Jeremy DeHay

Security Scripting w/ Python

Week 7 – June 23, 2017

**INTRODUCTION:**

I currently run a website, [http://mixed-messages.org](http://mixed-messages.org/), that allows users to upload video clips of almost any format up to 100MB in size. These clips are stored on Amazon S3 and must be reviewed by myself or my colleagues before they are publicly visible on the website. All browsers now support \*.MP4 file types, but not all submissions are encoded in \*.MP4. The problem that I aim to solve is to take the uploads, use a service to transcode them, and host those files to be streamed automagically.

**SOLUTION OVERVIEW:**

To solve this problem I intend to use the Flowplayer service and API to automate the login and job creation process for transcoding these accepted uploads. This will eventually be implemented into my website for administrators to simply click the “Accept” button on the upload and have it available to the public viewable in all web browsers.

**DATA NEEDS:**

The data will be whatever type of video was initially uploaded and the \*.mp4 format that will be hosted on Flowplayer. I also need to formulate JSON requests to handle the logging in function in Flowplayer, which will include strings.

**DATA AND DATA TYPES:**

The majority of commands will be through the Flowplayer API. Therefore, I need to know the source of the files I wish to transcode as well as the final destination of the transcoded files. In the interim, I need to log in to Flowplayer and create the ‘jobs’ that will direct Flowplayer to perform the transcoding. Because of these functions, and the fact that I don’t have to deal with the files directly, the data types I will be manipulating are Strings and JSON. The functions I expect to use are as follows:

* Amazon S3 login
* Get a list of videos available on S3
* Print the list of videos available on S3
* Flowplayer login
* File prep for job creation (this will call the next function)
* Flowplayer job creation
* Pull the location for the source and destination of the files
* Display the code allowing admins to publish the transcoded video.

**NETWORK FUNCTIONALITY:**

The first thing the program does is connect to AWS S3 and see what videos are available to transcode. Using the Boto3 library, S3 validates by checking credentials and configuration files that I had to create that contain information such as my access key and secret key. It then uses regular expressions to correctly format the list and there’s an option to display it. If the user chooses to transcode a video, he must type in the exact name of the video. Once that happens, the string is modified to include the entire path to that video and the string is sent to a function that sends it as an HTTP request, along with some generated JSON data, to Flowplayer Drive where it becomes transcoded. Before the video can be transcoded, however, the program needs to login to Flowplayer. This is done with another HTTP request, to which the program must handle the response via JSON. Out of that response, the program must note the authorization code and that authorization code must be sent with any API calls as header data. The final bit of data mangling comes from the video title, as the program replaces spaces with plus signs. Then it splits the filename at the extension so when it’s transcoded it has the proper extension.

**CONCLUSION:**

While this functionality will not be implemented on my website right away, the basic functions of this program will work from the command line by the end of the project. The goal will be to simulate the functionality of the website where a user can upload a file and “accept” it, which will then push the file through the transcoding process. From there it might direct you to the link to show the transcoded files or it might open up the default browser on your device to play the video. Due to the sensitive nature of using third party services that require log ins, I will be using free/trial versions for this course.