Chukwa Collector Setup Guide

Table of contents

1. Basic Operation

Chukwa Collectors are responsible for accepting incoming data from Agents, and storing the data. Most commonly, collectors simply write all received to HDFS. In this mode, the filesystem to write to is determined by the option `writer.hdfs.filesystem` in `chukwa-collector-conf.xml`. This is the only option that you really need to specify to get a working collector.

By default, collectors listen on port 8080. This can be configured in `chukwa-collector.conf.xml`

2. Configuration Knobs

There's a bunch more "standard" knobs worth knowing about. These are mostly documented in `chukwa-collector-conf.xml`

It's also possible to do limited configuration on the command line. This is primarily intended for debugging. You can say 'writer=pretend' to get the collector to print incoming chunks on standard out, or portno=xyz to override the default port number.

```
bin/jettyCollector.sh writer=pretend portno=8081
```

3. Advanced options

There are some advanced options, not necessarily documented in the collector conf file, that are helpful in using Chukwa in nonstandard ways.

While normally Chukwa writes sequence files to HDFS, it's possible to specify an alternate Writer class. The option `chukwaCollector.writerClass` specifies a Java class to instantiate and use as a writer. See the `ChukwaWriter` javadoc for details.

One particularly useful Writer class is `PipelineStageWriter`, which lets you string together a series of `PipelineableWriters` for pre-processing or post-processing incoming data. As an example, the SocketTeeWriter class allows other programs to get incoming chunks fed to them over a socket by the collector.

Stages in the pipeline should be listed, comma-separated, in option `chukwaCollector.pipeline`

```
<property>
  <name>chukwaCollector.writerClass</name>
<value>org.apache.hadoop.chukwa.datacollection.writer.PipelineStageWriter</value>
```

```
</property>

<property>
  <name>chukwaCollector.pipeline</name>
<value>org.apache.hadoop.chukwa.datacollection.writer.SocketTeeWriter,org.apache.hadoop
</property>
```

3.1. SocketTeeWriter

The `SocketTeeWriter` allows external processes to watch the stream of chunks passing through the collector. This allows certain kinds of real-time monitoring to be done on-top of Chukwa.

SocketTeeWriter listens on a port (specified by conf option `chukwaCollector.tee.port`, defaulting to 9094.) Applications that want Chunks should connect to that port, and issue a command of the form `RAW|WRITABLE <filter>\n`. Filters use the same syntax as the Dump command. If the filter is accepted, the Writer will respond `OK\n`.

Subsequently, Chunks matching the filter will be serialized and sent back over the socket. Specifying "WRITABLE" will cause the chunks to be written using Hadoop's Writable serialization framework. "RAW" will send the internal data of the Chunk, without any metadata, prefixed by its length encoded as a 32-bit int, big-endian. "HEADER" is similar to "RAW", but with a one-line header in front of the content. Header format is `hostname datatype stream name offset`, separated by spaces.

The filter will be inactivated when the socket is closed.

```
Socket s2 = new Socket("host", SocketTeeWriter.DEFAULT_PORT);
s2.getOutputStream().write("RAW datatype=XTrace\n".getBytes());
dis = new DataInputStream(s2.getInputStream());
dis.readFully(new byte[3]); //read "OK\n"
while(true) {
   int len = dis.readInt();
   byte[] data = new byte[len];
   dis.readFully(data);
   DoSomethingUsing(data);
}
```