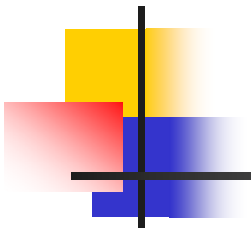
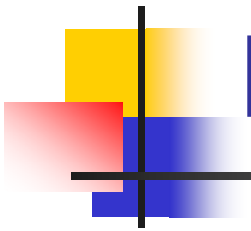
Chapter 6: System Data Files and Information

CMPS 105: Systems Programming
Prof. Scott Brandt
T Th 2-3:45
Soc Sci 2, Rm. 167



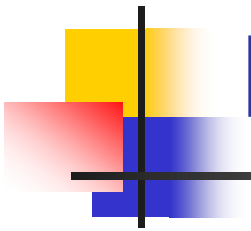
Introduction

- Lots of system parameters, configuration information, and status information is stored in files
- Usually ASCII text files
- Why?
- Why are some things compiled in and others in config files?



Password File

- User name: char *pw_name;
- Encrypted password: char *pw_passwd;
- Numerical user ID: uid_t pw_uid;
- Numerical group ID: gid_t pw_gid;
- Comment field: char *pw_gecos;
- Initial working directory: char *pw_dir;
- Initial shell: char *pw_shell;



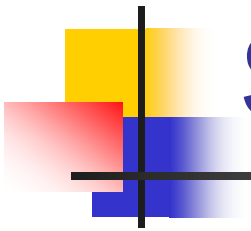
Details

- Usually an entry with username root
- One-way password encryption
  - 13 characters (from 64 character set)
- Fields can be empty
- Some unixes support other fields
- root:jheVopR58x9Fx:The superuser:/:/bin/sh

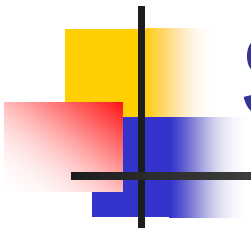
Accessing the password file

- #include <sys/types.h>
- #include <pwd.h>
- struct passwd *getpwuid(uid_t uid);
  - maps uid (from file i-node) to password entry
- struct passwd *getpwnam(const char *name);
  - maps username (from login) to password entry

Searching the password file

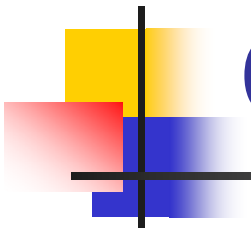


- #include <sys/types.h>
- #include <pwd.h>
- struct passwd *getpwent(void);
  - Return the next entry in the password file
- void setpwent(void);
  - Rewind password file to the beginning
- void endpwent(void);
  - Close the password file
  - Note no corresponding open: getpwent() does that for us



Shadow Passwords

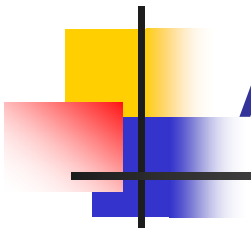
- Hackers can guess lots of passwords, encrypt them, and compare them to password file entries
  - If there is a match, they know the password
  - Note: do NOT try this at home
- Some systems avoid this by storing the encrypted passwords in a *shadow file*
- Sometimes also employ password aging
- Other password questions?



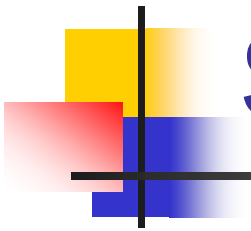
Group File

- Group name: char *gr_name;
- Encrypted password: char *gr_passwd;
  - Not part of POSIX
- Numerical group ID: gr_gid;
- Array of pointers to individual user names: char **gr_mem;

Accessing Group File

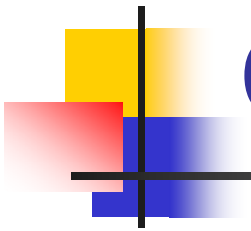


- #include <sys/types.h>
- #include <grp.h>
- struct group *getgrgid(gid_t gid);
- struct group *getgrnam(const char *name);
- struct group *getgrent(void);
- void setgrent(void);
- void endgrent(void);
- Parallel the passwd functions



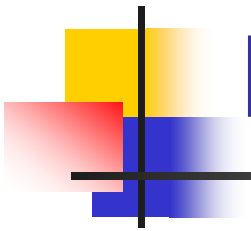
Supplementary Group IDs

- Used to have to use newgrp() to change groups
- Now, all group IDs are checked upon any access
- #include <sys/types.h>
- #include <unistd.h>
- int getgroups(int gidsetsize, gid_t grouplist[]);
  - Gets list of groups for current user
- int setgroups(int ngroups, const gid_t grouplist[]);
  - Sets list of groups for current user
- int initgroups(const char *username, gid_t basegid);
  - Reads group file and then calls setgroups() for a user
  - Used by login



Other data files

- Some systems support similar files for other purposes
- /etc/hosts
- /etc/services
- /etc/protocols
- /etc/networks
- All support get, set, end, similar to passwd and group files



Login Accounting

- Two data files: utmp and wtmp
- utmp tracks all users currently logged in
- wtmp keeps track of all logins and logouts
- Let’s check them out

System identification

- #include <sys/utsname.h>
- int uname(struct utsname *name);
- struct utsname {
  - char sysname[9]; // OS
  - char nodename[9]; // Host
  - char release[9]; // OS Release
  - char version[9]; // OS Version
  - char machine[9]; // Machine (hw)
- };
- Some systems: int gethostname(char *name, int namelen);

Time and Date Routines

- Basic time services counts seconds since the Epoch
- #include <time.h>
- See figure on page 156
- time_t time(time_t *calptr);
- struct tm *gmtime(const time_t *calptr);
- struct tm *localtime(const time_t *calptr);
- time_t mktime(struct tm *tmptr);
- char *asctime(const struct tm *tmptr);
- char *ctime(const time_t *calptr);
- size_t strftime(char *buf, size_t maxsize, const char *format, const struct tm *tmptr);