User management--
a generalized management script example

David Morgan

© David Morgan 2005

Adding users – actions/mechanics

- add record to /etc/passwd
- add record to /etc/shadow
- add record to /etc/group for user’s default group
- add user to pre-existing groups
- create user home directory /home/<username>
- copy default startup files to home directory
- set permissions on new files and directories
- set ownership on new files and directories
- set system password
- set other passwords (e.g., Samba)
- customize user info with, e.g., usermod or chage
- setup mail home/aliases
- set disk quotas

© David Morgan 2005
Process of adding users

- surprisingly extensive, isn’t it!?
- varies among sites (local policies differ)
- no utility does it all
  - some do it partially (useradd, passwd)

Steps performed by useradd

- add record to /etc/passwd
- add record to /etc/shadow
- add record to /etc/group for user’s default group
  - add user to pre-existing groups
- create user home directory /home/<username>
- copy default startup files to home directory
- set permissions on new files and directories
- set ownership on new files and directories
- set system password
- set other passwords (e.g., Samba)
- customize user info with, e.g., usermod or chage
- setup mail home/aliases
- set disk quotas
### Steps performed by passwd

- add record to /etc/passwd
- add record to /etc/shadow
- add record to /etc/group for user’s default group
- add user to pre-existing groups
- create user home directory /home/<username>
- copy default startup files to home directory
- set permissions on new files and directories
- set ownership on new files and directories
- set system password
- set other passwords (e.g., Samba)
- customize user info with, e.g., usermod or chage
- setup mail home/aliases
- set disk quotas

### A common approach -- adding users in 2 steps

- run useradd
- then set password with passwd
Other approaches

- manual - perform individual steps separately
- hybrid - some with utilities, others manually
- automated - all by script(s) you write

“Although the `useradd` and `userdel` commands are convenient, they are usually not sufficient to implement all of a site’s local policies. Don’t hesitate to write your own `adduser` and `rmuser` scripts; most larger sites do. … You homebrew scripts can call the standard utilities to accomplish part of their work.”

Linux Administration Handbook Nemeth, Snyder, and Hein

Where credit is due

Following approach and scripts are from:

Automating UNIX and Linux Administration,
Architecture

Ready! - loop thru these first

main script

Go! - then loop thru these

scripts to gather needed data

scripts to modify system using it

Default names/locations

/usr/local/sbin/adduser
!! and !!
/usr/local/sbin/deluser
(by symlink)

/scripts to gather needed data

/scripts to modify system using it

Default names/locations

Default names/locations

Default names/locations

Default names/locations

Default names/locations
Features of note

- 2-way caller <> callee variable communication
  - shells can’t pass variables back to callers, normally
- extensibility - loops capture any/all scripts provided
  - no code changes, mere placement plugs new callees in

Communication of variables

main script

via export (set -a)

via returned-then-eval’d assignment statements

fullname.sh:

echo "fullname='John Smith"

…which is an assignment statement…

…which eval proceeds to execute in caller main, establishing callee-assigned variable there (by extension available thereafter to “mod” scripts)
Auto-extensible by drop-in

main script

new.sh gets called, no change to main

main:

grab <list of files> in /usr/local/usertool/data/
for file in <list of files> ; do
eval "$file" # execute the output of the file
done

caveat: don’t let stray files in the directory!

note: new programs in any language OK, as long as they output valid shell commands on stdout

Example data flows

fullname.sh → basic.sh → /etc/passwd
shell.sh

extragroups.sh → extragroups.sh → /etc/group

homedir.sh → homedir.sh → /home or …

e.g., fullname.sh and shell.sh in the data directory solicit from user his full name and choice of shell, pass results to basic.sh in the mod directory, which from that and other info composes a standard user record and inscribes it into the “official user database” /etc/passwd

© David Morgan 2005
deluser alternative functionality

- “adduser” callable by alternative name “deluser”
- it checks by which name it was called
- undoes most (not all) of its “adduser” actions when called as “deluser” instead
- undoing remainder can be implemented as a custom add-on script you supply
Custom add-ons

- write additional programs in any language
- “implementation-by-placement”
  - data-gatherers
    - must print valid shell commands to stdout, any screen messages to stderr
    - drop into /usr/local/lib/usertool/data/
  - system-modifiers
    - no output restrictions
    - drop into /usr/local/lib/usertool/mod/
  - auto-called on next run
  - avoid stray files in script directories
- execution order is alphabetical within directory, name accordingly