MonitoringNutchCrawls

Monitoring Nutch Crawls

So you got Nutch all configured and turned it loose on your site, but your itchy trigger finger just needs to know how well it's working? Here are a couple ways you can keep an eye on your crawl:

Monitoring network traffic

One way is to watch Nutch suck up your bandwidth as it crawls its way around. If you look at a graph of historical bandwidth usage, you should see it spike up and stay at a fairly consistent plateau, with valleys every so often each time a segment completes (since while Nutch is merging segments it doesn't use any bandwidth).

Some tools for this:

- ntop (Linux, Windows) - A nifty program that gives you a Web-based history of your machine's bandwidth usage. You might get lucky and have it install easily... because the website isn't terribly helpful for install help.

Monitoring fetch statistics

Of course, the bandwidth alone doesn't tell the whole story. How many pages are you retrieving? How many failed?

Here's a quick little shell script to do this; I'm sure people can improve on this--edit this page if so!

```
#!/bin/sh
echo "Monitoring nohup.out crawl progress..."
while :
do
  echo "Tried `grep 'fetching' nohup.out | wc -l` pages; `grep 'failed' nohup.out | wc -l` failed."
  sleep 60
done
```

To run this script:

1. Save this script as something like `monitorCrawl.sh` 2. Run your preferred crawl script with nohup, like this: `nohup <nutch crawl command or script> &` 3. By default, this will output to nohup.out in the working directory. From the same directory, run: `sh monitorCrawl.sh`

(Alternately, you can process hadoop.log in the logs/ directory by changing the three references to `nohup.out` to `hadoop.log`. Be aware, though, that by default hadoop.log only contains activity from today, so your counts will reset to zero each night.)

This will give you minute-by-minute stats on how many pages nutch tried to fetch and how many failed with errors (e.g. 404, server unreachable).

A More expansion of above Script:

