CS 447/647

Shells

Ken Thompson has an automobile which he helped design. Unlike most automobiles, it has neither speedometer, nor gas gauge, nor any of the other numerous idiot lights which plague the modern driver. Rather, if the driver makes a mistake, a giant "?" lights up in the center of the dashboard. "The experienced driver," says Thompson, "will usually know what's wrong."

—Anonymous

Overview

What is a shell?

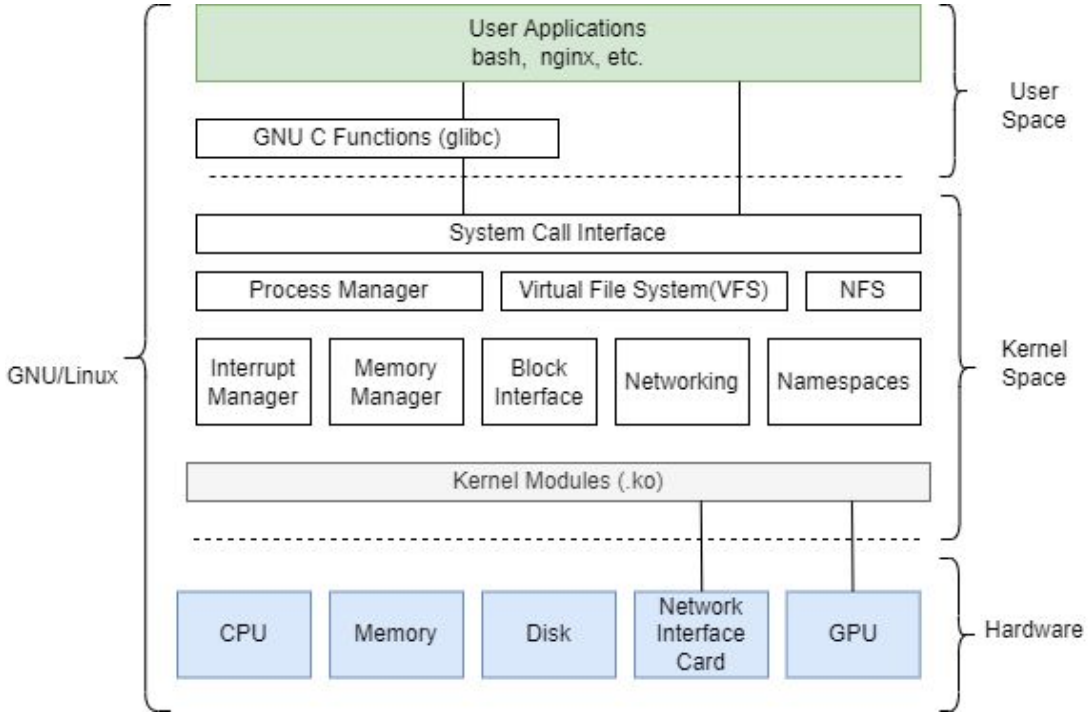
What are commands?

What are ELF binaries?

How are binaries located?

How are shared libraries located?

How do we build a dynamic user space?


User Applications
bash, nginx, etc.
GNU C Functions (glibc)
User Space
System Call Interface
Process Manager
Virtual File System(VFS)
NFS
Interrupt Manager
Memory Manager
Block Interface
Networking
Namespaces
Kernel Space
GNU/Linux
Kernel Modules (.ko)
CPU
Memory
Disk
Network Interface Card
GPU
Hardware

What is a shell?

- A command interpreter
  - fork()
  - exec()
  - wait()

```
158   // Read and run input commands.
159   while(getcmd(buf, sizeof(buf)) >= 0){
160     if(buf[0] == 'c' && buf[1] == 'd' && buf[2] == ' '){
161       // Chdir must be called by the parent, not the child.
162       buf[strlen(buf)-1] = 0;  // chop \n
163       if(chdir(buf+3) < 0)
164         printf(2, "cannot cd %s\n", buf+3);
165       continue;
166     }
167     if(fork1() == 0)
168       runcmd(parsecmd(buf));
169     wait();
170   }
171   exit();
```

cd

run

https://github.com/mit-pdos/xv6-public/blob/master/sh.c Line 158

What does it interpret?

