java:765)

at org.apache.hadoop.ipc.Client.call(Client.java:1480)

at org.apache.hadoop.ipc.Client.call(Client.java:1413)

at org.apache.hadoop.ipc.ProtobufRpcEngine$Invoker.invoke(ProtobufRpcEngine.java:229)

at com.sun.proxy.$Proxy15.sendHeartbeat(Unknown Source)

at org.apache.hadoop.hdfs.protocolPB.DatanodeProtocolClientSideTranslatorPB.sendHeartbeat(DatanodeProtocolClientSideTranslatorPB.java:152)

at org.apache.hadoop.hdfs.server.datanode.BPServiceActor.sendHeartBeat(BPServiceActor.java:402)

at org.apache.hadoop.hdfs.server.datanode.BPServiceActor.offerService(BPServiceActor.java:500)

at org.apache.hadoop.hdfs.server.datanode.BPServiceActor.run(BPServiceActor.java:659)

at java.lang.Thread.run(Thread.java:748)

Caused by: java.io.EOFException

at java.io.DataInputStream.readInt(DataInputStream.java:392)

at org.apache.hadoop.ipc.Client$Connection.receiveRpcResponse(Client.java:1085)

at org.apache.hadoop.ipc.Client$Connection.run(Client.java:980)

2018-05-16 21:05:54,537 INFO org.apache.hadoop.ipc.Client: Retrying connect to server: singlenode/127.0.0.1:9000. Already tried 0 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)

2018-05-16 21:05:54,585 ERROR org.apache.hadoop.hdfs.server.datanode.DataNode: RECEIVED SIGNAL 15: SIGTERM

2018-05-16 21:05:54,589 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: SHUTDOWN_MSG:

/**

SHUTDOWN_MSG: Shutting down DataNode at duquan-VirtualBox/127.0.1.1

**/

5.2.3 数据存储模块测试

MapReduce的工作日志

图5-1

图5-2

图5-3

词频统计结果如下：

图5-4

此外还收集了部分HDFS中NameNode的工作日志：

FileJournalManager: Finalizing edits file /opt/hadoop/tmp/dfs/name/current/edits_inprogress_0000000000000000386 -> /opt/hadoop/tmp/dfs/name/current/edits_0000000000000000386-0000000000000000386

2018-05-16 20:40:57,970 INFO org.apache.hadoop.hdfs.server.namenode.FSImage: Planning to load image: FSImageFile(file=/opt/hadoop/tmp/dfs/name/current/fsimage_0000000000000000385, cpktTxId=0000000000000000385)

2018-05-16 20:40:58,361 INFO org.apache.hadoop.hdfs.server.namenode.FSImageFormatPBINode: Loading 26 INodes.

2018-05-16 20:40:58,907 INFO org.apache.hadoop.hdfs.server.namenode.FSImageFormatProtobuf: Loaded FSImage in 0 seconds.

2018-05-16 20:40:58,907 INFO org.apache.hadoop.hdfs.server.namenode.FSImage: Loaded image for txid 385 from /opt/hadoop/tmp/dfs/name/current/fsimage_0000000000000000385

2018-05-16 20:40:58,907 INFO org.apache.hadoop.hdfs.server.namenode.FSImage: Reading org.apache.hadoop.hdfs.server.namenode.RedundantEditLogInputStream@2bef51f2 expecting start txid #386

2018-05-16 20:40:58,907 INFO org.apache.hadoop.hdfs.server.namenode.FSImage: Start loading edits file /opt/hadoop/tmp/dfs/name/current/edits_0000000000000000386-0000000000000000386

2018-05-16 20:40:58,909 INFO org.apache.hadoop.hdfs.server.namenode.EditLogInputStream: Fast-forwarding stream '/opt/hadoop/tmp/dfs/name/current/edits_0000000000000000386-0000000000000000386' to transaction ID 386

2018-05-16 20:40:58,940 INFO org.apache.hadoop.hdfs.server.namenode.FSImage: Edits file /opt/hadoop/tmp/dfs/name/current/edits_0000000000000000386-0000000000000000386 of size 1048576 edits # 1 loaded in 0 seconds

2018-05-16 20:40:58,941 INFO org.apache.hadoop.hdfs.server.namenode.FSImage: Initializing quota with 4 thread(s)

2018-05-16 20:40:59,205 INFO org.apache.hadoop.hdfs.server.namenode.FSImage: Quota initialization completed in 264 milliseconds

name space=26

storage space=463007

storage types=RAM_DISK=0, SSD=0, DISK=0, ARCHIVE=0

2018-05-16 20:40:59,205 INFO org.apache.hadoop.hdfs.server.namenode.FSNamesystem: Need to save fs image? true (staleImage=true, haEnabled=false, isRollingUpgrade=false)

2018-05-16 20:40:59,205 INFO org.apache.hadoop.hdfs.server.namenode.FSImage: Save namespace ...

