Scripting Challenge

The following six (6) scripting questions are used as a diagnostic tool to assess applicant skills. Scripting is not a primary goal of positions where this challenge is given, but applicants are expected to provide a best effort to answering these questions. Answers need to be submitted one (1) week after receiving this scripting challenge.

Applicant should assume the target platform is **Linux**. Answers for Windows or Mac may be accepted in limited circumstances if pre-approved. Script submission will consist of a tar/tgz/zip file containing a script file for each question and an optional README if necessary. The files can be submitted with renamed extensions to overcome spam filters. Accepted scripting languages are **bash**/**ksh**, **perl**, **ruby**, and/or **python** for questions 1 and 2, 4, 5, and 6. Question 3 needs to be in bash/ksh. If pre-approved to submit script answers for Windows, only **Powershell** is allowed.

At a minimum, two (2) answers need to be submitted for consideration, and candidates are encouraged to complete more that two.

- Write a script that makes a REST call to http://time.jsontest.com and uses the result to compare against the localhost's time/date setting. Script result should note the systems' discrepancy between the times in seconds.

- A log file in a compressed archive (`script_log.tgz`) is located on the Aspera demo server at https://shares.asperasoft.com under the **Scripting Challenge** "share" when logged in as user **scripting** (with password **test_files_for_2014**). To download this content you will need to download and install the *free* Aspera Connect plugin. Once the log file is downloaded, parse this log file. The script will report:

    - Total number of `ascp` transfers;
    - Listing of all transfers and their average rate in Mb/s;
    - PID of transfer(s) with the longest duration;
    - PID of the transfer(s) sending the most data;
    - PID of the transfer(s) with the fastest rate.

    The log file is a typical syslog `messages` log file. The `ascp` logs a lot of data in this file, but there is a set of metrics that are helpful for this. The lines are of the form:

    ```
    prog t/f/e=14604216/14604216/60189366
    ```

    This means:

    - ❑ `t` (running total of bytes received) == `14604216`
    - ❑ `f` (running total of file bytes received) == `14604216`
    - ❑ `e` (total elapsed time [microseconds]) == `60189366`

- Write a script that can copy a file, without using `cp`. The script will take two command line parameters: source file and destination file. Describe the performance implications of the approach that you took and compare it to other methods of copying a file.

- Devise a method for creating temporary file data for use in testing file transfer. The result of the script should be a directory that may have one or more nested directories (quantity randomly determined). Inside these directories are a random number of files in each directory. The files should be large (greater than 1MB) and mimic a collection of data. Applicant is encouraged to determine the make-up of the collection, but here are a couple examples:

    - a collection of frame images in sequential order
    - an archive of video files organized by title.

  Please note rationale in the collection chosen, and how the files were generated. What optimizations or considerations need to be made.

- Create a script that will SSH into a remote host and install the user's SSH public key. Assume a target host is Linux. Solution does not need to cover every edge case, but you will need to describe pitfalls with your approach, and how you would address those issues.

- Write a script that periodically executes. This script will check disk space utilization of a defined volume/directory that is passed to it. It will write to a configured file a log entry that indicates the expected date the storage will fill up based on the growth pattern of the storage.