Exercise 1

Stewart Johnston
CIS 150 – Intro to Database Administration
NCMC
johnstons1@student.ncmich.edu

September 4, 2018

1 Impactful databases

There are a handful of databases I interact with more or less directly on a regular basis:

- User database (Unix)
- User database (Windows)
- Google's user database
- Gmail's database
- Steam's game/user database
- etc

Which is not even to speak of those which I use indirectly, like site databases which track pages, posts, comments, tags, etc.

Of those listed, impactful is a toss-up between Unix's user system, and gmail's backend database. I use gmail on a frequent basis; rare is the day that I don't check it at least twice. I login to my machines, however, more frequently than that each day.

Once upon a time, Unix-based OSs relied on the `/etc/passwd` file almost exclusively. The name of the file relates to the utility of the same name. It has the dubious benefit of being a plaintext file, with no fancy bells or whistles going into its formatting other than the humble newline and colon. Each line lists one user's record, and each record's fields are separated by ":" characters. The file strongly resembles a table of rows and columns, and in this way bears similarity to the relational model so ubiquitous now.

`/etc/passwd` is further similar in the sense that each record starts with a unique field indicating the username. The list of fields for each user are as such:

1. Username. Must be unique across users listed in the file

2. Information to validate a user's password. Modern implementations almost always store the actual password information in some other, secure spot. The encrypted shadow file is one such example. This field is usually used to indicate to the OS where to find that information.

3. UID; the user identifier. This does not need to be unique. (One multiple-OS workaround is to retool each system's UID for that user to match so that files are shareable on one disk.)

4. GID; the (primary) group identifier. Depending on the umask, files created by this user may be initially available to this group. (Both this and UID are the primary means by which permissions are controlled on files, and are read by the kernel for this purpose. ACLs are a more modern implementation which allow for more complex permissions than simple UID and GID can allow.

5. The GECOS field. A normally comma-separated list describing, for example, the user's real name, phone number, and so on.

6. Absolute path to the user's home directory.

7. Path to the program started every time the user logs in.

I have born personal witness to this file and its partner named shadow on at least some linux distributions sharing a lineage with ubuntu and debian. This archaic method is still functional, certainly, and used. As mentioned, constructs like Access Control Lists now exist to supplement the functionality detailed here. The shadow file is itself a similar text-based database, but the primary fields of its records are encrypted. Shadow has similarly unique primary key functionality, since by definition it is attached to the username.

However, some modern OSs only use `/etc/passwd` in single-user mode. MacOS includes such a disclaimer at the top of the file, with commentary to check the documentation on Open Directory (found in man page `opendirectoryd(8)`). Single user mode precludes the need for checking standard user login information, instead assuming that the user is an administrator attempting to repair software. (Security concerned users should store sensitive information with encryption, or use whole-disk encryption such as Filvault on MacOS.) As such, the passwd file now serves the role of listing users created for internal use by the OS and other software. This is where almighty root may be found.

Exploring the inner workings of the Open Directory daemon and its kin is outside the scope of this report. However, the presence of such daemons as launchd and opendirectoryd betray MacOS' BSD lineage.