2018-05-16 20:40:59,295 INFO org.apache.hadoop.hdfs.server.namenode.FSImageFormatProtobuf: Saving image file /opt/hadoop/tmp/dfs/name/current/fsimage.ckpt_0000000000000000386 using no compression

2018-05-16 20:40:59,737 INFO org.apache.hadoop.hdfs.server.namenode.FSImageFormatProtobuf: Image file /opt/hadoop/tmp/dfs/name/current/fsimage.ckpt_0000000000000000386 of size 2431 bytes saved in 0 seconds.

2018-05-16 20:40:59,811 INFO org.apache.hadoop.hdfs.server.namenode.NNStorageRetentionManager: Going to retain 2 images with txid >= 385

2018-05-16 20:40:59,811 INFO org.apache.hadoop.hdfs.server.namenode.NNStorageRetentionManager: Purging old image FSImageFile(file=/opt/hadoop/tmp/dfs/name/current/fsimage_0000000000000000383, cpktTxId=0000000000000000383)

2018-05-16 20:41:00,160 INFO org.apache.hadoop.hdfs.server.namenode.FSEditLog: Starting log segment at 387

2018-05-16 20:41:01,171 INFO org.apache.hadoop.hdfs.server.namenode.NameCache: initialized with 0 entries 0 lookups

2018-05-16 20:41:01,171 INFO org.apache.hadoop.hdfs.server.namenode.FSNamesystem: Finished loading FSImage in 4448 msecs

2018-05-16 20:41:04,336 INFO org.apache.hadoop.hdfs.server.namenode.NameNode: RPC server is binding to singlenode:9000

2018-05-16 20:41:04,509 INFO org.apache.hadoop.ipc.CallQueueManager: Using callQueue: class java.util.concurrent.LinkedBlockingQueue queueCapacity: 1000

2018-05-16 20:41:04,626 INFO org.apache.hadoop.ipc.Server: Starting Socket Reader #1 for port 9000

2018-05-16 20:41:05,407 INFO org.apache.hadoop.hdfs.server.namenode.FSNamesystem: Registered FSNamesystemState MBean

2018-05-16 20:41:05,613 INFO org.apache.hadoop.hdfs.server.namenode.LeaseManager: Number of blocks under construction: 0

2018-05-16 20:41:05,613 INFO org.apache.hadoop.hdfs.server.namenode.LeaseManager: Number of blocks under construction: 0

2018-05-16 20:41:05,613 INFO org.apache.hadoop.hdfs.StateChange: STATE* Safe mode ON.
The reported blocks 0 needs additional 12 blocks to reach the threshold 0.9990 of total blocks 12.
The number of live datanodes 0 has reached the minimum number 0. Safe mode will be turned off automatically once the thresholds have been reached.

2018-05-16 20:41:05,722 INFO org.apache.hadoop.hdfs.server.blockmanagement.DatanodeDescriptor: Number of failed storage changes from 0 to 0

2018-05-16 20:41:06,343 INFO org.apache.hadoop.ipc.Server: IPC Server listener on 9000: starting

2018-05-16 20:41:06,342 INFO org.apache.hadoop.ipc.Server: IPC Server Responder: starting

2018-05-16 20:41:06,397 INFO org.apache.hadoop.hdfs.server.namenode.NameNode: NameNode RPC up at: singlenode/127.0.0.1:9000

2018-05-16 20:41:06,397 INFO org.apache.hadoop.hdfs.server.namenode.FSNamesystem: Starting services required for active state

2018-05-16 20:41:06,565 INFO org.apache.hadoop.hdfs.server.blockmanagement.CacheReplicationMonitor: Starting CacheReplicationMonitor with interval 30000 milliseconds

2018-05-16 20:41:11,393 INFO org.apache.hadoop.hdfs.StateChange: BLOCK* registerDatanode: from DatanodeRegistration(127.0.0.1:50010, datanodeUuid=08342554-a11c-4138-8689-cee4c0e0617e, infoPort=50075, infoSecurePort=0, ipcPort=50020, storageInfo=lv=-56;cid=CID-955cbe9a-d625-459a-9d5c-ddf40ff96570;nsid=2110769945;c=0) storage 08342554-a11c-4138-8689-cee4c0e0617e

