DOI:10.13718/j.cnki.xsxb.2017.04.019

基于 Hadoop 与 Storm 的日志实时处理系统研究①

李 洋[1], 吕家恪[1,2]

1. 西南大学 计算机与信息科学学院,重庆 400715; 2. 重庆市数字农业重点实验室,重庆 400716

摘要:日志数据记录着丰富的信息,具有较高的实用价值,但在当今大数据时代环境下,数据量的陡增为日志数据的处理带来了挑战. 为了有效地解决海量日志数据处理面临的瓶颈问题,本文整合 Hadoop 和 Storm 分布式框架,构建一种融合了实时计算与离线计算的分布式日志实时处理系统. 系统架构由数据服务层、业务逻辑层和 Web 展示层组成,数据服务层使用 Flume 实时采集日志数据,并分别采用 Kafka 与 HBase 完成实时日志流数据的缓冲和系统数据的持久化存储;业务逻辑层利用 Storm 对实时日志流数据进行实时分析,并使用 Hadoop 的计算引擎 MapReduce 结合数据挖掘技术完成对海量历史日志数据的离线分析,离线分析的结果为实时分析提供支持、参考;Web 展示层负责日志数据及其分析结果的展示. 实验结果表明,系统能有效地解决日志数据的采集存储、实时日志流数据的实时分析和历史日志数据的离线分析等问题,并成功地融合了 Hadoop 与 Storm 各自的优势,为日志数据的采集和分析系统的构建提供新的技术参考.

关 键 词:日志;Hadoop;Flume;Storm;HBase

中图分类号:TP399 **文献标志码**:A **文章编号**:1000-5471(2017)04-0119-08

作为记录系统与网络用户行为的管理工具,日志文件包含了十分重要的信息,它在网络管理、用户行为分析等领域应用广泛. 日志数据的采集和分析一直以来都是国内外学者研究的重点和热点[1-2],数据挖掘、机器学习等技术被广泛应用于日志分析中[3-5],用以充分提取日志数据所蕴含的有价值的信息. 随着互联网的高速发展,智能设备、网络设备、系统及服务程序等产生的日志数据的规模呈几何级数增长,传统的单机处理模式无论是存储还是计算上都面临日益严峻的瓶颈问题,云计算技术正好为此提供了新的处理思路[6]. Hadoop 生态圈中的 HBase 等 NoSQL 为大规模、高并发的日志数据存储提供了较好的支持[7];而在日志分析方面,Hadoop 通过高吞吐率的 MapReduce 并行计算完成对大规模日志数据的离线批处理[8],而 Storm 使用低时延的流式计算模式保证实时日志流数据处理的实时性[9]. 为整合实时计算与离线批处理的优点,Google 与 Twitter 发布了 Lambda 架构[10]作为大数据实时处理的框架. 但 Lambda 架构复杂,开发与运维的复杂性较高,计算过程中实时计算与离线批计算交互性不强[11]. 为了有效地解决海量日志数据处理面临的瓶颈问题,本文拟整合 Hadoop 与 Storm 设计并实现一个融合实时计算与离线计算的分布式日志实时处理系统,该系统可以用来解决日志数据的采集存储、实时日志流数据的实时分析和历史日志数据离线分析后的规则提取等问题,并具有一定的实时性与吞吐率.

① 收稿日期:2016-10-31
基金项目:中央高校基本科研业务费专项资金资助项目(XDJK2014B034).
作者简介:李 洋(1991-),男,贵州长顺人,硕士研究生,主要从事大数据与数据挖掘的研究.

1 Flume，Hadoop 和 Storm 分析

1.1 Flume

Flume[12]是 Cloudera 公司开源的分布式日志收集服务系统. Flume 以 Agent 为最小的独立运行单位，一个 Agent 主要由 Source，Channel，Sink 3 大组件组成. Source 将外部数据源的数据封装成 Flume 数据模型的最小单位 event，并把数据存储到 Channel 中，Sink 负责取出 Channel 中的数据并转发到数据宿中.

1.2 Hadoop

Hadoop[13]是 Apache 开发的分布式系统基础框架，提供分布式存储和计算，主要包括分布式资源调度程序 YARN、分布式文件系统 HDFS 和并行处理计算引擎 MapReduce. 此外还有不少周围组件，如分布式应用程序协调服务组件 Zookeeper、分布式消息发布订阅系统 Kafka、分布式数据库 HBase[14]等.

