Московский орден Трудового Красного Знамени
физико-технический институт
(государственный университет)
Факультет общей и прикладной физики
Кафедра физики взаимодействия частиц высоких энергий
Объединенный институт ядерных исследований
Учебно-научный центр

Водясов Д.Е.

Мониторинг грид сайта с применением технологий на основе Hadoop

Бакалаврская работа

Научный руководитель
Мицын С.В.

Дубна Июнь 2014

Оглавление

Введение

Грид, как метод доступа пользователей-учёных к большим объёмам вычислительных ресурсов, подразумевает объединение глобально распределённых вычислительных центров – грид-сайтов. Одно из направлений деятельности ОИЯИ также заключается в поддержке такого грид-сайта.

Мониторинг грид-инфраструктуры в целом и отдельно грид сайтов является важной составной частью процесса их поддержки и эксплуатации. Без мониторинга очень трудно, если вообще возможно, определить источники разнообразных проблем.

Существующие решения мониторинга реализуют классический подход к мониторингу грид-сайтов – после сбора данных о событиях осуществляется постобработка – группировка существенных характеристик по интервалам времени, например, в 10 минут. Таким образом, теряется часть информации о конкретных деталях событий с одной стороны, и сильно затрудняется проведение глубокого анализа и выявления взаимосвязей между событиями и их существенными характеристиками с другой.

Новый подход, входящий в широкое понятие «Big Data», в противоположность классическому, подразумевает сохранение всей информации о событиях и составление сложных, комплексных запросов для проведения глубокого анализа. Big Data – одно из современных направлений в области анализа данных, в рамках которого наибольшее распространение получило техническое решение в виде Hadoop. Это комплекс программ, компонентов и способов их эксплуатации для построения вычислительной инфраструктуры и методов сбора и анализа данных различного происхождения. В данной работе предложена схема использования инфраструктуры, построенной на основе Hadoop, для мониторинга грид-сайта и произведён сравнительный анализ с классическими решениями. [1]

Цель работы

Основной целью работы является исследование применимости Hadoop для мониторинга грид-сайта ОИЯИ.

Задачи:

1. Составить обзор текущих распространённых решений мониторинга грид-сайта.
2. Рассмотреть текущие решения мониторинга вычислительных центров на базе Hadoop.
3. Составить схему мониторинга грид-сайта с помощью Hadoop Flume.

Глава 1. Обзор решений мониторинга

Мониторинг является неотъемлемой частью администрирования любого дата-центра [2]. Разработано множество решений, как специализированных под конкретные центры (CERN Lemon), так и общие решения мониторинга (Nagios, Ganglia).

Nagios

Nagios – программа мониторинга компьютерных систем и сетей, предназначена для наблюдения, контроля состояния вычислительных узлов и служб. Включает в себя функционал:

1. Мониторинг состояния сети, узлов (загрузка процессора, использование диска, системные логи) и сервисов;
2. Оповещение в случае возникновения проблем с сервисом или узлом;
3. Составление и представление общей сводки о текущем состоянии инфраструктуры и изменении во времени.
4. Расширение функциональности с помощью плагинов (подключаемых модулей). [3]

Nagios – используется в мониторинге CRO NGI (the Croatian National Grid Infrastructure), EGEE (Enabling Grids for E-sciencE) ресурсов в центральной Европе.

Хотя Nagios и имеет детализованное текущее представление о системе, и с помощью подключаемых модулей можно расширять детализацию и функциональность, главный недостаток Nagios состоит в том, что исторические данные теряют детализацию. Отсутствует функциональность, позволяющая находить взаимосвязи в динамике. Это объясняется тем, что одна из целей Nagios состояла в том, чтобы не использовать большие вычислительные ресурсы для своей работы, которые понадобились бы для обработки исторических данных. [4]

Ganglia

Ganglia - масштабируемая распределенная система мониторинга кластеров параллельных и распределенных вычислений и облачных систем с иерархической структурой. Включает в себя функционал:

1. Мониторинг состояния сети, узлов (загрузка процессора, использование диска, системные логи) и сервисов в реальном времени;
2. Составление и представление общей сводки о текущем состоянии инфраструктуры и изменении во времени;

3. Расширение функциональности с помощью плагинов (подключаемых модулей).

