Security in Mobile Systems - UE/EX

FH Hagenberg, WS 2016
DI Dr. Erik Sonnleitner

Exercise 3: Shell Scripting

Exercise 3: Shell Scripting

(1) Write a shell script `backup.sh`, which is capable of backing up data supplied by the user.

- Notes:
  - Script takes one (1) argument, with is a path (relative or absolute) pointing to the directory to be backuped
  - Script has to create a `.tar.bz2` archive from the supplied data
  - Archive must (and must only) include files, that have been modified in the past 24 hours
  - If temporary files or diretories are used, their file-/dir-name must be uniquely random, and get deleted again
  - Archive must be created in the current working directory where the script is called
  - Script must handle recursive backups (i.e. including the original directory structure)
  - Script must include proper error handling (does path exist? was there an error creating the archive? etc)

Exercise 3: Shell Scripting

(2) Write a shell script `proc-control.sh`, which takes 2 mandatory arguments: the first one being either a program name or a PID, and the second one being an integer value N. The script has to continuously check every N seconds if the given process is still running. If so, print a message to **`stdout`**. If not, print a message to **`stderr`** and ask the user if he/she wants to restart the program.

- Notes:
  - If program name is given, the script has to determine the PID itself
  - If program name is given and there are multiple processes with the same name running, list all corresponding PIDs on terminal and force user to choose exactly one
  - If the user chooses to restart the program, it should be started in background (detached from the shell) and the script continues to monitor it
  - All arguments of the originally running program must be maintained when restarted
  - If restarting fails for some reason, exit with error.
  - Script should accept an optional argument `-auto`, which automatically performs the restart if program has exited, and informs the user if that happened
- Note: Echoing to **`stderr`** only is done via `>&2 echo "error"`

Exercise 3: Shell Scripting

(3) Write a shell script `query-user-details.sh` which takes either a username or UID (user ID) as argument. It subsequently parses `/etc/passwd`, locates the line of the given user and extracts all information contained therein, which are then pretty-printed to the terminal via field=value lines.

Moreover, it should print whether the user has a password set or not (active or inactive account). If the user has a home directory, also print the total number of files therein, as well as the total size of the directory. The script output must not contain any error messages of commands which have been used to acquire the needed information.

Exercise 3: Shell Scripting

(4) Write a shell script `cookie-search.sh`, which parses the cookie database of Mozilla Firefox. It should take the following arguments:

- `./cookie-search.sh host <host>` Exact host specification
- `./cookie-search.sh pattern <pattern>` Match pattern for hostname

For the given host/pattern, the script should return the total number of cookies stored, and print all matching cookies line-wise and human-readble in the format `NAME: VALUE` .
Notes:

- Firefox uses **sqlite** to store cookies. Install the `sqlite` package (`sqlite3` on Ubuntu or Mint) if not installed, and consult the manual page for how to query information.
- Cookie database typically located at: `~/.mozilla/firefox/XXXXXXXX.default/cookies.sqlite`
- You don't have to take care of multiple user-profiles (if you use multiple profiles, take the first one or hard-code it)