1.3 Storm

Storm[15]是 Twitter 公司开源的分布式实时流处理框架，可以实现单节点百万级的数据处理与运算. 集群中最重要的 2 个组件是 Spout 和 Bolt. Spout 负责从外部数据源中不间断地获取数据，并以元组的形式发送给相应的 Bolt. Bolt 负责对接收到的流数据进行计算处理.

2 系统架构

为适应大数据环境下的实时处理任务，系统架构设计为数据服务层、业务逻辑层和 Web 展示层 3 个层次，如图 1 所示. 数据服务层承担日志数据的采集分发与系统数据的存储任务；业务逻辑层负责历史日志数据的离线分析与低时延的实时日志流数据计算处理；Web 展示层实现日志与分析结果的可视化展示. 其中系统的支持服务由 YARN 与 Zookeeper 提供.

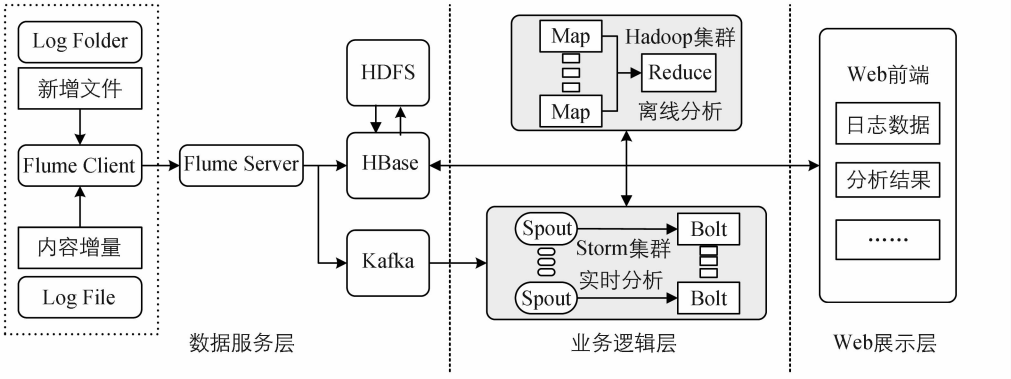

图 1 系统架构

2.1 数据服务层

数据服务层使用 Flume 实现日志数据的实时采集，使用 Kafka 实现实时日志流数据的缓冲以保证数据的完整性与顺序性，使用 HBase 实现海量系统数据的分布式存储. 采集端是部署在应用服务器上的 Flume Agent，负责监控采集指定目录下的新增文件与指定日志文件的增量内容，并推送到接收端. 接收端是部署在分布式集群中的 Flume Agent，对采集到的数据预处理后推送给 HBase 和 Kafka.

采集端与接收端分别配置 Flume Client 和 Flume Server 2 个 Agent，如图 2 所示. 在采集端，Tail Source组件使用 Linux 原生类型 exec 的执行命令：tail -f /pathname/filename 采集指定文件尾部新写入的每条日志内容；Spool Source 组件使用自定义反序列器实现日志数据读取的自定义格式以完成对指定目录下新增文件的内容采集. 在接收端，Kafka Sink 组件负责将接收到的 event 发送到 Kafka 集群中指定的数据队列中. HBase Sink 组件使用自定义编译的序列化器来实现数据存储过程中 event 字符串分割方式

以及 RowKey 生成的自定义设置. Source 与 Sink 之间的缓冲区本文采用基于内存的 Memory Channel 来实现高吞吐率，而 Agent 之间的通信采用 Flume 内置的 RPC Source 完成. 日志数据存储在 HBase 中，其中历史日志表设置不同列族以满足不同类型日志的存储，表中每条日志数据的唯一标识 Rowkey 由日志采集时间和日志 ID 组成.