Ganglia – используется в мониторинге множества проектов MIT (Massachusetts Institute of Technology), Twitter, Wikipedia, CERN, NASA и других.

Ganglia, так же как и Nagios, может расширять функциональность при помощи подключаемых модулей. Их функционал довольно сильно различается, и поэтому совместное их использование может компенсировать недостатки обоих продуктов, однако данное решение является слишком громоздким для администрирования. [5]

Lemon

Lemon – программа мониторинга компьютерных систем и сетей, предназначена для наблюдения, контроля состояния вычислительных узлов и служб. Включает в себя функционал:

1. Мониторинг состояния сети, узлов (загрузка процессора, использование диска, системные логи) и сервисов;
2. Составление и представление общей сводки о текущем состоянии инфраструктуры и изменении во времени через web-интерфейс.

Lemon - используется в CERN. Мониторит около 2100 компьютеров в 100 кластерах, собирает ежедневно 1ГБ данных.

В сравнении с Ganglia и Nagios, Lemon ограничен в функциональности и в гибкости задач. [6]

Глава 2. Обзор инструментов мониторинга и анализа на основе Apache Hadoop.

Hadoop – проект фонда Apache Software Foundation, свободно распространяемый набор утилит, библиотек и программный каркас для разработки и выполнения распределённых программ. Он позволяет производить распределённую обработку больших объёмов данных на компьютерных кластерах с помощью простых моделей программирования. Он поддерживает масштабирование от единичного сервера до тысяч машин, каждая из которых предоставляет вычислительные мощности и память. Вместо того чтобы полагаться на предоставляемую высокую надёжность оборудования, программное обеспечение само обнаруживает и обрабатывает сбои на уровне приложений, таким образом предоставляя высокодоступную систему на кластере компьютеров, каждый из которых может быть склонен к сбоям.

Apache HDFS

HDFS – распределённая файловая система, используется как основной компонент для хранения данных. Предназначена для хранения файлов больших размеров, распределённых между узлами вычислительного кластера. Поддерживается репликация, которая обеспечивается устойчивость распределённой системы к отказам отдельных узлов.

Apache Flume

Apache Flume – распределённый, надёжный сервис для эффективного сбора, агрегации и передачи больших объёмов лог файлов из различных источников в централизованное хранилище информации. Однако использование не ограничивается только лишь сбором лог файлов: так как присутствует возможность расширения вариантов источников данных, Flume может быть использован для передачи больших объёмов данных о событиях мониторинга, включая сетевой трафик, состояния о узлах в виде метрик и прочее.

Этот инструмент и предназначен для управления потоками данных: собирать их из различных источников и направлять их в централизованное хранилище. [7]

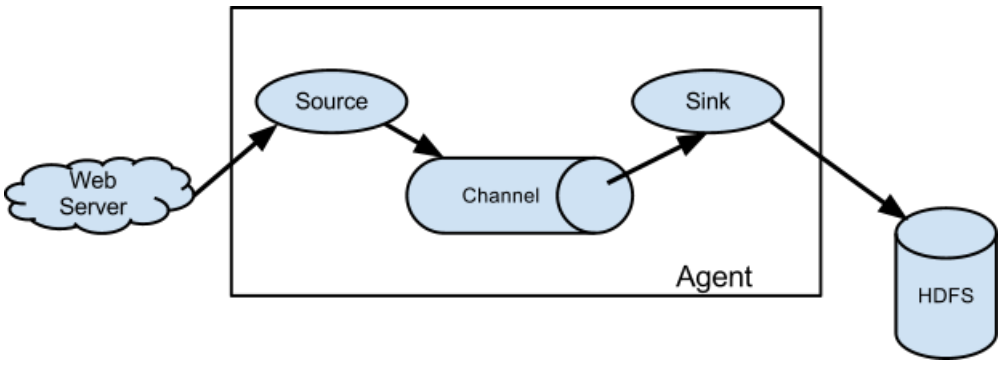

Рис.1. Принципиальная схема работы Flume.