Run this script by changing the three export paths in the script...

```
#!/bin/bash
###############################################################################################
##########################Author: Chalavadi Suman Kumar ##############################
##########################Email: sumankumar4@gmail.com ##############################
###############################################################################################

# Usage: sh monitor.sh existing/running logfile outputlogfile errorlogfile
# 'existing' - logfile is not a running log (already existing log)
# 'running'  - logfile is a running log (this is the default option)


#Edit your Log directory names...
export LOGFILE=/home/suman/nutch-0.8/logs/hadoop.log
export SAVEFILE=/home/suman/NutchMonitor/short_hadoop.log
export ERRORFILE=/home/suman/NutchMonitor/logs/hadoop_error.log

#Specify for reading the existing log or running log
#By default it assumes 'running'
case "$1" in
```

```
'existing') COMMAND="cat" ;;

'running') COMMAND="tail -f" ;;

*) COMMAND="tail -f" ;;
esac

#Change the LOGFILE,SAVEFILE and ERRORFILE, if provided through command line
#IF '-' is given, take the default values..
if [[ "$2" != "" && "$2" != "-" ]]; then LOGFILE=$2; fi
if [[ "$3" != "" && "$3" != "-" ]]; then SAVEFILE=$3; fi
if [[ "$4" != "" && "$4" != "-" ]]; then ERRORFILE=$4; fi

#Initializing the Variables....
minute=0
fetchcount=0
fetchfailcount=0
indexcount=0
mflcount=0
lasttime=0
newtime=0
totalfetchcount=0
totalindexedcount=0
totalfetchfail=0
totalmflcount=0

#Reads the appended content of the file as the file grows. So each log is processed through the while loop...
$COMMAND $LOGFILE | while read some; do

#to aviod exceptions. They some time start with 'java' and 'at'.
case $some in
[0-9]*)
#Get the time(current minute) in 'newtime' and entire date in 'totaltime'
newtime=`echo $some|perl -ne '@words=split(/\s+/);@temp=split(/:/,$words[1]);print $temp[1]'`
#totaltime=`echo $some|awk '{print $1" "$2}'`
totaltime=`echo $some|perl -ne '@word=split(/\s+/);print "$word[0] $word[1]"'`
;;
*) echo $some >>$ERRORFILE;;
esac

#echo $temp $newtime $totaltime $lasttime

#Pattern matches for important operations and prints with the change in time...
case "$some" in
*indexer.Indexer\ -\ Indexer:\ starting*)
                indexcount=0;mflcount=0;
                ;;

*indexer.Indexer\ -\ Indexer:\ done*)
                if [[ $indexcount -ne 0 || $mlfcount -ne 0 ]]; then
                echo "$totaltime Indexed:$indexcount MaxFieldLengthError:$mflcount
TotalPagesIndexed=$totalindexedcount TotalMFLErrors=$totalmflcount" >>$SAVEFILE
                lasttime="$newtime";
                fi
                indexcount=0;mflcount=0;
                ;;

*fetcher.Fetcher\ -\ Fetcher:\ starting*)
                fetchcount=0;fetchfailcount=0;
                ;;

*fetcher.Fetcher\ -\ Fetcher:\ done*)
                if [[ $fetchcount -ne 0 || $fetchfailcount -ne 0 ]]; then
                echo "$totaltime FetchTried:$fetchcount FetchFailed:$fetchfailcount
TotalPagesFetched=$totalfetchcount TotalFetchesFailed=$totalfetchfail" >>$SAVEFILE
                lasttime="$newtime";
                fi
                fetchcount=0;fetchfailcount=0;
                ;;

*fetching*)
```

```
                let fetchcount=$fetchcount+1;
                let totalfetchcount=$totalfetchcount+1;
                if [ "$newtime" -ne "$lasttime" ]; then
                echo "$totaltime FetchTried:$fetchcount FetchFailed:$fetchfailcount
TotalPagesFetched=$totalfetchcount TotalFetchesFailed=$totalfetchfail" >>$SAVEFILE
                fetchcount=0;fetchfailcount=0;
                lasttime="$newtime";
                fi
                ;;

*failed*)
                let fetchfailcount=$fetchfailcount+1;
                let totalfetchfail=$totalfetchfail+1;
                if [ "$newtime" -ne "$lasttime" ]; then
                echo "$totaltime FetchTried:$fetchcount FetchFailed:$fetchfailcount
TotalPagesFetched=$totalfetchcount TotalFetchesFailed=$totalfetchfail" >>$SAVEFILE
                fetchcount=0;fetchfailcount=0;
                lasttime="$newtime";
                fi
                ;;

*Indexing\ Filter*);;
*IndexingFilter*);;

*Indexing*)
                let indexcount=$indexcount+1;
                let totalindexedcount=$totalindexedcount+1;
                if [ "$newtime" -ne "$lasttime" ]; then
                echo "$totaltime Indexed:$indexcount MaxFieldLengthError:$mflcount
TotalPagesIndexed=$totalindexedcount TotalMFLErrors=$totalmflcount" >>$SAVEFILE
                indexcount=0;mflcount=0;
                lasttime="$newtime";
                fi
                ;;

*maxFieldLength*)
                let mflcount=$mflcount+1;
                let totalmflcount=$totalmflcount+1;
                if [ "$newtime" -ne "$lasttime" ]; then
                echo "$totaltime Indexed:$indexcount MaxFieldLengthError:$mflcount
TotalPagesIndexed=$totalindexedcount TotalMFLErrors=$totalmflcount" >>$SAVEFILE
                indexcount=0;mflcount=0;
                lasttime="$newtime";
                fi
                ;;
esac

#lasttime="$newtime";

#print the important operations(stages of crawling).
echo $some | perl -ne 'if(/fetcher.Fetcher\ -\ Fetcher:|indexer.Indexer\ -\ Indexer:|crawl.CrawlDb\ -\ CrawlDb
update:|crawl.Generator\ -\ Generator:|crawl.Injector\ -\ Injector:|crawl.LinkDb\ -\ LinkDb:|indexer.
DeleteDuplicates\ -\ Dedup:|indexer.IndexMerger|crawl.Crawl\ -\ crawl/g){print}' >>$SAVEFILE

echo $some | perl -ne 'if(/ERROR|WARN|FATAL/g){print}' >>$ERRORFILE

done
```

Running this script:

1. Change the exported paths(LOGFILE,SAVEFILE,ERRORFILE) in the script according to your nutch location. 2. Run the script. 3. Monitor your crawl by the following command: tail -f short_hadoop.log

This will give you minute-by-minute stats on how many pages nutch tried to fetch and how many failed with errors, how many pages indexed, how many pages truncated due to maximum field length(MLF) error during indexing. The errors are saved to another file.