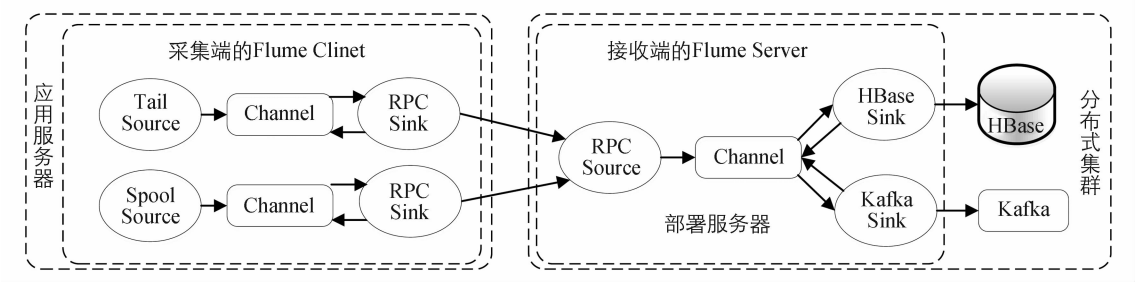

图 2　数据服务层的数据流程

2.2　业务逻辑层

业务逻辑层中日志分析的数据流程如图 3 所示，MapReduce 集群利用并行批计算对历史日志数据进行离线分析，提取大量历史数据所蕴含的信息，并将信息存储在 HBase 中；Storm 集群结合离线分析的结果对 Kafka 缓冲区中的日志流数据进行实时计算分析.

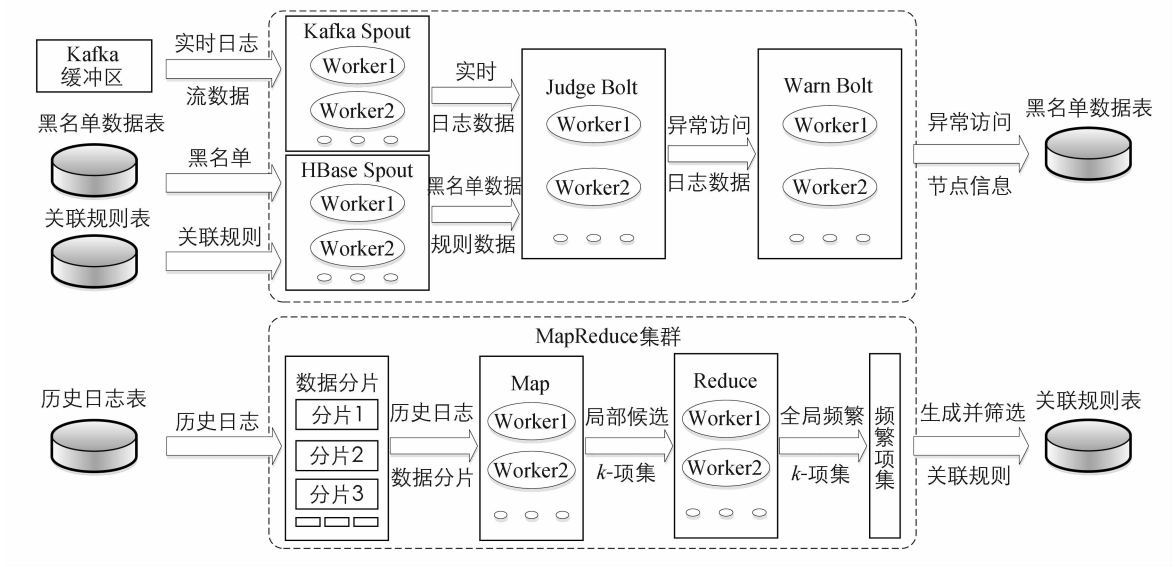

图 3　日志分析的数据流程

在历史日志数据离线分析中，以关联规则[16]挖掘的经典算法 Apriori 为例，使用基于 MapReduce 模型的 MapReduce-Apriori 分布式算法实现历史日志数据的离线分析. 算法原理：将原始事务数据集划分为 n 个数据分片，并分布在不同的节点上，每个分片由一个 Map 任务节点进行处理，m 个 Map 任务节点分别对这 n 个数据分片进行计算处理并输出键值对格式的〈候选 k -项集，1〉，在 Reduce 任务节点对所有 Map 任务输出的局部候选项集键值对进行合并，并统计出全局候选 k -项集及其支持度，筛选出满足最小支持度的全局频繁 k -项集；最后筛选出数据项较多并且内容包含网络异常事件的频繁项集，由这些频繁项集直接生成满足最小置信度且推断结果为发生网络异常事件的关联规则，并筛选掉内容出现重复的规则，经过筛选后的规则将存储到 HBase 的关联规则表中用以支持实时分析.

