Networking

networking lecture & practice for CSC420 Operating Systems Spring 2022 Lyon College

README

- This file accompanies lectures on the shell and `bash(1)`. To gain practice, you should type along in your own Org-mode file. You have to have Emacs and my `.emacs` file installed on your PC or the Pi you're working with.
- This section is based on chapter 16 of Shotts, The Linux Command Line (2e), NoStarch Press (2019) - "Networking".
- To make this easier, use the auto expansion (`<s`). This will only work if you have my `.emacs` file (from GDrive) installed and the `org-tempo` library loaded.

- Add the following two lines at the top of your file, and activate each line with `C-c C-c` (this is confirmed in the echo area as `Local setup has been refreshed`)):

```
#+PROPERTY: header-args:bash :results output :exports both
```

- Remember that `C-M-\` inside a code block indents syntactically (on Windows, this may only work if you have a marked region - set the mark with `C-SPC`).

- To **not** see the emphatic characters like ~ or * or / in the Org file text, run the following code chunk (or put the code in your `/.emacs` file): if successful, you should see `"t"` in the minibuffer.

```
(setq-default org-hide-emphasis-markers t)
```

  If you don't put it in your `/.emacs` file, the command will only work for the current Emacs session.

- If you have difficulty distinguishing the code blocks from the documentation, change your Emacs theme with `M-x custom-themes` - `Leuven` is great if you like a light theme, or `Manoj-dark` if you like it dark.

Overview

- When it comes to networking, there is nothing that cannot be done with Linux.
- Linux is used to build networking systems like firewalls, routers,name servers, network-attached storage (NAS) boxes, etc.

- There is a vast number of commands - we focus on the most frequently used ones to monitor networks, transfer files and facilitate remote work:

COMMAND	MEANING
`ping`	Send an echo request to network hosts
`traceroute`	Print the route packets trace to a host
`ip`	Show/manipulate routing, devices, tunnels
`netstat`	Print network connections, routing tables, interface statistics, masquerade connections and multicast memberships
`ftp`	Internet file transfer program

COMMAND	MEANING
`irc`	Internet relay chat program
`wget`	Non-interactive network downloader
`ssh`	OpenSSH secure shell client (remote login)

- This section also assumes some familiarity with the concepts
  - Internet Protocol (IP) address
  - Domain and host name
  - Uniform Resource Identifier (URI)

Internet addresses

- An IP address is a numerical label like `192.168.1.10`
- It identifies a network interface and enables connections to host computers (computers with an OS and user access)
- There are two types of (standard) Internet protocols, IPv4 (32-bit) and IPv6 (128-bit)

