

báo cáo apache Storm

Nhóm: Romano



1. Thành viên

|  |  |  |
| --- | --- | --- |
| MSSV | Họ Tên |  |
| 1412274 | Nguyễn Hoàng Kim |  |
| 1412278 | Phan Khánh Lâm |  |
| 1412414 | Vương Thiên Phú |  |
| 1412557 | Phạm Đức Toàn |  |
| 1412604 | Lữ Hoàng Phi Tuấn |  |

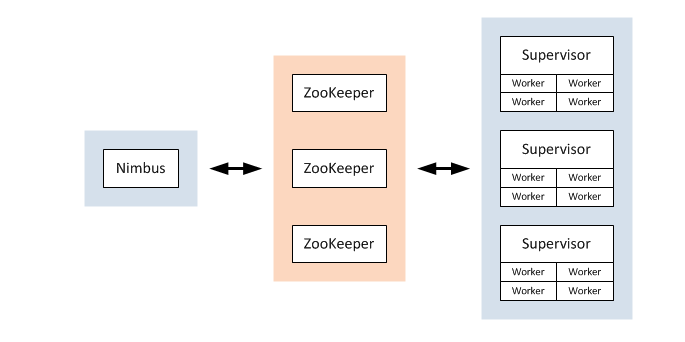
1. Khái niệm

* Apache Storm là một hệ thống tính toán thời gian thực phân phối mã nguồn mở để xử lý luồng dữ liệu. Apache Storm thực hiện các luồng dữ liệu không bị chặn một cách đáng tin cậy.
* Storm được thiết kế để xử lý khối lượng lớn dữ liệu với khả năng chịu lỗi và có thể mở rộng theo chiều ngang.
* Storm quản lý môi trường phân tán và trạng thái cluster thông qua Apache Zookeeper. Nó khá đơn giản và chúng ta có thể thực hiện song song nhiều thao tác trên các dữ liệu thời gian thực.
* Phiên bản đầu tiên ra mắt tháng 9/2011.
* Phiên bản hiện tại : 1.0.5 ( tháng 9/2017)
* [Website :storm.apache.org](https://storm.apache.org/)

1. Kiến trúc – cơ cấu hoạt động.
2. Cấu trúc :

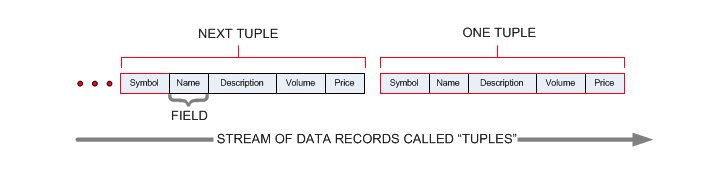
Một hệ thống Storm bao gồm hai thành phần chính là: master node được gọi là Nimbus và worker node được gọi là Supervisor. Nimbus chịu trách nhiệm phân phối các mã dữ liệu trong toàn bộ cluster, phân tác vụ đến các máy chủ và giám sát các tác vụ thất bại. Supervisor thực hiện tiếp nhận các tác vụ mà nó được phân công từ Nimbus;

Storm hoạt động dựa trên nền tảng Zookeeper với mục đích quản lý/ phối hợp giữa các node và quản lý trạng thái lưu trữ của dữ liệu.