在实时日志数据实时分析中，Storm 的 Kafka Spout 和 HBase Spout 作为 Judge Bolt 的数据源分别读取实时日志流数据、黑名单与关联规则数据. Judge Bolt 组件根据黑名单与关联规则数据对日志数据进行解析，判断是否发生网络异常事件，并做出相应的处理(将日志数据与关联规则和黑名单进行匹配，若满足

匹配则直接将该条日志数据的节点信息推送到 Warn Bolt 组件中，若不匹配 Judge Bolt 将继续对该条数据进行解析). 以防火墙日志解析为例，解析过程中对该条日志中主机访问的各项数据内容进行检查，主要检查 priority 和 operation 2 个数据项的内容，如果这 2 个数据项的内容为 Warning 和 Deny，则判定访问发生异常，并记录该访问主机的 IP 及其异常访问的次数，若在规定时间窗口内主机的异常访问次数超过阈值，则将该主机的日志数据推送到 Warn Bolt 组件中. Warn Bolt 组件负责接收发生网络异常事件的节点信息，发出异常警报并将该主机 IP 存入 HBase 的黑名单数据表中，同时更新该 IP 的异常访问次数.

2.3 Web 展示层

在 Web 展示层采用 Java Web 技术中的 Struts 框架进行可视化页面开发，用户通过 Web 终端完成日志管理并观察日志数据离线分析与实时分析的结果.

3 实验结果与分析

3.1 实验环境

在 vSphere 虚拟化平台上构建 5 台操作系统为 Ubuntu 14.04.1 的虚拟计算机(主机配置信息为：CPU Xeon L5420 @2.50 GHz，内存 4G，硬盘 200 GB)，分布式集群部署情况如表 1 所示. 其中 Storm 搭建在 YARN 之上，并在 host1 主机与应用服务器上安装配置 Flume Agent，系统开发环境为 Java 1.8.0_45.

表 1 分布式集群中节点的部署信息

主机名	IP 地址	Hadoop Node	Hadoop 进程	Zookeeper 进程	HBase 进程	Kafka 进程	Storm node	Storm 进程
host1	172.27.5.247	Master	NameNode ResourceManager SecondaryNameNode	无	HMaster	无	无	无
host2	172.27.5.248	Slave	DataNode NodeManager	QuorumPeerMain	HRegionServer	Kafka	Nimbus	Nimbus MasterServer Core
host3	172.27.5.249	Slave	DataNode NodeManager	QuorumPeerMain	HRegionServer	Kafka	Supervisor	Supervisor
host4	172.27.5.250	Slave	DataNode NodeManager	QuorumPeerMain	HRegionServer	Kafka	Supervisor	Supervisor
host5	172.27.5.251	Slave	DataNode NodeManager	QuorumPeerMain	HRegionServer	Kafka	Supervisor	Supervisor

3.2 实验设计

1) 日志采集实验

将 Flume Client，Flume Server 分别部署在某校园网的 Web 服务器和服务器 host1 上来完成日志采集，并将数据存储到 HBase 的历史日志表中. 实验主要分为两部分：采集指定的 apache 日志文件中 access.log 与 error.log 的增量内容，分别将这 2 种日志数据存入历史日志表中的 apacheaccesslog，apacheerrorlog 列族中；采集/var/log 目录下新增文件的内容，存入历史日志表中的 apachenewlog 列族中. 在第一部分实验中，系统从 08：00：00 运行到 24：00：00，在系统运行期间每隔 4 小时记录采集到的数据量、处理时延等相关数据；第二部分实验中使用 4 个 100 000 条日志事务总大小为 19 659 KB 的相同日志数据文件作为新增文件.

2) 历史日志数据离线分析实验

实验数据源于 VAST Challenge 2013 数据集中的 Firewall 日志，日志记录了某跨国公司内部网络防火墙一周的工作信息，共有 22 个不同的数据项，主要包括 srcIp，destIp，destPort，time，opreation，protocol，priority 等，数据集中记录了不少的网络异常事件. 本节实验使用数据集中2013-04-10 07：02：35 到 2013-04-15 09：36：41 时间段中的数据，共有 16 572 000 条事务. 实验使用基于 MapReduce 模型的

Apriori 算法提取数据集中的关联规则，并在分布式集群环境和本地单机环境下进行算法运行效率的对比. 因为异常事件在整个数据集中出现的频率较低，为保证算法的性能，本节实验中算法的最小支持度设置为 0.08，而为了得到高可信度的规则，最小置信度设置为 0.8.