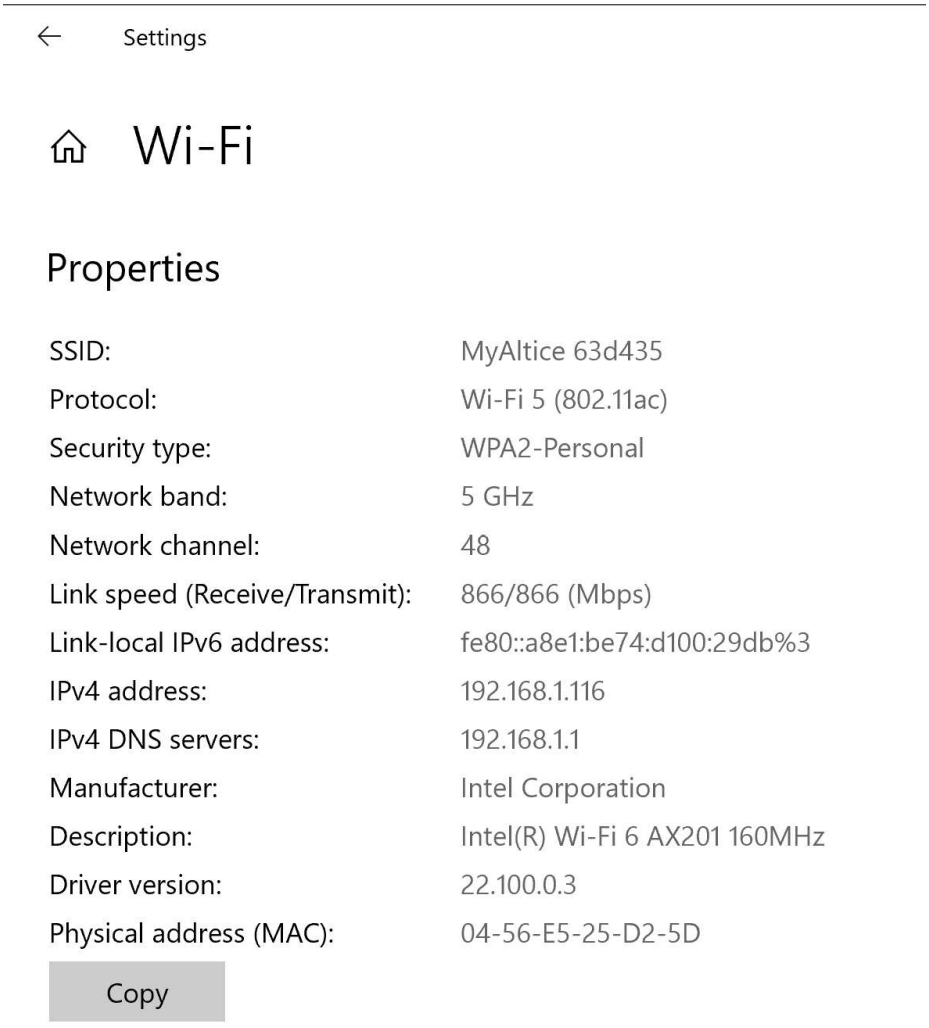

Figure 1: WiFi information on a Windows Box

- The IPv4 and IPv6 addresses are dynamical
- The MAC address is static

- The DHCP (Dynamical Host Configuration Protocol) assigns IP addresses to devices connected to the network
- [ ] Check if a `dhcp` process runs on your computer. Use two methods on the terminal or on an Emacs shell:

  1. process check with `ps aux` - use `grep` to search for `dhcp`.

     ```
     ps aux | grep --exclude=grep dhcp
     ```

     ```
     marcus      12  0.0  0.0  16208  1284 tty1     S    09:10   0:00 grep --exclude=grep dhcp
     ```

  2. search for `dhcp` in the output of the `systemctl` service program (the program that talks to `systemd`). Pipe the output of `systemctl status` into `grep`.

     ```
     systemctl status | grep dhcp
     ```

  3. [ ]

     Find out what the active flags `-b -q` for the `dhcpcd(8)` program mean.

     `-b` stands for "background" (startup scripts) `-q` stands for "quiet" (level of system messages)

  4. On the man page, you find the information that this daemon program implements an `RFC` - a Request For Comment. This is the traditional (since 1969) title for standard-setting documents for the Internet, or more specifically for TCP/IP (Transmission Control Protocol/Internet Protocol)

The Internet and the Web

- The World-Wide Web is a collection of web pages on the Internet
- Similar to a shopping mall with road access
- Web locations and Internet addresses are linked but not identical
- The Internet's name space is structured by standardized strings: Universal Resource Identifiers (URI), a Universal Resource Locators (URL) and a Universal Resource Name (URN).
- A URI contains both URL and URN.

- URI syntax:

  ```
  scheme://authority]path[?query][#fragment]
  ```

SYNTAX ELEMENT	EXAMPLES
Scheme	http, file, ftp, data, irc
Authority	userinfo@, host (IP), port (80)
Path	path to the resource
Query	query string
Fragment	direction to secondary resource

- URI Examples:

URI	WHAT
mailto:birkenkrahe@lyon.edu	user mail

URI	WHAT
https://github.com/birkenkrahe/os420/…/README.org#my-first-pi	GitHub link
http://ftp.gnu.org/gnu/emacs/	GNU Emacs file server
irc.freenode.net	Internet Relay Chat

Network address on Windows

- Go to Settings > Network & Internet > Wi-Fi > Hardware properties

  Here you see the IPv4 adress, e.g. for my computer: 192.168.1.116

- On Windows, the `hostname` command only gives you the

Examining and monitoring a network

Look at the man page for each of these programs if you're interested in learning more, and try some of the many options.

Knock-knock who's there: `ping`

- The `ping` command sends a special network packet called an `ICMP ECHO_REQUEST` to a specified host.
- Most network devices receiving this packet will reply to it, allowing the network connection to be verified.
- You can configure the network connection to ignore these packets (for security reasons).

- A typical packet reply looks like this:

  ```
  64 bytes from ord38s29-in-f14.1e100.net (142.250.191.142): icmp_seq=3 ttl=57 time=31.9 ms
  ```

  It contains the packet size, the target IP, time to live and transmitting time information.