Поток начинается с клиента, который передает событие агенту (говоря более точно, на источник в составе агента). Источник, получивший событие, передает его на один или несколько каналов. Из каналов событие передается на стоки, входящие в состав того же агента. Он может передать ее другому агенту, или (если это конечный агент) — на узел назначения.

Так как источник может передавать события на несколько каналов, потоки могут направляться на несколько узлов назначения. Это наглядно показано на рисунке ниже: агент считывает событие в два канала (Канал 1 и Канал 2), и затем передает их в независимые стоки.

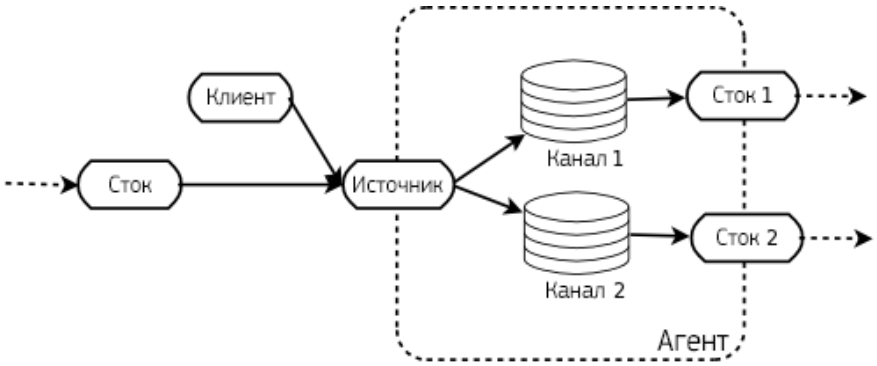

Рис.2. Распределение потоков по разным направлениям.

Несколько потоков можно объединить в один. Для этого нужно, чтобы несколько источников в составе одного и того же агента передавали данные на один и тот же канал. Схема взаимодействия компонентов при объединении потоков показана на рисунке ниже (здесь каждый из трех агентов, включающий несколько источников, передает данные на один и тот же канал и затем на сток)(Рис.3)