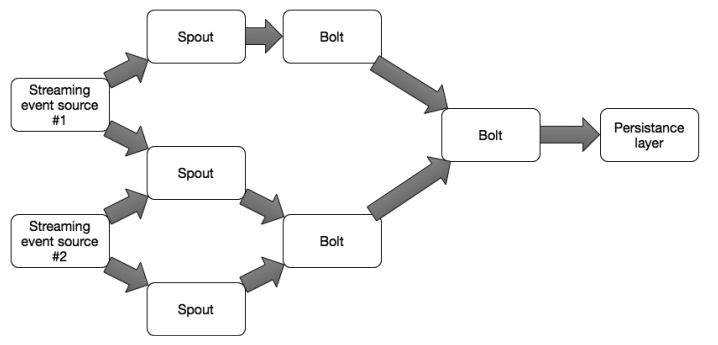


1. Cơ chế hoạt động:

* Ứng dụng xử lý luồng trong Storm được định nghĩa bởi các mô hình hệ thống. Các mô hình này thật chất là các cấu trúc luận lý phục vụ cho việc tính toán của ứng dụng.
* Các node trong mô hình sẽ quyết định phương thức (method) xử lý dữ liệu, và các node liên kết với nhau sẽ quyết định luồng (flow) xử lý dữ liệu.
* Thành phần cốt lõi của một hệ thống Storm được gọi là Stream.
* Cấu trúc của Stream bao gồm các tập hợp dữ liệu (tuples of data). Mỗi phần tử trong một tập hợp có thể là bất kỳ loại dữ liệu nào. Kết quả xử lý dữ liệu từ hệ thống Storm có thể xuất ra thành một hoặc nhiều luồng dữ liệu khác nhau; hoặc có thể chuyển đến hệ thống Kafka hay một hệ thống lưu trữ/ cơ sở dữ liệu.
* Storm cung cấp hai khái niệm cơ bản trong quá trình xử lý dữ liệu được gọi là Bolt và Spout. Bạn có thể triển khai bolt và spout để tạo ra một ứng dụng xử lý luồng dữ liệu theo mong muốn.



* Một spout có thể được xem là một nơi tiếp nhận các dữ liệu đầu vào trong mô hình kiến trúc Storm. Nó đóng vai trò thu thập các dữ liệu từ hệ thống Kafka hoặc từ Twitter API hoặc bất kỳ hệ thống nào cung cấp cơ chế xử lý dữ kiện theo luồng.
* Một bolt có thể tiếp nhận dữ liệu từ một hoặc nhiều nguồn dữ liệu mà spout chuyển đến; nó làm việc hoàn toàn dựa trên kiến trúc mà bạn đã quy hoạch ban đầu. Dữ liệu đầu ra của một bolt có thể phục vụ cho một bolt khác để tiếp tục xử lý. Bolt có thể làm được mọi thứ từ việc chay các hàm xử lý, thu thập các trường dữ liệu trong một tập hợp, phân tích các luồng dữ liệu, thực hiện xử lý streaming-join, tương tác với cơ sở dữ liệu và nhiều tính năng khác. Một hệ thống mạng lưới các thành phần bolt/spout tạo thành một mô hình hệ thống Storm như hình dưới.



* Một hệ thống như vậy sẽ hoạt động cho đến khi bạn kết thúc nó. Mỗi node trong hệ thống bạn có thể thiết lập xử lý song song và Storm sẽ tiến hành phân bổ tài nguyên theo yêu cầu dựa trên việc tạo thêm các thread xử lý. Khi có một tác vụ thất bại, Storm sẽ tự động khởi khộng lại chúng.
* Storm cung cấp ba cơ chế đảm bảo an toàn cho dữ liệu:

- At-most-once processing: đây là chế độ đơn giản nhất và thích hợp trong trường hợp một tập dữ liệu yêu cầu xử lý không quá một lần. Việc mất mát dữ liệu trong quá trình xử lý có thể được chấp nhận trong tình huống này. Nếu quá trình xử lý thất bại tại chế độc này, Storm sẽ tiến hành loại bỏ dữ liệu và không xử lý gì cả.

- At-least-once processing: chế độ này yêu cầu các tập dữ liệu (typle) phải được xử lý ít nhất một lần. Tuy nhiên, việc xử lý dữ liệu nhiều hơn một lần sẽ được áp dụng. Trong tình huống này, hệ thống của bạn sẽ phải chấp nhận có sự sai sót dữ liệu (không đáng kể) trong kết quả xử lý.

- Exactly-once processing: đây là một chế độ phức tạp và tốn kém hơn các chế độ còn lại. Thông thường, một hệ thống bên ngoài như Trident sẽ được sử dụng để đảm bảo tính chính xác của dữ liệu.