- When you interrupt the communication with `C-c C-c` or `CTRL-c`, you get some stats:

  ```
  --- google.com ping statistics ---
  2 packets transmitted, 2 received, 0% packet loss, time 1002ms
  rtt min/avg/max/mdev = 31.358/31.437/31.517/0.079 ms
  ```

- A properly performing (transparent) network will exhibit 0 % packet loss and indicate that different network elements (interface cards, cabling, routing, gateways) are OK.
- `[ ]`

  Go to the terminal or to an Emacs shell and ping Lyon College, and then Google. Do you see a difference?

  ```
  ping lyon.edu
  ping google.com
  ```

What's the route: `traceroute`

- This program lists all the hops network traffic takes to get from the local system to the specified host. Here is the route from my house to lyon.edu:

  ```
  ~/GitHub $ traceroute lyon.edu
  traceroute to lyon.edu (40.119.1.254), 30 hops max, 60 byte packets
   1  Docsis-Gateway (192.168.1.1)  8.721 ms  8.383 ms  8.302 ms
  ```

```
 2  * * *
 3  173-219-255-40.suddenlink.net (173.219.255.40)  18.387 ms  18.322 ms  18.240 ms
 4  173-219-221-143.suddenlink.net (173.219.221.143)  40.121 ms  39.998 ms  39.903 ms
 5  173-219-221-138.suddenlink.net (173.219.221.138)  39.556 ms  39.471 ms  39.343 ms
 6  173-219-17-110.suddenlink.net (173.219.17.110)  38.896 ms  42.320 ms  41.960 ms
 7  173-219-152-172.suddenlink.net (173.219.152.172)  41.850 ms  41.737 ms  41.700 ms
 8  66-76-232-151-chic.tex.sta.suddenlink.net (66.76.232.151)  41.498 ms  35.773 ms  35.771 ms
 9  ae35-0.icr01.ch4.ntwk.msn.net (104.44.237.19)  45.126 ms  45.163 ms ae30-0.icr01.ch2.ntwk.msn.r
10  be-100-0.ibr01.ch2.ntwk.msn.net (104.44.11.252)  58.979 ms be-120-0.ibr02.ch2.ntwk.msn.net (104
11  be-6-0.ibr02.dsm05.ntwk.msn.net (104.44.18.217)  60.892 ms be-4-0.ibr01.dsm05.ntwk.msn.net (104
12  be-7-0.ibr02.sn1.ntwk.msn.net (104.44.16.38)  53.861 ms  53.767 ms be-9-0.ibr01.sn1.ntwk.msn.ne
13  ae100-0.icr01.sn6.ntwk.msn.net (104.44.23.78)  64.181 ms ae124-0.icr03.sn1.ntwk.msn.net (104.44
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
~/GitHub $
```

- Traversing the entire route requires 13 routers.
- Each router stop is accompanied by 3 rount-trip times.
- For routers that do not give identifying information (because of network congestion, firewalls, etc.) you see asterisks
- The `-T` and `-I` options (different probes) sometimes gives more information (and requires `sudo` rights)

- `[ ]` Check the `traceroute` to `lyon.edu` yourself, and contrast it again with the `traceroute` to `google.com`. Check if `-T` or `-I` make a difference.

What's interfaced: `ip` and `ifconfig`

- The `ip(8)` program is a multi-purpose network configuration tool
- The `ifconfig(8)` program is the older (deprecated) version of `ip`
- `[ ]`