- Commands typed by a user or read from a file.
  - Symbols: `<>|&$` ()`
  - Keywords: `cd, if, while, for, exit`
- Searches for the commands on the system.
  - PATH environment variable
- What does it execute?
  - language features
    - |
    - for, if, while, function
  - ELF binaries*

```c
typedef struct elf64_hdr {
  unsigned char e_ident[EI_NIDENT]; /* ELF "magic number" */
  Elf64_Half e_type;        /* ELF Type, commonly: executable, shared libary */
  Elf64_Half e_machine;     /* Architecture, commonly: AMD x86-64 */
  Elf64_Word e_version;     /* ELF version, CURRENT or INVALID */
  ...
```

https://github.com/torvalds/linux/blob/master/include/uapi/linux/elf.h

GNU coreutils (ELF binaries)

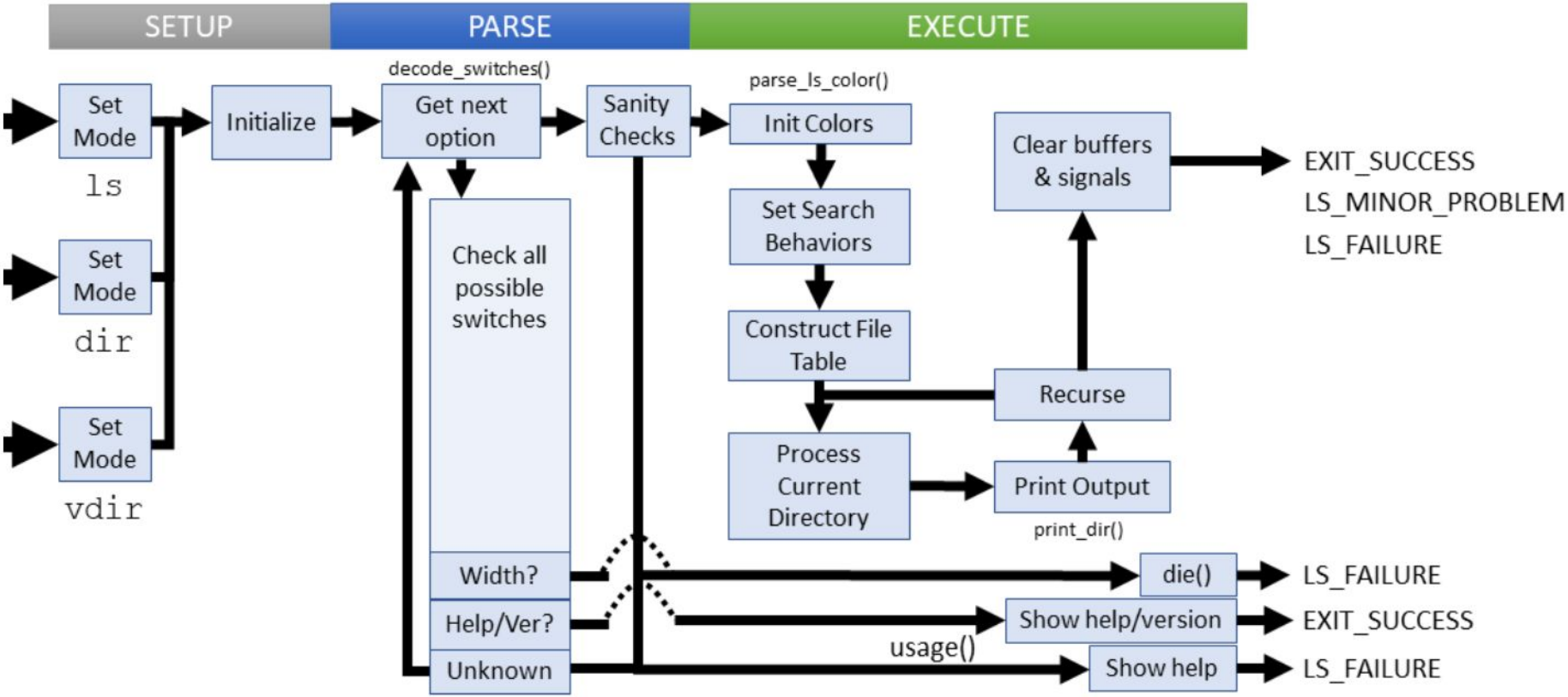
Core utilities which are expected to exist on every operating system