3) 实时日志数据实时分析实验

本实验主要统计系统运行过程中实时分析的警报数、误报数、漏报数. 考虑到日志数据的特殊性和相关性，为了能使用上一节实验分析出的关联规则，本次研究继续使用 Firewall 日志中最后一部分数据来模拟实时日志流数据. 实验数据采用 Firewall 日志中 2013-04-15 09：36：42 到 2013-04-15 10：00：00 时段的数据，共有 28 931 条事务. 把数据导入 MySQL 中，每 10 ms 顺序取出一条事务，日志生成时间改为系统当前时间，并推送到 Kafka 集群中来模拟实时日志流数据. 为检测系统对新危险节点的应对能力，每隔 10 s 向 Kafka 集群推送一条新模拟节点的异常访问日志.

3.3　实验结果与分析

1) 日志采集

由表 2 看出，系统运行 16 h，成功采集到 8 715 条实时日志，对实时日志数据的平均处理时延大约为 1.015 s，体现了系统采集模块对实时数据的有效处理. 表 3 则体现了采集模块对离线文件的有效处理，最快每秒大约可以处理 328 个 event，处理速度最高可达到 64.59 kb/s. 实验结果表明系统能有效完成日志数据的采集与存储，并具有一定的实时性.

表 2　文件增量采集结果

系统运行时间/h	采集日志数/条	处理时延/s
4	3 076	1.013
8	4 056	1.012
12	6 677	1.016
16	8 715	1.014

表 3　新增文件采集结果

文件个数	处理时间/s	文件处理速度/(kb/s)	event 处理速度/s
1	314	62.60	318.47
2	623	63.11	321.02
3	913	64.59	328.58
4	1 265	62.16	316.20

2) 历史日志数据离线分析

系统利用内容包含网络异常事件的 6 -项频繁项集提取出了 60 条关联规则，经过筛选后得出 4 条规则作为 Storm 实时分析的支持依据，其余 56 条规则的内容在这 4 条规则中都有出现. 这 4 条关联规则的内容如表 4 所示，包括提取出的关联规则及其支持度计数、支持度与置信度. 如第一条关联规则“srcIp：10.12.15.152，srcIPNumber：168562584，messageCode：ASA-4-106023，protocol：TCP⇒防火墙拒绝主机访问”表示防火墙频繁拒绝 IP 为 10.12.15.152、IPNumber 为 168562584 的主机基于 TCP 协议访问系统，消息代码为 ASA-4-106023，支持度与置信度分别为 20.50% 与 99.99%，该规则揭示了 10.12.15.152 是危险节点. 实验结果表明系统能够从一定规模的日志数据中提取出关联规则.

由图 4 可知，集群环境比单机环境运行提取相同频繁项集的消耗时间大概减少了一半. 因为集群运行时将任务分成多个小任务交由多个节点并行批处理，较之单节点处理显著提高了运行效率. 2 条曲线在后半段趋于平行，由此看出在提取 6～8 项频繁项集时 2 种运行方式所花的时间差异不大. 但在提取 1～5 项频繁项集时单机环境运行时间大大高于集群环境运行时间. 这是因为 1～5 项频繁项集中的候选项较多，需要计算的数据量也较大，针对达到一定规模的数据量时分布式批处理更能体现它的优势. 实验结果体现了

结合分布式批处理的数据挖掘算法在处理一定规模的数据时的优越性，也说明了系统的离线计算层具有较高的吞吐率.

表4 历史日志数据离线分析提取出的关联规则

防火墙状态	关联规则	支持度计数	支持度/%	置信度/%
拒绝访问	srcIp：10.12.15.152，srcIPNumber：168562584，messageCode：ASA-4-106023，protocol：TCP	3 397 532	20.50	99.99
拒绝访问	srcIp：10.13.77.49，srcIPNumber：168643889，messageCode：ASA-4-106023，protocol：TCP	1 687 655	10.18	99.87
拒绝访问	srcIp：10.138.235.111，srcIPNumber：176876399，messageCode：ASA-4-106023，protocol：TCP	1 402 862	8.47	99.99
拒绝访问	srcIp：10.6.6.7，srcIPNumber：168166919，messageCode：ASA-4-106023，protocol：TCP	1 461 421	8.82	99.99