  Run `ip a` in the code block below.

```
ip a
```

```
18: eth0: <> mtu 1500 group default qlen 1
    link/ether 04:56:e5:25:d2:61
    inet 169.254.0.79/16 brd 169.254.255.255 scope global dynamic
       valid_lft forever preferred_lft forever
    inet6 fe80::5809:6379:fd0e:4f/64 scope link dynamic
       valid_lft forever preferred_lft forever
10: eth1: <BROADCAST,MULTICAST,UP> mtu 1500 group default qlen 1
    link/ether 0a:00:27:00:00:0a
    inet 192.168.56.1/24 brd 192.168.56.255 scope global dynamic
       valid_lft forever preferred_lft forever
    inet6 fe80::69da:73c3:7432:ff6a/64 scope link dynamic
       valid_lft forever preferred_lft forever
1: lo: <LOOPBACK,UP> mtu 1500 group default qlen 1
```

```
    link/loopback 00:00:00:00:00:00
    inet 127.0.0.1/8 brd 127.255.255.255 scope global dynamic
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host dynamic
       valid_lft forever preferred_lft forever
3: wifi0: <BROADCAST,MULTICAST,UP> mtu 1500 group default qlen 1
    link/ieee802.11 04:56:e5:25:d2:5d
    inet 192.168.1.116/24 brd 192.168.1.255 scope global dynamic
       valid_lft 2243sec preferred_lft 2243sec
    inet6 fe80::a8e1:be74:d100:29db/64 scope link dynamic
       valid_lft forever preferred_lft forever
2: wifi1: <> mtu 1500 group default qlen 1
    link/ieee802.11 04:56:e5:25:d2:5e
    inet 169.254.118.103/16 brd 169.254.255.255 scope global dynamic
       valid_lft forever preferred_lft forever
    inet6 fe80::5fc:9e9a:14eb:7667/64 scope link dynamic
       valid_lft forever preferred_lft forever
9: wifi2: <> mtu 1500 group default qlen 1
    link/ieee802.11 06:56:e5:25:d2:5d
    inet 169.254.5.2/16 brd 169.254.255.255 scope global dynamic
       valid_lft forever preferred_lft forever
    inet6 fe80::a58b:f6a0:f53d:502/64 scope link dynamic
       valid_lft forever preferred_lft forever
```

    - My system has three network interfaces:
    - `lo` for loopback, used by the OS to talk to itself
    - `eth` for Ethernet interface
    - `wlan0` for the WiFi interface
- When peforming diagnostics, look for the word `UP` in the first line (which means it's enabled), and a valid IP address in the `inet` field. E.g. I only have WiFi right now, and no Ethernet.
- `[ ]`

  Run `ifconfig -a` in the code block below. This command is a little easier to understand, I think.

```
ifconfig -a
```

What's on the net: `netstat`

- This program is used to examine network settings and statistics.
- `[ ]`

  Run `netstat -ie` to examine network interfaces.

```
netstat -ie
```

- The output of `netstat -ie` looks similar to the `ifconfig` command because it focuses on network interfaces. Only `lo` and `wlan0` transport any packets.
- `[ ]`

  Run `netstat -r` to see the routing table, which shows how the network is configured to send packets from network to network:

```
netstat -r
```

    - This is a typical table for a client on a local area network (LAN) behind a firewall/router. The first line shows the destination IP, `192.168.1.0`. The last 0 means that the adress refers to multiple hosts.
    - The Gateway is the name or router to go from the current host to the destination network.
    - The I[nter]face to connect is WiFi (`wlan0`).

- [ ]

  Run the `hostname -I` command to see your own host on your LAN:

  ```
  hostname -I
  ```

  ```
  192.168.56.1 192.168.1.116
  ```

  Mine is `192.168.1.160`. One of about 20 network devices in my house, including: PS5, SmartTV, Kindle tablets, PCs, Mac, Raspberry Pi (this is it), network printers etc.

Transporting files over a network with `ftp` and `wget`

FTP

- `ftp` (File Transfer Protocol) is a "classic" program. It is supported by all web browsers.
- [ ]

  Check is the `ftp` daemon `ftpd` is awake.

  ```
  systemctl status | grep ftpd
  ```

- [ ]

  Check if the program `ftp` is even available/installed.

  ```
  which ftp
  ```

- FTP in its original form is **not safe** because it sends account names and passwords in clear text, i.e. unencrypted. Anyone sniffing the network can see them.
- Therefore, all FTP traffic on the Internet is done by *anonymous* FTP servers that allow you to login using the username *anonymous* and a meaningless password.
- Try `lftp` - a better FTP client. On Windows, I used Cyberduck years ago. Dedicated clients offer a lot more features.