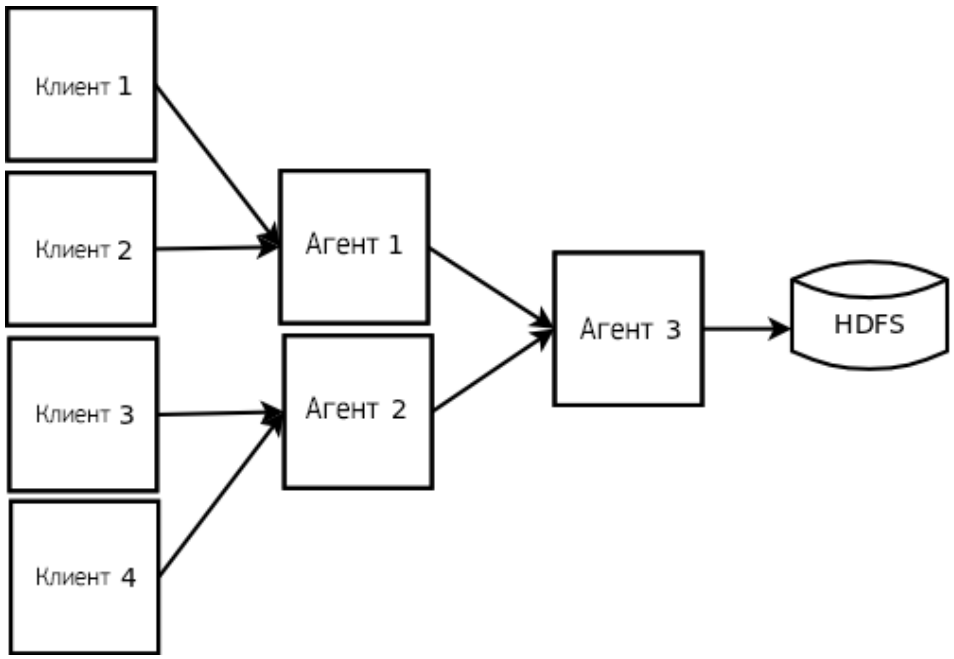

Рис.3. Взаимодействие компонентов при объединении потоков. [8]

Map/Reduce в Hadoop.

Map/Reduce – модель распределённых вычислений, представленная компанией Google (не путать модель с её одноименной реализацией от Google), используемая для параллельных вычислений над очень большими, несколько петабайт, наборами данных в компьютерных кластерах.

Это фреймворк для вычисления некоторых наборов распределенных задач с использованием большого количества компьютеров (узлов), образующих кластер. Работа Map/Reduce состоит из двух шагов: Map и Reduce.

На Map-шаге происходит предварительная обработка входных данных. Для этого один из компьютеров (называемый главным узлом – master node) получает входные данные задачи, разделяет их на части и передает другим компьютерам (рабочим узлам – worker node) для предварительной обработки. Название данный шаг получил от одноименной функции высшего порядка. На Reduce-шаге происходит свёртка предварительно обработанных данных. Главный узел получает ответы от рабочих узлов и на их основе формирует результат – решение задачи, которая изначально формулировалась. [9]

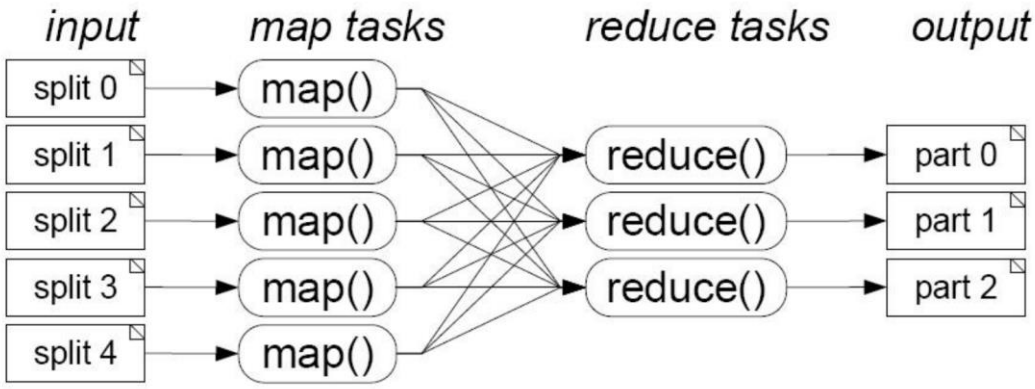

Рис. 4. Модель вычислений Map/Reduce

Преимущество Map/Reduce заключается в том, что он легко распараллеливается и позволяет распределённо производить операции предварительной обработки и свертки. Операции предварительной обработки выполняются независимо друг от друга и могут производиться параллельно (хотя на практике это ограничено источником входных данных и/или количеством используемых процессоров). Аналогично, множество рабочих узлов могут осуществлять свертку – для этого необходимо только чтобы все результаты предварительной обработки с одним конкретным значением ключа обрабатывались одним рабочим узлом в один момент времени. Хотя этот процесс может быть менее эффективным по сравнению с более последовательными алгоритмами, Map/Reduce может быть применен к большим объёмам данных, которые могут обрабатываться большим количеством серверов. Так, Map/Reduce может быть использован для сортировки петабайта данных, что займет всего лишь несколько часов. Параллелизм также дает некоторые возможности восстановления после частичных сбоев серверов: если в рабочем узле, производящем операцию предварительной обработки или свертки, возникает сбой, то его работа может быть передана другому рабочему узлу (при условии, что входные данные для проводимой операции доступны).