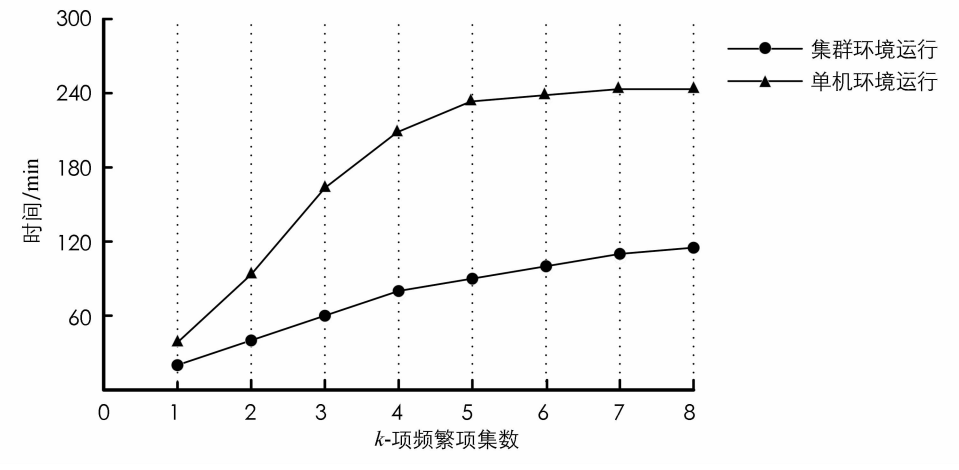

图4 MapReduce-Apriori算法提取k-项频繁项集的时间消耗

3) 实时日志数据实时分析

实时分析结果的统计情况如表5所示，从表中可看出系统能有效地针对网络异常事件做出警报处理，但也存在一定的误报率和漏报数. 误报的存在是由于系统一致拒绝关联规则和黑名单中危险节点的访问造成的，即使危险节点的正常访问也会被系统拒绝. 此外模拟的新异常节点与规则和黑名单均不匹配，且给定时间窗口内的异常访问计数没超过阈值，系统不会对此做特殊处理，这样导致了系统存在一定的漏报数. 表6的性能对比结果说明了基于Storm集群的实时分析处理时延较低，因为Storm集群在数据的读取与计算中都采用分布式流式处理，多个节点同时分别处理不同的任务，与单节点的处理模式相比节省了一定的处理时间. 实验结果表明系统成功地实现了对日志流数据的实时分析，对网络异常事件有一定的识别能力，能够及时发现网络中的异常事件，实时计算层具有较高的实时性.

表5 实时日志数据实时分析结果统计

系统运行时间/min	警报数/个	误报数/个	误报率/%	漏报数/个
1	5 246	704	13.41	6
2	10 499	1 433	13.64	12
3	15 709	2 167	13.79	18
4	20 761	2 852	13.73	24

表 6　集群与单机环境下实时分析性能对比

系统运行时间/min	集群环境处理时延/s	单机环境处理时延/s
1	0.525	0.575
2	0.535	0.557
3	0.554	0.657
4	0.527	0.574

4　结　　语

本文在研究云计算理论的基础上，基于 Hadoop 与 Storm 设计并实现了一个集成日志数据采集、存储、管理与分析功能的分布式日志实时处理系统. 目前 Storm 虽然很好地支持了实时流计算，但是它仍不适用于大规模数据的复杂计算中，如 Storm 在关联规则挖掘的应用中也只能对给定时间窗口内一定规模的事务数据进行计算处理[17]. 因此本文在进行 Storm 实时计算的同时，利用 MapReduce 完成对历史日志数据的离线分析. 相较于 Lambda 架构[11]，在数据处理层中本文实时计算与离线计算的交互性更强，实时计算能直接依赖离线计算的结果；此外本文将 Storm 移植到 Hadoop 的资源管理系统 YARN 之上，避免了一定的 Storm 运维开销. 本文采用的大数据实时处理框架在海量无线传感信息存储、网络安全管理、智能家居、智慧城市、实时反馈的股票分析预测等多个领域均能被沿用，具有较好的应用前景.

参考文献：

[1] 程　苗，陈华平. 基于 Hadoop 的 Web 日志挖掘 [J]. 计算机工程，2011，37(11)：37－39.

[2] OLINER A，GANAPATHI A，XU W. Advances and Challenges in Log Analysis [J]. Communications of the ACM，2012，55(2)：55－61.

[3] CHEN M S，PARK J S，YU P S. Efficient Data Mining for Path Traversal Patterns [J]. Knowledge & Data Engineering IEEE Transactions on，1998，10(2)：209－221.