2018-05-16 20:41:11,393 INFO org.apache.hadoop.hdfs.server.blockmanagement.DatanodeDescriptor: Number of failed storage changes from 0 to 0

2018-05-16 20:41:11,407 INFO org.apache.hadoop.net.NetworkTopology: Adding a new node: /default-rack/127.0.0.1:50010

2018-05-16 20:41:11,829 INFO org.apache.hadoop.hdfs.server.blockmanagement.DatanodeDescriptor: Number of failed storage changes from 0 to 0

2018-05-16 20:41:11,829 INFO org.apache.hadoop.hdfs.server.blockmanagement.DatanodeDescriptor: Adding new

storage ID DS-27a6561f-f569-461c-816e-de1bae032bb8 for DN 127.0.0.1:50010

2018-05-16 20:41:12,228 INFO org.apache.hadoop.hdfs.StateChange: STATE* Safe mode extension entered.

The reported blocks 11 has reached the threshold 0.9990 of total blocks 12. The number of live datanodes 1 has reached the minimum number 0. In safe mode extension. Safe mode will be turned off automatically in 29 seconds.

2018-05-16 20:41:12,229 INFO org.apache.hadoop.hdfs.server.namenode.FSNamesystem: initializing replication queues

2018-05-16 20:41:12,262 INFO BlockStateChange: BLOCK* processReport 0x36d5d45d1b: from storage DS-27a6561f-f569-461c-816e-de1bae032bb8 node DatanodeRegistration(127.0.0.1:50010, datanodeUuid=08342554-a11c-4138-8689-cee4c0e0617e, infoPort=50075, infoSecurePort=0, ipcPort=50020, storageInfo=lv=-56;cid=CID-955cbe9a-d625-459a-9d5c-ddf40ff96570;nsid=2110769945;c=0), blocks: 12, hasStaleStorage: false, processing time: 90 msecs

2018-05-16 20:41:12,279 INFO org.apache.hadoop.hdfs.server.blockmanagement.BlockManager: Total number of blocks = 12

2018-05-16 20:41:12,279 INFO org.apache.hadoop.hdfs.server.blockmanagement.BlockManager: Number of invalid blocks = 0

2018-05-16 20:41:12,279 INFO org.apache.hadoop.hdfs.server.blockmanagement.BlockManager: Number of under-replicated blocks = 0

2018-05-16 20:41:12,293 INFO org.apache.hadoop.hdfs.server.blockmanagement.BlockManager: Number of over-replicated blocks = 0

2018-05-16 20:41:12,294 INFO org.apache.hadoop.hdfs.server.blockmanagement.BlockManager: Number of blocks being written = 0

2018-05-16 20:41:12,294 INFO org.apache.hadoop.hdfs.StateChange: STATE* Replication Queue initialization scan for invalid, over- and under-replicated blocks completed in 32 msec

2018-05-16 20:41:32,282 INFO org.apache.hadoop.hdfs.StateChange: STATE* Safe mode ON, in safe mode extension.

The reported blocks 12 has reached the threshold 0.9990 of total blocks 12. The number of live datanodes 1 has reached the minimum number 0. In safe mode extension. Safe mode will be turned off automatically in 9 seconds.

2018-05-16 20:41:42,285 INFO org.apache.hadoop.hdfs.StateChange: STATE* Leaving safe mode after 50 secs

2018-05-16 20:41:42,285 INFO org.apache.hadoop.hdfs.StateChange: STATE* Safe mode is OFF

2018-05-16 20:41:42,285 INFO org.apache.hadoop.hdfs.StateChange: STATE* Network topology has 1 racks and 1 datanodes

2018-05-16 20:41:42,285 INFO org.apache.hadoop.hdfs.StateChange: STATE* UnderReplicatedBlocks has 0 blocks

2018-05-16 20:48:30,132 INFO org.apache.hadoop.hdfs.server.namenode.top.window.RollingWindowManager: topN size for command getfileinfo is: 0

2018-05-16 20:48:30,132 INFO org.apache.hadoop.hdfs.server.namenode.top.window.RollingWindowManager: topN size for command listStatus is: 0

2018-05-16 20:48:30,132 INFO org.apache.hadoop.hdfs.server.namenode.top.window.RollingWindowManager: topN size for command * is: 0