Фреймворк в большой степени основан на функциях map и reduce, широко используемых в функциональном программировании, хотя фактически семантика фреймворка отличается от прототипа.[10]

Анализ данных в Hadoop

Так как разработка с помощью Map/Reduce подразумевает работу с некоторыми деталями реализации подсистемы обработки, были построены решения, позволяющие пользователю выражать вычисления на абстракциях более высокого уровня.

Hive – это надстройка над Hadoop для того, чтобы облегчить выполнение таких задач, как суммирование данных, непрограммируемые запросы и анализ больших наборов данных:

- Hive может быть использован теми, кто знает язык SQL.
- Hive создает задания MapReduce, которые исполняются на кластере Hadoop.
- Определения таблиц в Hive надстраиваются над данными в HDFS.

Pig – это платформа, предназначенная для анализа больших наборов данных и состоящая из языка высокого уровня для написания программ анализа данных и инфраструктуры для запуска этих программ. Язык характеризуется относительно простым синтаксисом. Написанные сценарии скрыто преобразуются в задачи MapReduce, которые исполняются на кластере Hadoop.

Apache Mahout — это новый открытый проект Apache Software Foundation, основной целью которого является создание масштабируемых алгоритмов машинного обучения. Mahout содержит реализации кластеризации, классификации, CF (коллаборативной фильтрации) и эволюционного программирования. В случаях необходимости он использует библиотеку Apache Hadoop, что позволяет Mahout эффективно масштабироваться в облаке.

Рассмотрим реальное применение Mahout – кластеризацию.

Алгоритм k-Means: группирует элементы в k-кластеры, основываясь на расстоянии от этих элементов до центроида, или центра тяжести предыдущей итерации. Кластеризация данных состоит из следующих этапов:

1. Подготовка входных данных. При кластеризации текста его нужно преобразовать в числовое представление.
2. Запуск выбранного алгоритма кластеризации с помощью одного из множества поддерживающих Hadoop программ-драйверов, имеющихся в Mahout.
3. Оценка результатов.
4. Повторение по мере необходимости.

Реализация принимает два входных каталога: один для данных и один для начальных кластеров. Каталог данных содержит несколько входных файлов из SequenceFile(Key, VectorWritable), в то время как каталог кластеров содержит один или более SequenceFiles(Text, Cluster), содержащее k начальных кластеров или Canopy. Ни один из входных каталогов не модифицируют путем реализации, что позволяет экспериментировать с начальной кластеризацией и значением сходимости.

Кластеризацию Canopy можно использовать для вычисления начальных кластеров K-KMeans (Рис.5).

```
// run the CanopyDriver job
CanopyDriver.runJob("testdata", "output",
ManhattanDistanceMeasure.class.getName(), (float) 3.1, (float) 2.1,
false);

// now run the KMeansDriver job
KMeansDriver.runJob("testdata", "output/clusters-0", "output",
EuclideanDistanceMeasure.class.getName(), "0.001", "10", true);
```

Рис.5. Листинг примера кода Mahout.

В приведенном выше примере, входные данные хранятся в «TestData» и CanopyDriver выполнен с возможностью выводить на каталог «output/clusters-0». Как только драйвер будет выполнен, он будет содержать файлы, определяющие кластеризацию Canopy. После выполнения KMeansDriver выходной каталог будет иметь два или более новых каталогов: "кластеры-N", содержащие кластеры для каждой итерации, и« clusteredPoints» содержать кластерные данные. [11]

Глава 3. Методы выполнения и личный вклад

В процессе выполнения работы автор ознакомился с документацией к разнообразным компонентам Hadoop, принципиальной схемой работы Hadoop и некоторых распространённых систем мониторинга грид сайтов, а также некоторыми научными работами по теме мониторинга.

