Shell Filtering

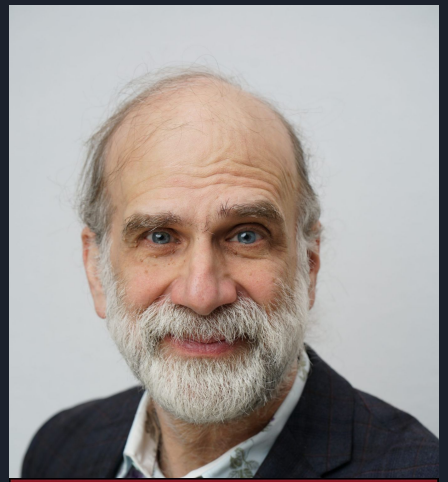
Level 0x02

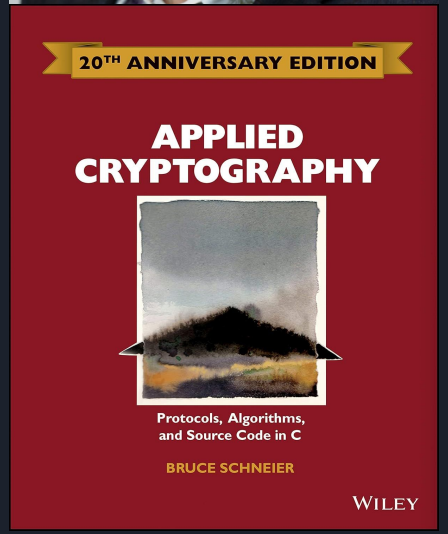
Quick Overview

- Shell Commands
- Unix Philosophy
- Filtering

Bruce Schneier

- Author of Applied Cryptography (1996)
  - He had to write this book cause it didn't exist / no one wrote one yet
  - 50 pages of source code, but no disks (export restrictions)
  - Came out before AES selected by NIST
- Wrote Blowfish and Twofish encryption
- Has a blog
- Critical of current crypto / blockchain craze
  - blockchain a solution in search of a problem
  - cryptocurrencies are useless and are only used by speculators looking for quick riches
  - It's not secure, it's not safe, it's not reliable, it's not trustworthy, it's not even decentralized, it's not anonymous, it's helping destroy the planet. I haven't found one positive use.





Directory Commands

Linux	Windows / DOS	Explanation
`mkdir DIRECTORY`	`mkdir`	**Ma**k**es** a new **dir**ectory
`cd DIRECTORY`	`cd`	**C**hanges **d**irectory
`cd ..`	`cd ..`	Changes to the parent directory
`rmdir DIRECTORY`	`rmdir`	**R**e**m**oves a **dir**ectory (must be empty)
`ls`	`dir`	Lists directory contents
`tree`	`dirtree`	Lists all files / subdirectories

File Commands

Linux	Windows / DOS	Explanation
`touch FILE`	`copy con FILE`	Creates a blank file
`cat FILE`	`type FILE [FILE2 …]`	Displays contents of a file, or con**cat**enates files
`head FILE`		Displays beginning (the **head**) of a file
`tail FILE`		Displays ending of a file (the **tail**)
`hexdump -C FILE`		Displays contents of a binary in **hex**adecimal
`file FILE`		Tells you what type of a file

File Commands

Command	Explanation
`strings FILE`	Prints out printable strings of a binary file
`sort [FILE]`	Prints lines in alphabetical order
`uniq [FILE]`	Removes redundant lines out output
`wc [FILE]`	**W**ord **c**ount. Counts number of words in a file
`dos2unix / unix2dos [FILE]`	Converts file line endings
`more / less [FILE]`	Shows output 1 page at a time
`grep needle [FILEs]`	Searches for a string

Unix Philosophy

1. **Small is Beautiful**
2. **Each Program Does One Thing Well**
3. Prototype as Soon as Possible
4. Choose Portability Over Efficiency
5. **Store Data in Flat Text Files**
6. Use Software Leverage
7. Use Shell Scripts to Increase Leverage and Portability
8. **Avoid Captive User Interfaces**
9. **Make Every Program a Filter**

