Roadmap

- Introduction
- Motivations
- Architecture
- Community
Introduction

Apache HTrace is a tracing framework for distributed systems. Currently in incubation.
HTrace Goals

- To monitor system performance in production.
- To diagnose performance issues, node failures, and hardware problems.
- To help developers identify bottlenecks.
HTrace Concepts

- Trace Span
  - A labelled length of time. Has a start time and end time, a unique ID, and a description.

```json
{
  "s": "092d6961d7e7a5a2",
  "b": 1424813328586,
  "e": 1424813328595,
  "d": "ClientNamenodeProtocol#getListing",
  "i": "51fbdaf67e364d18",
  "p": [
    "9840b24cedd01fcc"
  ],
  "r": "FsShell"
}
```
HTrace Concepts

- Span Receiver
  - A library that handles spans generated by an application.
  - Several different span receivers are available...
Big Idea #1

- Follow a single request across the entire cluster.
  - Get timing and performance information back from each node that helped to handle the request
  - Create trace spans for each bit of work.
  - Trace spans can have “parent spans”
Example Trace Span Graph

newDFSInputStream

getBlockLocations

ClientNamenodeProtocol#getBlockLocations
Big Idea #2

- Sampling
  - Sample a small percentage of all requests made. Less than 1% usually.
  - Avoid the overhead of sampling every request, but still get a good idea of where cluster resources are going.
  - Can run HTrace in production, not just on a test cluster. Find performance bottlenecks as they arise.
Motivations for building HTrace

- Diagnosing performance in distributed systems is **hard**!
  - Often difficult to reproduce
  - Can be caused by a flaky network switch, heavy traffic on a particular day, a bug, or the phase of the moon.
Motivations for building HTrace

- Need to break down silos
  - Easy to check metrics for HDFS, HBase, and Hive.
  - Hard to figure out why your Hive query is slow.
  - It is difficult to correlate 100 different log files from 100 nodes!
  - We’ve tried it
Pluggable Architecture

- Two main parts
  - Clients
  - SpanReceivers
- Clients create spans
- SpanReceivers handle them
HTrace Architecture

Java Client

C Client

spans

HTracedRESTReceiver

LocalFileSpanReceiver

FlumeSpanReceiver

and other span receivers...
Configuring Span Receivers

- Receivers are decoupled from the client.
- Can configure Hadoop to use any HTrace span receiver you want.
- Set `hadoop.htrace.spanreceiver.classes` to the class name(s).
- For HBase, use `hbase.htrace.spanreceiver.classes`
LocalFileSpanReceiver

- Writes spans to a local file in JSON format
- A very basic span receiver
- Useful for debugging HTrace.
- Not that useful in production.
HTracedRESTReceiver

- Sends spans asynchronously to the htraced daemon
- Uses a REST interface
- More about that in a bit...
FlumeSpanReceiver

- Sends spans to an Apache Flume endpoint.
- Useful for moving spans between clusters.
The htraced daemon

- A central point to gather span data
- Written in Go
htraced

- Receives spans via a REST interface.
- Stores spans in several LevelDB instances
  - A write-optimized datastore
  - Can take advantage of multiple disk drives
- Exposes a web interface.
htrace command

- Can query the htraced daemon.
- More information via --help

usage: ./build/htrace [<flags>] <command> [<flags>] [<args> ...]

The Apache HTrace command-line tool. This tool retrieves and modifies settings and other data on a running htraced daemon.

If we find an htraced-conf.xml configuration file in the list of directories specified in HTRACED_CONF_DIR, we will use that configuration; otherwise, the defaults will be used.

Flags:
- --help       Show help.
- --Dmy.key="my.value"
  Set configuration key 'my.key' to 'my.value'. Replace 'my.key' with any key you want to set.
- --addr=ADDR  Server address.
...

...
htrace command

- Can get server info
- Can load spans into htraced from a file
- Can dump the contents of htraced into a file
- Can generate a .dot file from a file containing span JSON strings
  - This can then be used to generate a JPG via graphviz
Dumping the contents of HTraced

cmccabe@keter:~/src/htrace/htrace-core/src/go> ./build/htrace dumpAll

{"s":"092d6961d7e7a5a2","b":1424813328586,"e":1424813328595,"d":"
    ClientNamenodeProtocol#getListing","i":"51fbdaf67e364d18","p":[
        "9840b24cedd01fcc"],"r":"FsShell"}

{"s":"3f48698cf024f40b","b":1424813328325,"e":1424813328522,"d":"
    ClientNamenodeProtocol#getFileInfo","i":"9c2ff557d606c968","p":[
        "d9be93a8cf076e97"],"r":"FsShell","t":[
        {"t":1424813328485,"m":"IPC client connecting to a2402.halxg.cloudera.
        com/10.20.212.10:8020"},
        {"t":1424813328506,"m":"IPC client connected to a2402.halxg.
        cloudera.com/10.20.212.10:8020"}]

...
Finding a Span in HTraced

cmccabe@keter:~/src/htrace/htrace-core/src/go> ./build/htrace findSpan 0x3f48698cf024f40b
{
    "s": "3f48698cf024f40b",
    "b": 1424813328325,
    "e": 1424813328522,
    "d": "ClientNamenodeProtocol#getFileInfo",
    "i": "9c2ff557d606c968",
    "p": [
        "d9be93a8cf076e97"
    ],
    "r": "FsShell",
...

htraced web UI

- A graphical web interface for htraced

<table>
<thead>
<tr>
<th>ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>092d6961d7e7a5a2</td>
<td>ClientNamenodeProtocol#getListing</td>
</tr>
<tr>
<td>3f48698cf024f40b</td>
<td>ClientNamenodeProtocol#getFileInfo</td>
</tr>
<tr>
<td>9840b24ced01fcc</td>
<td>listPaths</td>
</tr>
<tr>
<td>d9be93a8cf076e97</td>
<td>getFileInfo</td>
</tr>
</tbody>
</table>
htraced web UI planned features

- “Search” screen to search for spans by description, time, duration, etc.
- “Span Details” screen to view detailed information about a trace span, including a graph of its parents and descendents
- “Histogram” screen to show statistics
Community

● Very active community
● Many mailing list messages every day
● Integrated into HDFS, Hadoop, HBase, Accumulo, and others
Hadoop with HTrace

- HTrace has been integrated into HDFS
  - The main work remaining is the HDFS write path
- No stable release with Apache HTrace yet (Hadoop 2.6 used the pre-apache version of HTrace)
- The next Hadoop release (Hadoop 2.7) will include support for the Apache version of HTrace.
HBase with HTrace

- HTrace has been integrated into HBase
- HBase 1.0.0 uses the Apache 3.1.0 release