[4] CARVALHO C，JORGE A M，SOARES C. Personalization of E-newsletters Based on Web Log Analysis and Clustering [C]// Web Intelligence，2006. WI 2006. IEEE/WIC/ACM International Conference on. Hong Kong：IEEE，2006：724－727.

[5] YADAV M P，KESERWANI P K，SAMADDAR S G. An Efficient Web Mining Algorithm for Web Log Analysis：E-Web Miner [C]//2012 1st International Conference on Recent Advances in Information Technology(RAIT). Dhanbad：IEEE，2012：607－613.

[6] ARMBRUST M，STOICA I，ZAHARIA M，et al. A View of Cloud Computing [J]. Communications of the ACM，2010，53(4)：50－58.

[7] 陈庆奎，周利珍. 基于 HBase 的大规模无线传感网络数据存储系统 [J]. 计算机应用，2012，32(7)：1920－1923，1977.

[8] HINGAVE H，INGLE R. An Approach for MapReduce Based Log Analysis Using Hadoop [C]//2015 2nd International Conference on Electronics and Communication Systems (ICECS). Coimbatore：IEEE，2015：1264－1268.

[9] MERA D，BATKO M，ZEZULA P. Towards Fast Multimedia Feature Extraction：Hadoop or Storm [C]//2014 IEEE International Symposium on Multimedia(ISM). TaiChung：IEEE，2014：106－109.

[10] KIRAN M，MURPHY P，MONGA I，et al. Lambda Architecture for Cost-Effective Batch and Speed Big Data Processing [C]//2015 IEEE International Conference on Big Data (Big Data). SantaClara：IEEE Computer Society，2015：2785－2792.

[11] LAKHE B. Case Study：Implementing Lambda Architecture [M]//Practical HadoopMigration. Berkeley：Apress，2016：279－302.

[12] 郝　璇. 基于 Apache Flume 的分布式日志收集系统设计与实现 [J]. 软件导刊，2014(7)：110－111.

[13] GHAZIMR，GANGODKARD. Hadoop，MapReduce and HDFS：A Developers Perspective [J]. Procedia Computer Science，2015，48：45－50.

[14] 周相兵，马洪江，苗放. 云计算环境下的一种基于 Hbase 的 ORM 设计实现 [J]. 西南师范大学学报(自然科学版)，2013，38(8)：130－135.

[15] 胡宇舟，范　滨，顾学道，等. 基于 Storm 的云计算在自动清分系统中的实时数据处理应用 [J]. 计算机应用，2014，34(z1)：96－99.

[16] 王仕平，蒋　玲，熊　江，等. 一种基于序列数的关联规则挖掘算法 [J]. 西南大学学报(自然科学版)，2011，33(3)：122－127.

[17] ANDERSON Q. Storm 实时数据处理 [M]. 卢誉声译. 北京：机械工业出版社，2014.

Research on Log Data Real-Time Processing System Based on Hadoop and Storm

LI　Yang[1]，　LV Jia-ke[1,2]

1. *School of Computer and Information Science，Southwest University，Chongqing 400715，China*；

2. *The Chongqing Key Laboratory of Digital Agriculture，Chongqing 400716，China*

Abstract：Log data record rich information is of high practical value. In today′s era of big data environment，the amount of data brought challenges to the processing of log data. In order to solve the bottleneck problem of massive log data processing effectively，Hadoop and Storm have been integrated in this paper to design and implement a distributed log real-time processing system integrated with off-line computing and real-time computing. The system architecture consists of data service layer，business logic layer and Web presentation layer. In data service layer，Flume is used to collect log data in real time，Kafka and HBase are used to achieve the real-time log stream data buffer and the system data storage. In business logic layer，Storm is used to carry on the real-time analysis to the real-time log stream data，and Hadoop computing engine MapReduce and Data Mining are used to complete the massive historical log data off-line analysis，the results of off-line analysis provide support for real-time analysis. Web presentation layer is responsible for the log and analysis results show. Experimental results show that the system can effectively solve the problem of log data acquisition and storage，real-time analysis of the real-time log stream data and depth analysis of the massive historical log data. And the advantages of Hadoop and Storm are successfully fused. The system also is a new technical reference for the construction of the log data acquisition and analysis system.

Key words：log；Hadoop；Flume；Storm；HBase

责任编辑　崔玉洁