Standard Input / Output

- 3 file descriptors open by CLI application
  - 0 = stdin (standard input)
  - 1 = stdout (standard output)
  - 2 = stderr (standard error)
- Pipes (|) can be used to connect output from one application to input of another application

```
strings somefile | grep -i password
```

```
cat logfile | sort | uniq
```

- Chaining of commands together end up like data stream processing, or filtering

I/O Redirection

- Using "`> file.txt`" after a command causes output from stdout to be redirected into a file
  - You won't be able to see it on screen
  - stderr will still be displayed
- Using "`2> file.txt`" after a command causes stderr to be redirected into file
- `tee` will write standard output to a file and also write it to the screen
  - Ex: `./myprogram arg1 arg2 | tee logfile.txt`
- >> will append to existing file, > overwrites it

Whats my favorite Linux command

- How could I go about figuring this out?
- What do you think the answer is?
- What about your system?

history

- Lists all the commands you have previously used
- Each type you type a shell command in it gets added to `~/.bash_history`
  - `ls -al ~ | grep history` revealed a python and gdb history too!
  - Security sidetrack….. Does history file contain sensitive data?

```
-P password
--password password
       Use  password  to  encrypt zipfile entries (if any).  THIS IS INSECURE!
```

- `history` output

```
1993  git push
1994  cd ..
1995  ls
1996  exit
1997  pavucontrol
1998  exit
1999  python3
2000  exit
2001  history
2002  history > scratch/history.txt
```

General Strategy

- Get rid of numbers in front
- Isolate the commands from arguments
- Sort and count the commands

Pitfalls Ahead:

- Formatted output with extra spaces
- Dangling quotes and double quotes

```
1993  git push
1994  cd ..
1995  ls
1996  exit
1997  pavucontrol
1998  exit
1999  python3
2000  exit
2001  history
2002  history > scratch/history.txt
```

Getting rid of extra spaces?

- Some programs output into easy-to-read columns
  - column
  - printf("%20s = %d\n", name, value);

```
mwales@Metroid:~/scratch/csclub$ cat nintendo_games.csv | grep Mario | cut -d ',' -f1-4 | tail -n 5
91,Mario Tennis,N64,"Aug 28
91,Mario Golf,N64,"Jun 30
79,Mario Party,N64,"Feb 8
83,Mario Kart 64,N64,"Feb 10
94,Super Mario 64,N64,"Sep 26
```

```
mwales@Metroid:~/scratch/csclub$ cat nintendo_games.csv | cut -d ',' -f 1-4 | grep -v DLC | column -t -s ',' | sort -n | tail -n 10
96          The Legend of Zelda: Breath of the Wild                    WIIU          "Mar 3
96          The Legend of Zelda: The Wind Waker                        GC            "Mar 24
96          The Legend of Zelda: Twilight Princess                     GC            "Dec 11
97          Metroid Prime                                              GC            "Nov 17
97          Perfect Dark                                               N64           "May 22
97          Super Mario Galaxy 2                                       WII           "May 23
97          Super Mario Galaxy                                         WII           "Nov 12
97          Super Mario Odyssey                                        Switch        "Oct 27
97          The Legend of Zelda: Breath of the Wild                    Switch        "Mar 3
99          The Legend of Zelda: Ocarina of Time                       N64           "Nov 23
```

Getting rid of extra spaces

- `echo`
- `tr`
- `sed`
- `awk`

```
mwales@Metroid:~/scratch/csclub$ echo "  too    many   spaces" | xargs echo
too many spaces
mwales@Metroid:~/scratch/csclub$ echo "  too    many   spaces" | sed -e "s/[[:space:]]\+/ /g"
 too many spaces
mwales@Metroid:~/scratch/csclub$ echo "  too    many   spaces" | tr -s ' '
 too many spaces
mwales@Metroid:~/scratch/csclub$ echo "  too    many   spaces" | awk '{$1=$1}1' OFS=" "
too many spaces
```