1. So sánh
2. So sánh với Hadoop

|  |  |
| --- | --- |
| Storm | Hadoop |
| Xử lý phân luồng thời gian thực | Xử lý hàng loạt |
| Phi trạng thái | Có trạng thái |
| Kiến trúc Master-Slave dựa trên sự kết hợp với Zookeeper. Thành phần master node gọi là nimbus và các slave node gọi là supervisor | Kiến trúc Master-Slave có thể/không dựa trên sự kết hợp với Zookeeper. Thành phần master node gọi là job tracker và các slave node gọi là task tracker |
| Xử lý phân luồng của Storm có thể truy cập đến 10000 thông điệp/giây trong các cluster | Hadoop Distributed File System (HDFS) sử dụng MapReduce để xữ lý khối lượng lớn dữ liệu mất vài phút hay vài giờ |
| Cấu trúc liên kết của Storm chỉ dừng khi bị tắt bởi người dùng hay gặp sự cố bất ngờ không thể khắc phục được. | Các công việc Mapreduce được thực hiện một cách tuần tự |
| Cả 2 đều phân luồng và có khả năng chịu lỗi | |
| Nếu nimbus/supervisor chết, sau khi khởi động lại sẽ tiếp tục từ chỗ dừng trước đó, do đó không có gì bị ảnh hưởng | Nếu job tracker chết thì các công việc đang thực hiện đều bị mất |

1. So sánh với Apache Spark Streaming

**Giống nhau:**

Được viết trên ngôn ngữ dựa trên nền tảng JAVA: Scala và Clojure.

|  |  |  |
| --- | --- | --- |
|  | Spark | Storm |
| Mô hình xử lí | Spark Streaming là bao bọc lại mô hình xử lí theo lô của Spark | Xử lí trực tiếp thông qua các lõi storm layer. |
| Các hàm | Gồm 2 loại phương thức phổ biến : Stream transformation operators và ouput operators. | Cung cấp rất nhiều các hàm để xử lí ở mức tuple , trong các khoảng khác nhau trong 1 luồng.  Hỗ trợ kết bảng trái, phải |
| Cơ chế đảm bảo an toàn cho dữ liệu | Tự định nghĩa mức độ sai lệch (fault tolerance). Mức độ đảm bảo phục thuộc vào công thức và toán tử output (output operator).  Khi gặp lỗi , thì các executors sẽ mất hết tất cả những mà nó đã nhận trong bộ nhớ => Spark cung cấp data checkpointing để khắc phục. | Cung cấp 3 tiến trình đảm bảo: at least once, at-most-once, exactly once.  Khi gặp lỗi, các storm deamons được tạo ra để fail-fast( sẽ có phương thức tự huỷ khi gặp sự cố) và phi trạnh thái. |
| Auto Scaling. | Hiện đang được phát triển dynamic scaling để streaming dữ liệu. | Hỗ trợ việc thiết lập cấu hình song song ở các mức topology.  Hỗ trợ cân bằng động, cho phép thêm hoặc giảm số lượng worker process mà không cần khởi động lại cluster. |
| Các ngôn ngữ tuỳ chọn | Java, Scala , Python, R. | Java, Clojure, Scala |
|  |  |  |

1. So sánh với Azure

|  |  |  |
| --- | --- | --- |
|  | Azure | Apache Storm |
| Input data source | Azure Event Hubs và Azure Blob storage | Sử dụng nhiều nguồn dữ liệu tùy theo custom code, kể cả Azure Hub và Blob |
| Input data formats | Avro, Json, CSV | Implement bất cứ định dạng file nào tùy theo code |
| Output data | Hỗ trợ đa dạng nhiều dạng output, ưu tiên các dạng Azure cung cấp | Hỗ trợ nhiều kiểu output, mỗi output có hỗ trợ custom logic, user có thể tạo mới 1 connector bằng code |
| Data-encoding | UTF8 | Đa dạng |

