Tricks of the Trade

Hands-On Unix System Administration DeCal

Lab 7 — 7 March 2011

Due 14 March at 6:10 PM

1 Group exercises

1. **[SSH keys]** We'll be giving you access to more powerful VMs for your final projects shortly (they live on `blizzard.ocf` instead of `decal.ocf`), but instead of handing out passwords, we'll be using SSH keys. Generate an SSH public/private keypair as per the instructions in today's lecture, securely distribute the private key among everyone in your group, and email your `id_rsa.pub` file to `jordan.salter at berkeley.edu`. (Put "Sysadmin DeCal" and your group number in the subject line of your email.)

2. **[Final project]** The end is nigh! Well, if you believe the people at Northgate and Sproul Plaza, anyway — they're warning that the world will come to an end one week after final exams. More relevantly for this class, though, we're halfway through the semester and it's time to start brainstorming for project ideas. As a group, come up with 3–4 ideas for a project, with a few notes about how you might implement each one. Submit your ideas as a group.

2 Individual exercises

You may use `apt-get` to install RCS and Mutt for this exercise.

1. **[RCS]** Use your Google-fu to answer the following questions. You might also find ci(1), co(1), rcs(1), and rcsintro(1) instructive.

   - How do you put a file under version control with RCS, without deleting your working copy? (`touch foobar && ci foobar` will delete `foobar`, leaving only `foobar,v`.)
   - The RCS log for a file that is owned by root and chmodded 600 will list "root" as being responsible for every checkin. Find a way for different users with root access to differentiate their checkins.
   - How do you (a) view the RCS log (who made what changes and when) and (b) revert to a previous version of a file?

2. **[Mutt]** There's a gzipped mbox file (a flat-file mailstore) containing posts to the CS 61A newsgroup from October 2009. Download it, extract it, and try reading it with mutt. Figure out how to…

    - …switch between individual messages and list view
    - …enable threading ("sort mail by threads")
    - …show only messages matching a certain rule (e.g., "from Brian Harvey" or "sent before October 15")
    - …tag all messages matching a certain rule
    - …do something to all tagged messages (e.g., mark as read, forward, delete)

    If you're already conversant with mutt, you might try playing around with mail/mailx.

3. **[Makefiles]** There's a nice introduction to Makefiles at

    `http://www.cems.uvm.edu/∼snapp/maketutorial/make.html`

    Read through that page and then write a Makefile such that…

    - `make update` runs `apt-get update`.
    - `make install name=package` uses the Make dependency system to update APT, and then runs `apt-get install package`.
    - `make sandwich` prints out "Okay." if you're root and "What? Make it yourself." otherwise. (If this makes no sense to you, Google "sudo make me a sandwich.")

      To do this, you'll want to look up how to do conditionals and shell substitution in Makefiles. Note that prefixing a line with the "@" symbol will prevent make from printing that command as it is executed.
    - `make` with no arguments prints a usage statement.