2018-05-16 20:48:30,134 INFO org.apache.hadoop.hdfs.server.namenode.top.window.RollingWindowManager: topN size for command getfileinfo is: 1

2018-05-16 20:48:30,135 INFO org.apache.hadoop.hdfs.server.namenode.top.window.RollingWindowManager: topN size for command listStatus is: 1

2018-05-16 20:48:30,135 INFO org.apache.hadoop.hdfs.server.namenode.top.window.RollingWindowManager: topN size for command * is: 1

2018-05-16 20:48:30,135 INFO org.apache.hadoop.hdfs.server.namenode.top.window.RollingWindowManager: topN size for command getfileinfo is: 1

2018-05-16 20:48:30,135 INFO org.apache.hadoop.hdfs.server.namenode.top.window.RollingWindowManager: topN size for command listStatus is: 1

2018-05-16 20:48:30,135 INFO org.apache.hadoop.hdfs.server.namenode.top.window.RollingWindowManager: topN size for command * is: 1

2018-05-16 21:05:49,092 ERROR org.apache.hadoop.hdfs.server.namenode.NameNode: RECEIVED SIGNAL 15: SIGTERM

2018-05-16 21:05:49,224 INFO org.apache.hadoop.hdfs.server.namenode.NameNode: SHUTDOWN_MSG:

/**

SHUTDOWN_MSG: Shutting down NameNode at duquan-VirtualBox/127.0.1.1
**/

5.3本章小结

使用经典模型对系统进行测试，测试过程中出现部分异常，经过调整得以解决。测试结果表明，系统基本功能实现，运行结果正确。

10. 53140204_关鸿睿_计算机科学与技术_基于Apache Flume的数据收集与分析系统的设计与实现_第10部分

总字数：1049

相似文献列表　文字复制比：0%(0)　疑似剽窃观点：(0)

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

第六章总结与展望

6.1 全文概述

本文从大数据时代的到来切入，分析了互联网数据收集与分析系统的研究背景与发展现状。介绍了进入大数据时代以来，传统的数据挖掘模型已经无法满足当前的互联网数据量，从数据的收集、存储到数据的分析与处理，遇到很多的难题和大数据研究人员所取得实质性的进展。其中，主要介绍的是Hadoop与Apache Flume。

关于Hadoop的部分介绍了HDFS这一高效的分布式文件系统，以及MapReduce数据分析模型。为了更好地了解和熟悉数据收集和分析的过程，在写论文的同时，尝试建立基于Apache Flume的数据收集和分析系统。首先建立了用户端，用作数据源，由Client端发出数据，之后配置Flume接收并简单处理数据，通过接口与Hadoop集群连接，经过HDFS的传输，数据到达MapReduce系统，可以对数据进行简单的分析与处理，并得出相应结论。最后对系统进行测试，测试系统的准确性与稳定性。

6.2 问题总结

在论文撰写与系统实现过程中，遇到过一部分难以解决的问题。

首先，不够熟悉Linux系统的使用，在命令行处理方面步履维艰，不论是文件的操作以及相应执行过程都浪费了许多时间与精力。其次，在Hadoop集群配置方面经过长时间的尝试，只配置出伪Hadoop集群，可以实现一部分的功能，但是无法全部发挥Hadoop的性能。在Flume的重构过程中，也遇到了许多问题，接口类型的选择以及Agent的创建与工作过程，都出现或多或少的小插曲。除此之外，Flume在工作过程中有时会发生重启或宕机的情况，因此数据优势会出现重复发送或丢失的现象，因此整个系统的稳定性还需要日后自己不断地学习与钻研来解决问题。

总体而言，在整个毕业设计的过程中，第一次亲身接触到了大数据的工作系统，系统的了解到了数据的收集与分析过程，感受到了大数据的魅力。

6.3 未来展望

经过本次毕设设计，大数据为我打开新世界的大门。在学习的过程，深刻地认识到自己知识的欠缺。论文中的内容甚至不能算得上大数据的皮毛，还有无数的只是等待着我去探索，去学习。

在收集材料的过程中，看到了大数据时代从崛起到逐渐走上世界舞台，愈发地感觉得到未来是属于大数据的。尤其是区块链的出现，使得大数据的地位再一次得到显著地提升。作为一名计算机专业的学生，以后走向社会，无论在任何方向的工作岗位上，大数据都将与我们的工作和生活息息相关，紧密相连。因此，了解、熟悉并掌握一部分大数据有关的知识势在必行。