Uhhh ohhh

- Random dangling unpaired quote messing us up…

```
mwales@Metroid:~/scratch/csclub$ cat history.txt | xargs -I{} echo {} 2>&1 | tail -n 5
1728  ls
1729  cd running-interpreter/
1730  ls
xargs: unmatched single quote; by default quotes are special to xargs unless you use the -0 option
1731  vi run
mwales@Metroid:~/scratch/csclub$ cat history.txt | grep -C3 1730
 1727  git push
 1728  ls
 1729  cd running-interpreter/
 1730  ls
 1731  vi run
 1732  ls -l'
 1733  ls -l
mwales@Metroid:~/scratch/csclub$
```

sed - Stream EDitor

- Can probably do many things with it, but I mainly search and replace
  - s/old_text/new_text/g

```
mwales@Metroid:~/scratch/csclub$ echo "i'm told i'm old" | sed 's/old/new/g'
i'm tnew i'm new
```

- Can do regular expressions too
- Use ‘single’ quotes to avoid shell expansion

```
mwales@Metroid:~/scratch/csclub$ cat history.txt | sed 's/"//g' | sed "s/'//g" | grep -C3 1730
 1727  git push
 1728  ls
 1729  cd running-interpreter/
 1730  ls
 1731  vi run
 1732  ls -l
 1733  ls -l
```

Look ma, no spaces

- How do I get rid of those numbers? Command args?


```
mwales@Metroid:~/scratch/csclub$ cat history.txt | sed 's/"//g' | sed "s/'//g" | xargs -I{} echo {} | tail -n 5
1998  exit
1999  python3
2000  exit
2001  history
2002  history > scratch/history.txt
mwales@Metroid:~/scratch/csclub$ cat history.txt | sed 's/"//g' | sed "s/'//g" | tr -s ' ' | tail -n 5
 1998 exit
 1999 python3
 2000 exit
 2001 history
 2002 history > scratch/history.txt
mwales@Metroid:~/scratch/csclub$ cat history.txt | sed 's/"//g' | sed "s/'//g" | xargs -I{} echo {} | tr -s ' ' | tail -n 5
1998 exit
1999 python3
2000 exit
2001 history
2002 history > scratch/history.txt
```

cut

- Can specify a delimiter and which fields to keep
- Can select certain bytes

```
mwales@Metroid:~/scratch/csclub$ echo "| markdown | table | data | sep | by | pipes" | cut -d '|' -f 2,4,5
 markdown | data | sep
mwales@Metroid:~/scratch/csclub$ echo "this also works good with spaces" | cut -d ' ' -f 2,4,6
also good spaces
```

Just words now

- Now what can we do with a long sequence of words?

```
mwales@Metroid:~/scratch/csclub$ cat history.txt | sed 's/"//g' | sed "s/'//g" | xargs -I{} echo {} | tr -s ' ' | cut -d ' ' -f 2 | tail -n 5
exit
python3
exit
history
history
mwales@Metroid:~/scratch/csclub$
```

`sort` and `uniq`

- `sort`: is sorts…
  - `-f` = ignore case
  - `-n` = numeric sort
- `uniq`: removes duplicates
  - `-c` = count duplicates
  - `-i` = ignore case

```
mwales@Metroid:~/scratch/csclub$ cat history.txt | sed 's/"//g' | sed "s/'//g" | xargs -I{} echo {} | tr -s ' ' | cut -d ' ' -f 2 | sort | uniq -c | sort -n | tail -n 15
      5 pdftk
      6 chmod
      7 rm
     12 mkdir
     13 mv
     15 cp
     19 cat
     20 python3
     25 sudo
     28 docker
     37 exit
     89 vi
    117 git
    206 cd
    292 ls
```

Attributions

- Linux and the Unix Philosophy by Mike Gancarz.
- https://github.com/yaylinda/nintendo-games-ratings/blob/master/data.csv
- https://www.schneier.com/photo/