Assignment 3 - Writing a Script To Make Your Life Easier

COMP 3532 - System Administration and Maintenance

Usually, as a System Administrator, you will have a great number of multi-step tasks that you need to do on a regular basis. In order to keep your work consistent it is usually better to automate work with a script rather to rely on yourself (or other people) to make sure the work gets finished and done the same way each time.

**Your assignment is to create a script that automates a fairly involved task in a safe, intelligent and efficient way.**

This assignment is intended to give you experience writing a script of a complexity and quality that could be used in a production environment.

**Important:**

Your script **must** operate in a default mode (with no arguments at all) and **must** also have a mode where it takes one or more standard arguments (a filename to read or create, for example) and should have at least three flags which modify its function (for example -v for verbose (talks about what it's generating), -h for human readable output)

# The Password Generator

Sometimes you need to generate passwords for users. You may need to create new accounts for many users at a time, you may have users who need their passwords reset or your users may need help finding good passwords.

The best a password can be, is a long string of random characters, but these are difficult for people to remember and so usually some combination of words with numbers and special characters added to provide security. A popular method for this is the "Horse Battery Staple" approach laid out in XKCD ​<https://xkcd.com/936/>​.

This script will use a variation of this technique to make good passwords. It will concatenate several words, obscure some of the vowels with numbers and add a few special characters. It's usage should be as follows

|  |  |
| --- | --- |
| Command | Output |
| goodpass | assass3nate^Swaggeddr2ids  \*Will generate one password |
| goodpass passfile | \*Will store the generated password to passfile |
| goodpass -n 3 | @Nutmegsol3newscasters perfectlyMo4setraps#fraught  v5ndersAlforddrawing&  \*Will generate ​*n*​ passwords |
| goodpass -w 2 | Affability4nchovy%  \*Will generate a password made out of only w words (2 here) |
| goodpass -w 5 | p0tencysForgotsurfboard&serializeenchantingly \*Will generate a password made out of only w words (5 here) |
| goodpass -s | t3rtn4ssD47nys4s\*C2b7s  \*Will generate a stronger password by converting all vowels |
| goodpass -s -w 2 -n 2 | f0$m8r3C1rt2s0$ #fl8cksc@7tn2ps  \*Flags can be used together, so here two strong passwords of 2 words |

The script produces one password by default. If the script is given a file name then it will save the password(s) to that file. The script can also be given a -n flag to generate a number of passwords, a -w flag to change the number of words used to generate the password and a -s flag that produces a 'stronger' password by replacing all vowels rather than just some of them.

## Hints

The words fill will provide you with a good supply of words and the shuf command can be helpful for choosing them randomly. Bash's string manipulation tools

<https://www.tldp.org/LDP/abs/html/string-manipulation.html>​ will be usefull as well. Also remember that the ​$RANDOM ​variable is your easy source for getting random numbers (although there are ​*many*​ different tools that might help.)

# The Rubric

I will be grading this assignment using the following rubric:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Acceptable - (1 point) | Good - (2 points) | Very Good - (3 points) |
| Functionality  **(Double points)** | Script implements a minimal subset of the intended functions it should. | Script implements a full set of intended functions. | Script implements a full set of intended functions as well as an additional interesting function. |
| Useability | Script gives sufficient error messages and feedback on function to user. | Script is easy to use, with good error messages and feedback. | Script feels in line with standard \*nix utilities. |
| Best Practices | Script is written with limited use of good practices | Script is written with regular use of good practices, with a few notable exceptions | Script is written with good use of good practices |
| Style | Script is difficult to read and understand. | Script is readable and understandable, cites all example code used. | Script is stylish and informative, cites all example code used. |
| Demo | Script is presented in a confusing way. | Script is presented in a way that is mostly understandable but leaves questions. | Script is presented is  an easy to understand way. |