Также был установлен на один узел и настроен прототип системы сбора данных мониторинга согласно вышеупомянутой схеме, и на нём был реализован сбор данных в виде сбора модельного лог-файла, в который непосредственно заносятся события. В результате создания прототипа показано, что схема сбора данных на основе Flume согласно схеме может быть использована для сбора лог-файлов и другой информации для последующей обработки.

Выработка конкретных методов анализа и представления данных не входила в поле задач данной работы. Более того, ожидается настолько большая изменчивость методов обработки данных и запросов, что практически сложно сделать универсальное представление метода анализа.

В противоположность, текущие используемые решения анализа данных мониторинга, в основном, предполагают только группировку в интервалах времени по атрибутам, что, по сути, является просто построением гистограмм, а способы представления таких данных хорошо изучены. Также автор предполагает, что после обработки данных их объём будет достаточно мал для последующего представления в формате Excel, а средства представления таких данных тоже хорошо изучены. [12]

Заключение

Автором был установлен и протестирован прототип системы сбора данных мониторинга на основе Hadoop Flume и был проведён сравнительный анализ с текущими решениями, и выявлены следующие преимущества прототипа:

1. Масштабируемость, возможность работы со сколь угодно большим объёмом данных.

2. Удобство администрирования – отсутствие необходимости развёртывания систем Ganglia или Nagios при наличии Hadoop-инфраструктуры.

3. Возможность задавать произвольную обработку данных мониторинга через задачи MapReduce или компоненты Hadoop, что значительно упрощает администрирование мониторинга.

4. Возможность углублённого анализа

Список литературы

[1] Paul C. Zikopoulos, Chris Eaton, Dirk deRoos, Thomas Deutsch, George Lapis,Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data. 2012

[2] P.V. Dmitrienko, A.G. Dolbilov, V.V. Korenkov и др. «Development of the Monitoring System for the CICC JINR Resources». – Научный отчёт ОИЯИ. 2010-2011

[3] Nagios - The Industry Standard in IT Infrastructure Monitoring [Электронный ресурс]. URL: http://www.nagios.org/ (дата обращения: 29.05.2014)

[4] Grid Infrastructure Monitoring System Based on Nagios, E.Imamagic, D. Dobrenic, SRCE, HPDC 2007, Workshop on Grid Monitoring. [Электронный ресурс]. URL: http://osg-docdb.opensciencegrid.org/cgi-bin/RetrieveFile?docid=666;filename=nagios.pdf (дата обращения: 29.05.2014)

[5] Ganglia – an open source monitoring tool. Monitoring of Power Systems – Best Practices. M. Perlz, 2012 IBM Corporation. [Электронный ресурс]. URL: http://public.dhe.ibm.com/systems/power/community/aix/Central-VUG-Replays/Files/VUG_Webinar_Ganglia_July_2012.pdf (дата обращения: 29.05.2014)

[6] Lemon for system administrators. M.Siket, CERN-IT, 2008. URL: http://lemon.web.cern.ch/lemon/index.shtml [Электронный ресурс]. (дата обращения: 29.05.2014)

[7] “Welcome to Apache Flume – Apache Flume” [Электронный ресурс]. URL: http://flume.apache.org/ (дата обращения: 29.05.2014)

[8] “Flume 1.5.0 User Guide — Apache Flume” [Электронный ресурс]. URL: http://flume.apache.org/FlumeUserGuide.html (дата обращения: 29.05.2014)

[9] Jeffrey Dean, Sanjay Ghemawat MapReduce: Simplfied Data Processing on LargeClusters. Google Inc, 2004

[10] Ralf Lammel, Google’s MapReduce Programming Model — Revisited. DataProgrammability Team, Microsoft Corp, 2007

[11] K Shvachko. Apache Hadoop: the Scalability Update. 2011.

[12] Michael G. Noll, Running Hadoop on Ubuntu Linux (Single-Node Cluster). 2011