1. So sánh với Google DataFlow

|  |  |  |
| --- | --- | --- |
|  | Google Dataflow | Apache Storm |
| Hỗ trợ | Hỗ trợ batch và streaming compution trên Window API | Không hỗ trợ |
| Api định nghĩa Topology(các bộ phận cấu thành) | Bắt chước FlumeJava, vận dụng các bộ dữ liệu logic  Xây dựng bộ dữ liệu thật sử dụng  Parallel operation  Vì sử dụng flumeJava, Dataflow có các chức năng tương tự query trong database, hỗ trợ batch và streaming rất tốt | Xây dựng computing network trên nền sprout và bolt  Không có khái niệm rõ ràng về logic dataset, hay parallel operation |
| Nhất quán |  |  |
| implement | Ít rắc rối, không phải set up cluster, monitoring system, chỉ đơn giản là submit pi, peline lên cloud API, rùi gọi thực thi | Thủ công |

1. Use Case sử dụng

**Twitter**

* Storm được sử dụng trong nhiều hệ thống của Twitter như: phân tích thời gian thực, cá nhân hóa, tìm kiếm, tối ưu hóa tổng thu nhập, … Storm đồng hóa với các cơ sở hạ tầng khác của Twitter bao gồm hệ thống CSDL như Cassandra, Memcached, hệ thống giám sát và cảnh báo.
* Lịch cô lập của Storm cho phép nó tận dụng chung một cluster cho các ứng dụng thuộc dạng sản xuất hoặc đang phát triển.

**Yahoo!**

* Yahoo! hiện đang làm việc trên platform cho phép hợp nhất giữa Big Data và xử lý độ trễ thấp. Mặc dù Hadoop là công nghệ chính dùng để xử lý hàng loạt, Storm cho phép xử lý theo luồng các sự kiện của người dùng, nhật ký.

**Infochimps**

* Infochimps sử dụng Apache Storm như là nguồn cho một trong ba dịch vụ dữ liệu đám mây của mình - DDS (Data Delivery Services). Tập dữ liệu thu được bằng cách dùng Storm có khả năng chịu lỗi và mở rộng cao. Giống với Hadoop trong việc thực hiện quá trình ETL và xử lý phân tích hàng loạt, DDS còn hỗ trợ thực hiện chúng theo thời gian thực.

**Flipboard**

* FiIpboard là một ứng dụng tổng hợp những bài báo, blog hay những tin tức. Nó sử dụng Storm trong các dịch vụ như tìm kiếm, phân tích theo thời gian thực, tùy chỉnh nguồn các tạp chí, … Storm được tích hợp với cơ sở hạ tầng bao gồm các hệ thống như ElasticSearch, Hadoop, HBase và HDFS để tạo ra các nền tảng dữ liệu mà có khả năng mở rộng cao.

**Ooyala**

* Ooyala là một công ty tư nhân với sự hỗ trợ của nhiều doanh ngiệp, nó chủ yếu cung cấp các sản phẩm về công nghệ video trực tuyến và Ooyala-power player trên toàn thế giới. Ooyala sử dụng Storm để thực hiện phân tích trực tuyến hành vi của người xem. Storm cho phép khai thác tập dữ liệu video trực tuyến một cách nhanh chóng để cung cấp các phân tích kinh doanh như các mẫu view theo thời gian thực, cá nhân hóa những nội dung đề nghị, …

**Taobao**

* Dùng Storm đễ tạo ra các số liệu thống kê về log và trích xuất các thông tin hữu ích từ các số liệu theo thời gian thực. Các log được đọc từ các thông điệp sẽ được đưa vào hàng đợi trong các spout, sau đó được xử lý và truyền thông qua các cấu trúc liên kết để tính toán các kết quả cần thiết. Số lượng input log của Taobao nằm từ 2 triệu đến 1.5 tỷ mỗi ngày.

**Klout**

* Klout là một ứng dụng sử dụng phân tích truyền thông cộng đồng để xếp hạng người dùng của nó. Xếp hạng sẽ dựa trên mức ảnh hưởng xã hội trực tuyến thông qua “Klout Score” là con số từ 1 đến 100. Storm được sử dụng để tạo ra các cấu trúc liên kết phức tạp để truyền dữ liệu từ các mạng thông qua Kafka, sau đó được xử lý và ghi xuống HDFS.