An FTP session

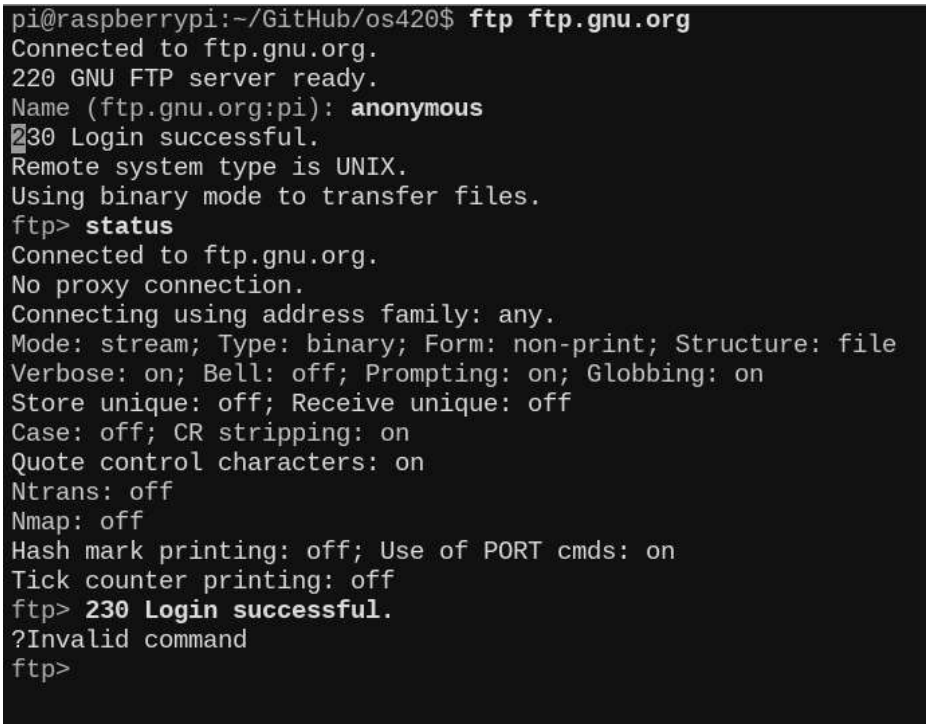

Figure 2: FTP session example

- [ ] Install `ftp` as `sudo` - use an Emacs shell or a terminal for that.
- [ ] In a terminal or on an Emacs shell, run `ftp`
- [ ] On the `ftp>` shell, type `help` to see the available commands
- [ ] Open a connection with `open`
- [ ] As target IP address, enter `ftp.gnu.org`
- [ ] Login as `anonymous`
- [ ] List the current directory with `ls`
- [ ] Get the `README` file with `get README`
- [ ] Rename `README` on your computer to `README1`
- [ ] Send `README1` to the other location with `send README1`
- [ ] Close the connection with `close` and quit with `quit`

wget

- `wget` is a tool for file downloading both from web and FTP sites. It exhibits network resilience, e.g. it will keep trying to get the job done even if the network is slow or unstable. It does the job in the background
- `wget` uses "recursive downloading" and recreates the entire file structure that it finds remotely at the local site while respecting the local "Robot Exclusion Standard" of the `robots.txt` file[1].
- You can download files, directories, and entire sites.
- [ ]

  Download the Lyon College landing page with `wget` using `lyon.edu` as the only command. The program will substitute any other information necessary.

  - Check the current directory for the result (`index.html`).

```
wget lyon.edu
```

  - Go to a terminal (or the Emacs shell) and run the command there again to see the full screen message:

```
pi@raspberrypi:~/GitHub/os420$ wget lyon.edu
--2022-04-21 08:25:02--  http://lyon.edu/
Resolving lyon.edu (lyon.edu)... 40.119.1.254
Connecting to lyon.edu (lyon.edu)|40.119.1.254|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: http://www.lyon.edu/ [following]
--2022-04-21 08:25:02--  http://www.lyon.edu/
Resolving www.lyon.edu (www.lyon.edu)... 40.119.1.254
Reusing existing connection to lyon.edu:80.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://www.lyon.edu/ [following]
--2022-04-21 08:25:02--  https://www.lyon.edu/
Connecting to www.lyon.edu (www.lyon.edu)|40.119.1.254|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 146047 (143K) [text/html]
Saving to: ‘index.html’

index.html          100%[===================>] 142.62K  --.-KB/s    in 0.1s

2022-04-21 08:25:03 (1.08 MB/s) - ‘index.html’ saved [146047/146047]

pi@raspberrypi:~/GitHub/os420$
```

Figure 3: wget terminal screen message

  - The man page is better-than-average and highly readable.

- In Emacs, you can also go to the `Dired` buffer with `C-x d` and type `! chromium browswer RET` on the file `index.html`. This will open the page locally in a browser.