head
tail
wc
md5sum
sort
shuf
uniq
comm
split
tr
basename
mktemp
uptime

ls
rm
cp
mv
ln
chgrp
chown
chmod
echo
printf
tee
false

https://www.gnu.org/software/coreutils/

ls(1)



https://www.maizure.org/projects/decoded-gnu-coreutils/ls.html

Why do we use a shell?

- Managing Services and Hardware
  - Application Layer Protocols
    - DHCP, DNS
- Engineering and Research
  - Scripting
  - Glueing
- Debugging
- Modify runtime parameters

Why are shells still popular?

- Lightweight
- Accessibility
  - Blanket exception for CLI applications
  - Receive Federal money, must be accessible
- Ease of use*
- Reproducibility
- Automation
- Change Control
  - git

The UNIX-HATERS Handbook

"Two of the most famous products of Berkeley are LSD and Unix. I don't think that is a coincidence."

Edited by Simson Garfinkel, Daniel Weise, and Steven Strassmann

Illustrations by John Klossner

Million Dollar Settlements of Closed Captioning Website Accessibility Lawsuits Highlight Need for Dual Approach

MIT - $1,000,000 in attorney fees

Arizona State University

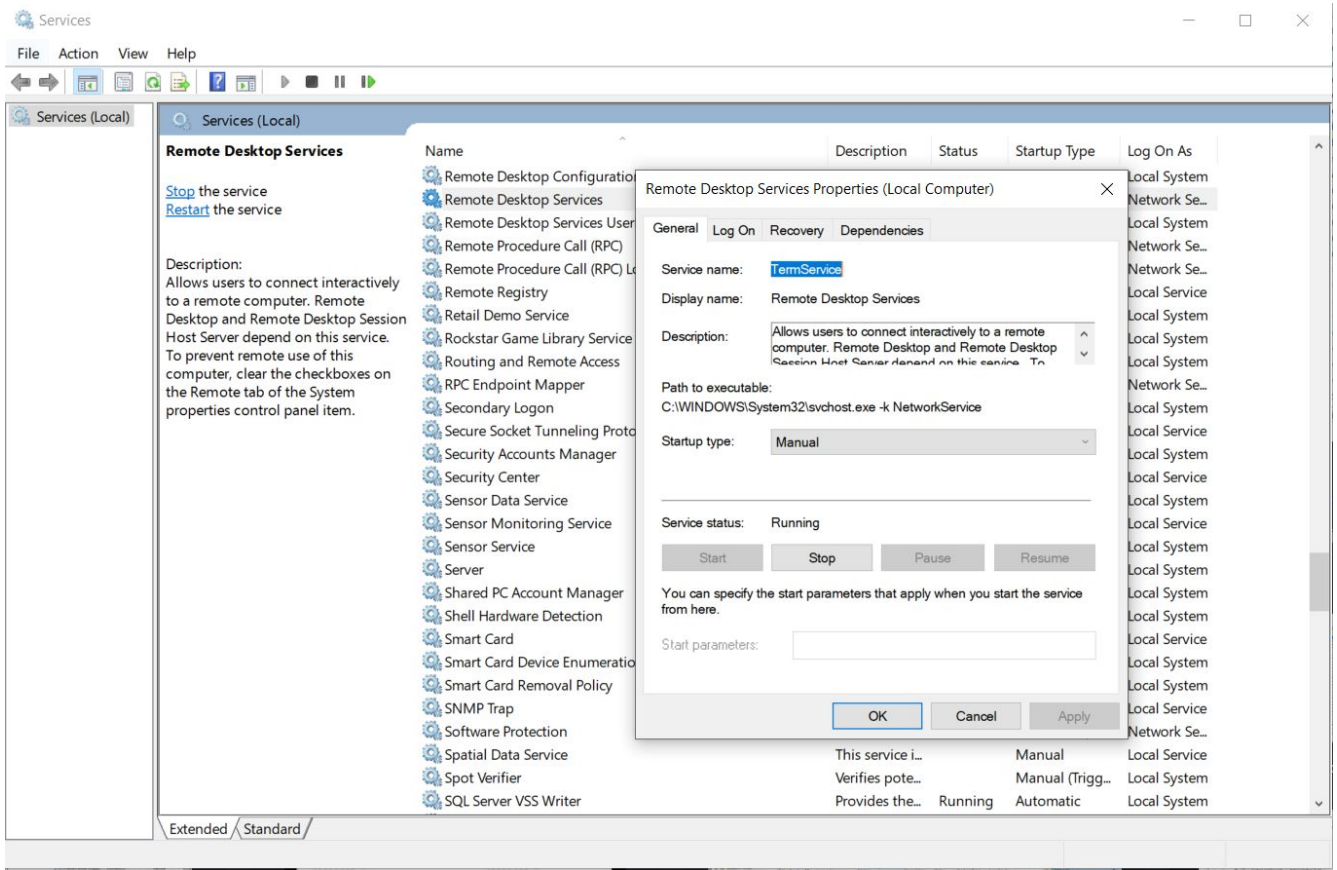
PennState

OHIO UNIVERSITY

United States Supreme Court Denies Petition from Domino's Pizza

Lazar, J. Managing digital accessibility at universities during the COVID-19 pandemic. *Univ Access Inf Soc* (2021). https://doi.org/10.1007/s10209-021-00792-5

Example - Windows Service Management



Example - Linux

```
systemctl restart bind9
systemctl status bind9
```

```
# systemctl status bind9
● named.service - BIND Domain Name Server
     Loaded: loaded (/lib/systemd/system/named.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2022-02-10 17:44:20 PST; 1 day 18h ago
       Docs: man:named(8)
   Main PID: 10800 (named)
      Tasks: 8 (limit: 9493)
     Memory: 56.7M
        CPU: 23.505s
     CGroup: /system.slice/named.service