**Wego**

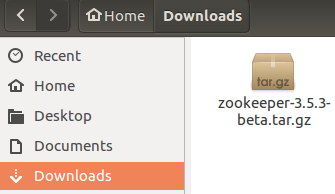
* Là một công cụ tìm kiếm về du lịch và được sử dụng bởi nhiều người. Weigo so sánh và hiển thị lịch các chuyến bay theo thời gian thực, các khách sạn hiện có, giá cả và các địa điểm du lịch trên thế giới. Storm truyền các dữ liệu tìm kiếm thời gian thực từ các nguồn trên khắp thế giới đến các người dùng cuối. Khái niệm về cấu trúc liên kết trong Storm giải quyết nhiều vấn đề và cùng lúc có thể thực hiện các công việc như tích hợp, phân tích và làm sạch dữ liệu.

**NaviSite**

* Sử dụng Storm để giám sát các bản ghi sự kiện (Event logs) của hệ thống. Mỗi bản ghi khi được tạo đều phải thông qua Storm. Storm sẽ kiểm tra các thông điệp với các biểu thức đã được cấu hình, nếu có sự trùng khớp thì thông điệp sẽ được lưu xuống CSDL.

1. Demo
2. **Hướng dẫn cài đặt:**
3. **Cài zookeeper:**

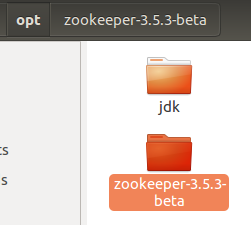
* B1: tải **zookeeper**



* B2: đi đến thư mục **/opt** bằng lệnh **cd /opt**
* B3: extract file tar.gz bằng lệnh: **sudo tar xvf /Đường/Dẫn/Đến/File/apache-storm-vừa-mới-tải.tar.gz**



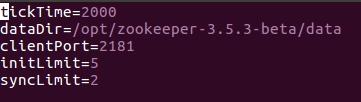
* Sau khi **extract** sẽ được thư mục như vậy:



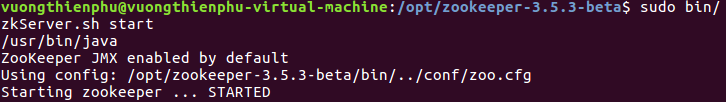
* B4: Đi vào thư mục mới extract và tạo thư mục data bằng 2 lệnh:
* **cd zookeeper-thư-mục-được-extract**
* **sudo mkdir data**



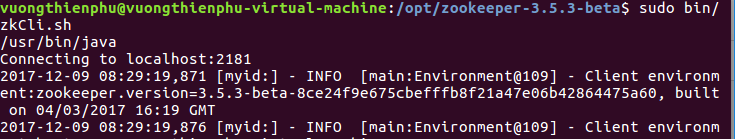
* B5: Tạo file **zoo.cfg** bằng dòng lệnh: **sudo vi conf/zoo.cfg**
* B6: Nhập nội dung file sau đó lưu lại



* B7: Start zookeeper server bằng lệnh: **bin/zkServer.sh start**

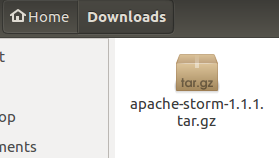


* B8: Start CLI bằng lệnh: **bin/zkCLI.sh**



1. **Cài apache-storm:**

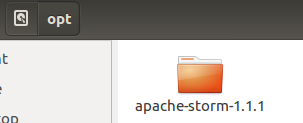
* B1: tải **apache-storm**



* B2: đi đến thư mục **/opt** bằng lệnh **cd /opt**
* B3: extract file tar.gz bằng lệnh: **sudo tar xvf /Đường/Dẫn/Đến/File/apache-storm-vừa-mới-tải.tar.gz**



* Sau khi **extract** sẽ được thư mục như vậy:



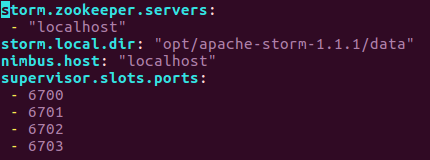
* B4: Đi vào thư mục mới extract và tạo thư mục data bằng 2 lệnh:
* **cd apache-storm-thư-mục-được-extract**
* **sudo mkdir data**