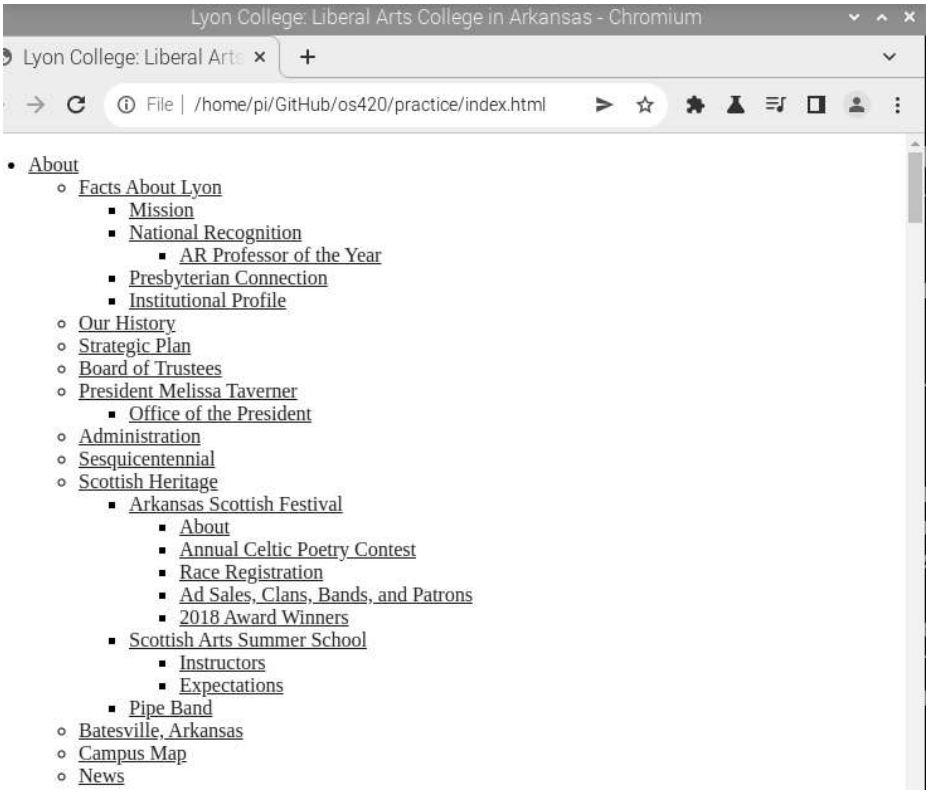

Figure 4: local copy of lyon.edu/index.html in Chromium browser

Secure communication with `ssh`

- Popular remote access programs included `rlogin` and `telnet`
- Like `FTP` these transmit all their communications in cleartext
- `ssh` (Secure SHell) on the other hand:
  - authenticates that the remote host is who it says it is (preventing so-called "man-in-the-middle" attacks)
  - encrypts all of the communication between the local and remote hosts
- On Raspberry Pi, you may need to enable `ssh` using the `sudo raspi-config` terminal command. This opens a screen dialog.
- `[ ]`

  To check if `ssh` is active, run

  ```
  ps aux | grep sshd
  ```

  ```
  marcus      45  0.0  0.0  16208  1280 tty1     S    09:10   0:00 grep sshd
  ```

  I get this result, which shows that I (as `pi`) have one active secure shell connections. It is controlled by `root` and asleep.

  ```
  : root     17230  Ss   Apr20   0:00 sshd: pi [priv]
  : pi       17236  S    Apr20   0:00 sshd: pi@pts/1
  ```

  Here are the Emacs buffers that show the remote connection:

  ```
  %  /rclone:pi@gdrive:/       1971 Dired by name    /rclone:pi@gdrive:/
  * *tramp/rclone pi@gdrive*     0 Fundamental
  ```

- SSH consists of two parts:
  - An SSH server runs on the *remote host*, listening for incoming connections by default on port 22
  - An SSH client runs on the *local system* to communicate with the remote server.
- To enable a system to receive remote connections, it must have the OpenSSH-server package installed, configured, and running, and (if the system is behind a firewall) it must allow incoming connections on TCP port 22.
- `[ ]` Connect with your neighboring Pi using SSH.

  - Make sure ssh is alive and running: type `systemctl status ssh` in a terminal or Emacs shell or run the block below and check the file.