             └─10800 /usr/sbin/named -f -u bind

Feb 10 17:44:20 cs447-s22-newellz2-server named[10800]: automatic empty zone: view internal: B.E.F.IP6.ARPA
Feb 10 17:44:20 cs447-s22-newellz2-server named[10800]: automatic empty zone: view internal: 8.B.D.0.1.0.0.2.IP6.ARPA
Feb 10 17:44:20 cs447-s22-newellz2-server named[10800]: automatic empty zone: view internal: EMPTY.AS112.ARPA
Feb 10 17:44:20 cs447-s22-newellz2-server named[10800]: automatic empty zone: view internal: HOME.ARPA
Feb 10 17:44:20 cs447-s22-newellz2-server named[10800]: obtaining root key for view external from '/etc/bind/bind.keys'
Feb 10 17:44:20 cs447-s22-newellz2-server named[10800]: set up managed keys zone for view external, file 'external.mkey>
Feb 10 17:44:20 cs447-s22-newellz2-server named[10800]: configuring command channel from '/etc/bind/rndc.key'
Feb 10 17:44:20 cs447-s22-newellz2-server named[10800]: command channel listening on 127.0.0.1#953
Feb 10 17:44:20 cs447-s22-newellz2-server named[10800]: configuring command channel from '/etc/bind/rndc.key'
```

bash configuration and dotfiles

- `/etc/bash.bashrc` - System-wide .bashrc file for interactive bash(1) shells.
- `/etc/profile` - Profile file for sh(1) and bash(1)
- `/etc/profile.d` - Collections of profile scripts. Often used for $PATH modifications
  - modules!
- `~/.bashrc` - Interactive shell configuration file
- `~/.bash_profile` - Executed for login shells

Best way to learn bash?

Use it!

Getting help - man(1)

“Classic Unix documentation is written to be **telegraphic** but complete… The style assumes an active reader, one who is **able to deduce obvious unsaid consequences** of what is said, and who has the self-confidence to trust those deductions. **Read every word carefully**, because you will seldom be told anything twice.”

