Laboratory #8: Systemd & SSH keys

Unix (420-321-VA) - Fall 2021
Teacher: Tassia Camoes Araujo

Goals:

1. Connect to remote machines using SSH
2. Create and use SSH keys
3. Practice service management with Systemd

Instructions

Part I: Connect to a remote machine

In Lab#7 we started using SSH, but still in the local machine (localhost alias). This time you should find a partner and exchange ip addresses to experience a remote connection.

1. You can find your own ip address with the command *$ ip address show.* Your ip will be listed in one of your network interfaces information, just after the “inet” word. Look for a standard ip for local networks, like 192.168..., 10.0.0..., or 172.... Note: the address 127.0.0.1 represents your loopback interface (localhost alias).
2. Each of you should create a user to the other in your local machines.
3. Connect to the remote machine by using *$ ssh user@host*
4. If the command doesn’t work, check if the service is running in the other machine, and in which port (maybe you need ot use the -p option).

Part II: SSH Keys

Authentication through SSH keys is more secure than using password. You can think of a key as being a super big password. And more than size, it brings the power of asynchronous encryption scheme.

1. Start by changing the password of your partner’s account, so that she/he won’t be able to login again using the old password anymore
2. Create a pair of private-public SSH keys for yourself. Follow instructions on the following article and record the commands you’ve used:
   https://docs.ovh.com/ca/en/dedicated/creating-ssh-keys-dedicated/
3. Find a partner and exchange your public keys (e.g.: ~/.ssh/id_rsa.pub). Alternatively, you could practice using 2 different users, ideally, in 2 different Unix systems.
4. Place each other keys in your authorized_keys file (~/.ssh/authorized_keys). You can actually add multiple keys to this file (find other partners if you wish).

5. Both of you should restart the sshd service, and you should be able to login to each other accounts in the remote machine via ssh using your keys. Your local private key is used in the authentication to the remote machine.

Part III: Service management with systemd

A service is a software that runs in the background waiting to be used. In a Linux system, many of its essential capabilities are provided by services, such as login, graphical interface (eg. X11 and Wayland), printing (common unix printing system – cups), secure shell (ssh), web server (eg. Apache and nginx), database (eg. MariaDB) and many others. The process of a service is referred to as a *daemon process*, and their names usually end with the letter "d" (eg. sshd, httpd, cupsd etc).

1. Check all services currently being offered by your system:

   ```
   # systemctl list-units --type=service
   ```

2. Check the status of particular services, for example:

   ```
   # systemctl status systemd-timesyncd
   # systemctl status sshd
   ```

3. Stop, start, restart services, and check the status after each action:

   ```
   # systemctl stop systemd-timesyncd
   # systemctl status systemd-timesyncd
   # systemctl start systemd-timesyncd
   # systemctl status systemd-timesyncd
   # systemctl restart systemd-timesyncd
   # systemctl status systemd-timesyncd
   ```

4. Enable (and disable) persistent services, to always (or not) start it at boot time:

   ```
   # systemctl disable systemd-timesyncd
   # systemctl reboot
   # systemctl status systemd-timesyncd
   # systemctl enable systemd-timesyncd
   ```

5. Create an executable script file with an infinite loop that logs data in a file.
6. Add the new service to systemd, to be started along with the graphical interface:
   a) Reading: *https://www.digitalocean.com/community/tutorials/understanding-systemd-units-and-unit-files*
   b) Create the new service configuration unit file in */etc/systemd/system/mydaemon.service*
   c) Enable the service with systemctl
   d) Reboot and show the status of the service just after rebooting

Part IV: Deliverables

1. For parts I and II, include your commands and screenshots showing that the remote connections were authorized.
2. For part III, include your daemon script and configuration file, as well as a record of the change of status at boot (once you enable or diable the service).

Good luck!