    ```
    systemctl status ssh > ssh.status
    cat ssh.status
    ```

    ```
    ● ssh.service - OpenBSD Secure Shell server
         Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
         Active: active (running) since Fri 2022-04-15 12:48:23 CDT; 5 days ago
           Docs: man:sshd(8)
                 man:sshd_config(5)
        Process: 531 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
       Main PID: 556 (sshd)
          Tasks: 1 (limit: 4915)
            CPU: 4.204s
         CGroup: /system.slice/ssh.service
    ```

```
             └─556 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

Apr 20 21:06:12 raspberrypi sshd[17170]: Connection reset by authenticating user pi 192.168.1
Apr 20 21:07:34 raspberrypi sshd[17201]: Connection closed by 192.168.1.116 port 64135 [preau
Apr 20 21:09:37 raspberrypi sshd[17209]: Accepted password for pi from 192.168.1.116 port 641
Apr 20 21:09:37 raspberrypi sshd[17209]: pam_unix(sshd:session): session opened for user pi(u
Apr 20 21:10:20 raspberrypi sshd[17230]: Accepted password for pi from 192.168.1.116 port 641
Apr 20 21:10:20 raspberrypi sshd[17230]: pam_unix(sshd:session): session opened for user pi(u
Apr 20 21:10:43 raspberrypi sshd[17295]: Accepted password for pi from 192.168.1.116 port 641
Apr 20 21:10:43 raspberrypi sshd[17295]: pam_unix(sshd:session): session opened for user pi(u
Apr 20 21:11:49 raspberrypi sshd[17332]: Accepted password for pi from 192.168.1.116 port 641
Apr 20 21:11:49 raspberrypi sshd[17332]: pam_unix(sshd:session): session opened for user pi(u
```

- Find your own and the other party's hostname with `hostname -I`: this is the only information you need, apart from the username.

```
hostname -I
```

```
192.168.56.1 192.168.1.116
```

- In a terminal: enter `ssh pi@hostname` then enter the password, which is `ly0Np1_Numb3r_xx` where `xx` is the number of your Pi. `hostname` is the IP address you just obtained.
- If successful, check that you're on the other machine by checking the SSH `systemctl status` and/or the `hostname`. You can even open Emacs here with `emacs -nw` (non-graphical Emacs).

- In Emacs: open a `Dired` buffer with `C-x d` and at the prompt, enter

```
/ssh:pi@hostname:~/
```

  You should now see the other computer's `/home/pi` directory. Open a shell with `M-x shell` and you'll see that it will open on the other computer.

Emacs special

- Tramp ("Transparent Remote (file) Access, Multiple Protocol" is a built-in GNU Emacs package that provides remote file editing.
- Tramp works directly with Dired using a command like:

```
C-x d /ssh:pi@192.168.1.160:~/
```

  to connect to a user `pi` on a local network machine. You can open a shell on the other machine, too, and work remotely at ease.

- The `rclone` program e.g. uses Tramp as an external method to connect to network servers like GDrive, or to facilitate cloud backup.

Challenge: How to connect your Pi to GDrive

Sources: rclone.org/drive.

1. Install rclone on Raspbian

```
$ sudo apt-get install rclone
```

2. Change file permissions of `$HOME/.config/rclone` to `rwx` for owner only

```
$ chmod 0700 ./.config/rclone  # change permissions
$ ls -la ./.config/rclone      # check - you should see drwx------
```

3. Follow the detailed instructions in rclone.org/drive until you see the `Success!` web page after connecting rclone to GDrive

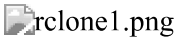

Figure 5: Selecting GDrive account for rclone

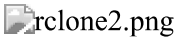

Figure 6: Give rclone permission to access GDrive

> "Note that rclone runs a webserver on your local machine to collect the token as returned from Google if you use auto config mode. This only runs from the moment it opens your browser to the moment you get back the verification code. This is on http://127.0.0.1:53682/ and this it may require you to unblock it temporarily if you are running a host firewall, or use manual mode."

The access information is stored in `$HOME/.config/rclone/rclone.conf`.

You can now access your GDrive from the Pi from the command line with: `rclone [cmd] gdrive:`, or inside GNU Emacs as an external method:~ `C-x d /rclone:gdrive:/`.

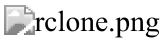

Figure 7: GDrive via rclone in Emacs

TODO Summary

Footnotes:

[1] This file specifies rules for web crawlers. If you have your own web server somewhere with an Internet connection, you're likely to have such a file. You can e.g. use it to block sites. It has a simple syntax. Here is a simple guide.

Author: Marcus Birkenkrahe
Created: 2022-04-21 Thu 09:10