Eric Raymond

Pathname	Contents
/bin	Core operating system commands
/boot	Boot loader, kernel, and files needed by the kernel
/compat	On FreeBSD, files and libraries for Linux binary compatibility
/dev	Device entries for disks, printers, pseudo-terminals, etc.
/etc	Critical startup and configuration files
/home	Default home directories for users
/lib	Libraries, shared libraries, and commands used by **/bin** and **/sbin**
/media	Mount points for filesystems on removable media
/mnt	Temporary mount points, mounts for removable media
/opt	Optional software packages (rarely used, for compatibility)
/proc	Information about all running processes
/root	Home directory of the superuser (sometimes just /)
/run	Rendezvous points for running programs (PIDs, sockets, etc.)
/sbin	Core operating system commands [a]
/srv	Files held for distribution through web or other servers
/sys	A plethora of different kernel interfaces (Linux)
/tmp	Temporary files that may disappear between reboots
/usr	Hierarchy of secondary files and commands
/usr/bin	Most commands and executable files
/usr/include	Header files for compiling C programs
/usr/lib	Libraries; also, support files for standard programs
/usr/local	Local software or configuration data; mirrors **/usr**
/usr/sbin	Less essential commands for administration and repair
/usr/share	Items that might be common to multiple systems
/usr/share/man	On-line manual pages
/usr/src	Source code for nonlocal software (not widely used)
/usr/tmp	More temporary space (preserved between reboots)
/var	System-specific data and a few configuration files
/var/adm	Varies: logs, setup records, strange administrative bits
/var/log	System log files
/var/run	Same function as **/run**; now often a symlink
/var/spool	Spooling (that is, storage) directories for printers, mail, etc.
/var/tmp	More temporary space (preserved between reboots)

a. The distinguishing characteristic of **/sbin** was originally that its contents were statically linked and so had fewer dependencies on other parts of the system. These days, all binaries are dynamically linked and there is no real difference between **/bin** and **/sbin**.

manpage sections

Section	Contents
1	User-level commands and applications
2	System calls and kernel error codes
3	Library calls
4	Device drivers and network protocols
5	Standard file formats
6	Games and demonstrations
7	Miscellaneous files and documents
8	System administration commands
9	Obscure kernel specs and interfaces

man

```
STAT(1)                          User Commands                          STAT(1)

NAME
       stat - display file or file system status

SYNOPSIS
       stat [OPTION]... FILE...

DESCRIPTION
       Display file or file system status.

       Mandatory  arguments  to  long  options are mandatory for short options
       too.

       -L, --dereference
              follow links

       -f, --file-system
              display file system status instead of file status

       -c  --format=FORMAT
              use the specified FORMAT instead of the default; output  a  new-
              line after each use of FORMAT

       --printf=FORMAT
              like  --format, but interpret backslash escapes, and do not out-
              put a mandatory trailing newline; if you want a newline, include
              \n in FORMAT

       -t, --terse
              print the information in terse form

       --append-exe
              append .exe if cygwin magic was needed

       --help display this help and exit

       --version
              output version information and exit

       The valid format sequences for files (without --file-system):

       %a     access rights in octal (note '#' and '0' printf flags)

       %A     access rights in human readable form

       %b     number of blocks allocated (see %B)

       %B     the size in bytes of each block reported by %b

       %C     SELinux security context string

       %d     device number in decimal

       %D     device number in hex

       %f     raw mode in hex
```

How are man pages rendered?

`groff`

- System for typesetting documents
  - Similar to TeX..
  - troff in 1971.
  - groff in 1990
- More like a compiler
- Text input files with embedded formatting

`man 7 groff_man`

man

```
apt install -y man manpages manpages-dev info groff

mandb #Regenerate manpages from roff source.
      #Config in /etc/manpath.config

man -k ext4   # Keyword search for string “ext4”

man -K ext4   # Page through manpages that  contain ext4

man -a intro  # Page through the intro manual

man ls        # manpage for ls

# Why do we need any of this when Google exists?
```