This document describes the procedures used to receive and prepare the data gathered by HarvardX courses that are hosted on the edX platform. This all presumes that all of the python scripts in the directories

Tools/src/main/python

Tools/src/main/python/convertfiles

Tools/src/main/python/logs

are in the path of the shell, along with

Tools/shellscripts

The python scripts have been written against python 2.7.1, but seem to work fine with python 2.6. The shell scripts assume the tcsh (or csh).

**Receipt of Data**

All of the data is received via a posting on Amazon's S3 service. There are two sets of data. The first are the log files, gathered from the edX production servers and containing information about the Harvard course interactions that have been recoreded on each of the servers. The second set of data is extracted from the database that backs the courses themselves, and contains student interaction and demographic information. Both sets of data are encrypted using a key supplied by Harvard. The log files are then gathered together into a single tar file, while the course data is gathered together into a single .zip file. These files are to be posted weekly, generally on Sunday.

**Course Data files**

Upon unzip, the files for all of the courses are contained in a single directory. The files are named with a prefix

HarvardX-CourseNumber

but are otherwise not segregated by course. All of the files are encypted. Each course has an extraction of all of the interactions students have had on the course discussion forum in a file with the extension .mongo. All of the other files come in two forms. The first is a tab-separated .sql form, the second is an xml formatted file. These files have the same content. There are also files that contain information about wiki articles from each course. To save time, the xml-formatted versions of the data files and the wiki-article files are removed. At that point, the script uncompAndDecrypt.sh can be run, which will decrypt all of the remaining files.

The shell script separateFiles.sh will create a separate directory for each of the courses, named for the course, and will then move all of the data files for a particular course into that directory. Sections of the HLS1x course are treated as different courses by this script. Note that the courses for which the data is received is hard-coded in the separateFiles.sh script. When new courses are added, or if data is no longer being received for a course, this script needs to be updated. If used by a different institution, the script will need to be re-written.

At this point, the shell script renameInDirectory.sh should be run in each of the course directories; this will rename the much longer names the files have at this point to a more easily typed and remembered set of names. In each directory, the result will be

users.sql : Basic information about all of the students signed up for the course

profiles.sql : Self-reported demographic information for each of the users in the course

enrollment.sql : A listing of when each of the students enrolled in the course

certificates.sql : A tracking of the certificate state for each of the students. This is only interesting when the course ends

studentmodule.sql : All the courseware state for the students

forum.mongo : A record of the forum activity for the course

A more complete description of these files can be found in the edXDataDesc.pdf file.

Since most of the researchers prefer .csv files as input, a python script, sqltocsv.py, is then run over all of the various directories and all of the files to convert the .sql file into a .csv with the same content.

All of this processing has been automated in the shell script processClassData.sh.

**Log files**

The log files are downloaded as a single tar file; on running the command

tar xf downloadFile

the result is a single directory, named HarvardX, that contains a set of directories, one for each server on which HarvardX courses are or have been running. In each of those directories will be a file for each day that a HarvardX course ran on the server. All of the files are encrypted.

Step one is to remove files that have already been processed. This can be done by specifying the date of the first file in the latest log dump in the format YYYY-MM-DD that you want to keep as the first argument to the program cullLogFiles.py in each directory. This will cut down significantly on the set of logs that need to be processed. It is good to run rmdir on all of the directories; this will only remove the empty ones but will save effort in what follows.

The second step is to decrypt all of the files, using the uncompAndDecrypt.sh script run interatively over all of the production server directories. Once this is done, run separateAllByClass.py in each directory, which will create a set of log files for each class for that production server.

At this point, we can run mergeClassLogs.sh, which creates a directory for each class and moves the logs that were created in each of the production directories to the directory for the class. When this is done, it is a good idea to go into each directory and remove any empty logs, and then to remove any empty directories.

Once this is done, we can go into each of the remaining class directories and combine the logs by running buildWeekLog.py. This will create a single log file for the class out of all of the logs for the class on the different servers, with the log entries ordered by timestamp. While it is the intention of this program to build a complete log that starts with the new entries since the last set of logs were downloaded and processed, it depends on the use of cullLogFiles.py to do this; if you did not cull any log files (or did not cull them correctly) more than a single week of log files could be included in the combined log file.

All of these steps have been automated in the script processLogData.sh, which takes a single argument which should be the date, in YYYY-MM-DD format, for the first date for which log files should be included. If there is no date given, the script will not cull any of the log files and a complete log file for each course will be constructed.