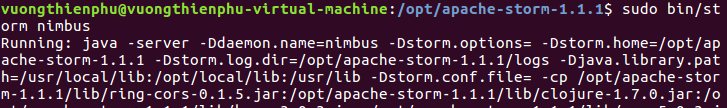
* B5: Tạo file storm.yaml bằng dòng lệnh: **sudo vi conf/storm.yaml**

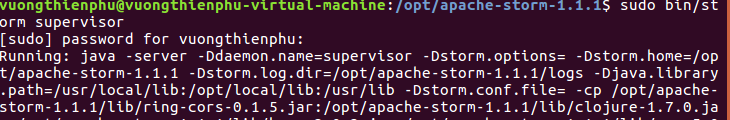


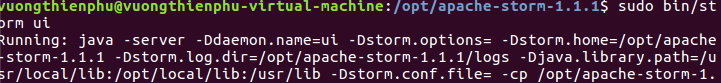
* B6: Nhập nội dung file sau đó lưu lại



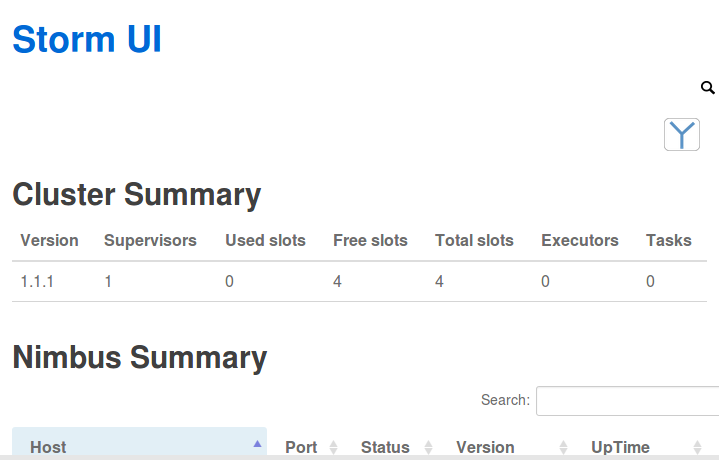
* B6: khởi chạy các dịch vụ của storm
* **bin/storm nimbus**
* **bin/storm supervisor**
* **bin/storm ui**

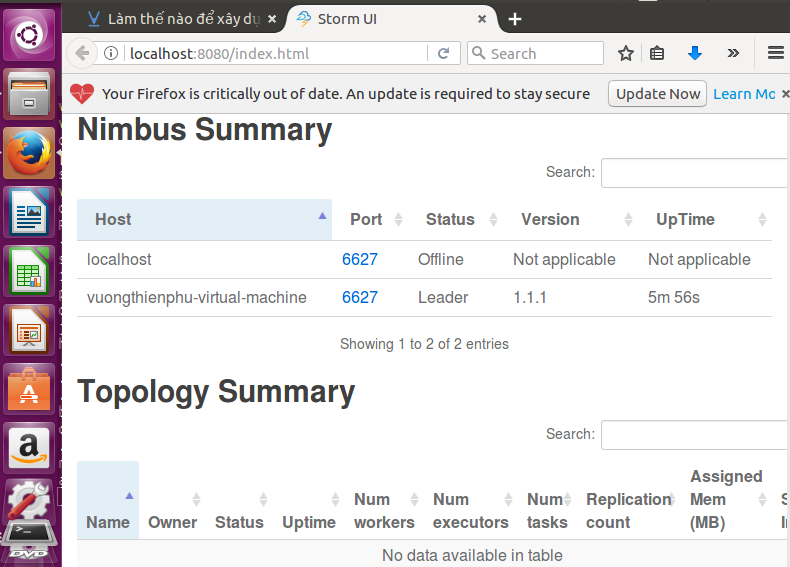






* Kết quả:





1. **Tài liệu tham khảo:**

* <https://viblo.asia/p/lam-the-nao-de-xay-dung-mot-pipeline-phan-tich-xu-ly-big-data-part-2-aRBeXnXBvWE>
* <https://www.edureka.co/blog/apache-storm-use-cases/>
* <https://www.tutorialspoint.com/apache_storm/>
* <https://zdatainc.com/2014/09/apache-storm-apache-spark/>