VE482 — Introduction to Operating Systems

Lab 10

Manuel — UM-JI (Fall 2020)

Goals of the lab

- Proper device handling with udev
- Operating system init with systemd
- Interprocess communication with dbus
- A basic hack to gain root priviledges

1 A clean setup

After a couple of hours working in the back of your mum and Mr. Frown you have finally completed your dice device driver. Although everything is working you have quite enjoyed the process and to have a feeling a full accomplishment you want to polish your work. For instance you want the module to be automatically loaded when the system boot ups, and you imagine that if your grandpa had friends with an account on the same computer they would enjoy playing too.

To solve those questions you need to check the following points:

- Where to copy the dice module for it to be officially known to the kernel?
- What command to run in order to generate the `modules.dep` and `map` files?
- How to ensure the dice module is loaded at boot time, and how to pass it options?
- How to create a new `friends` group and add grandpa and his friends to it?
- What is `udev` and how to define rules such that the group and permissions are automatically setup at device creation?

2 A discreet gambling setup

The challenge is now to get this module on the family computer without anybody noticing it. If someone tells your mother then both you and your grandpa will get in trouble. So you need a way to get root access without inputing a wrong password and without knowing it! As it is too risky to open the computer and take the hard disk to plug it on your own computer you have find another strategy.

2.1 Hacking mum's computer

After a bit of thinking you have an interesting idea: on Linux systems the executables can be found in the `PATH`, a colon separated list of directories. If more than one binary have the same name, then the one found first is used. As you know your mum often uses the command `su` to become root, it thus suffices to write a simple script called `su`, place it in a directory of your choice that you then prepend to the `PATH` variable. When she will type `su` it will run your script. To your mum its behaviour should feel similar to the real `su`, i.e. prompt for the root password and behave as if a wrong password had been input. In fact in the background it should send you an email containing the root password and clean up all your traces, that is delete the script, and reset the `PATH` to its original value. That way your mum will think she has made a mistake and will rerun `su`. That time the correct command will be called and she will gain root access.

Therefore you think of the following points:

- How adjust the PATH, ensure its new version is loaded but then forgotten?
- What is the exact behaviour of `su` when wrong password is input?
- When using the `read` command how to hide the user input?
- How to send an email from the command line?

Once you have completed this simple script you patiently wait for the email.

2.2 Automatic setup

While you are connected as root you check the security setup. As expected it unfortunately uses `tripwire`, meaning the database is totally encrypted. However you notice that the plain text configuration files have not been deleted. Therefore you exactly know what directories are monitored and which are not! In particular you notice that `/etc/systemd`, and none of the `/etc/cron` folders are monitored. This is more than enough for you to develop a plan of action.

Your idea is simple: (i) each time mum connects, regardless of your grandpa being connected or not, all dice devices and the module must be removed, (ii) each time grandpa connects and mum is not connected the module should be loaded and the devices created, and (iii) finally ensure that `tripwire` does not detect you.

To effectively implement your idea you decide to use `systemd` to have a service starting up a daemon monitoring the user connections through `dbus`. Then you decide to run the daemon inside a `tmux` session, such that you can access it an any time. Now you can apply two strategies: the first one consists in "hiding" your new files in the middle of other files, for instance pretending your daemon is a library, so putting it in `/usr/lib` with a name similar to a library name; the second strategy is to hide it using a name above any suspicion.

While you think of those two ideas you feel a mix of both would be useful: everyday when `tripwire` is run your new files should not appear, so for instance you will have to remove your module before it runs. During this time you can use the first strategy, for the rest of the time the second will work perfectly fine. So you need a safe filename...

All of a sudden you have The Idea! The command line calculator or Computer Algebra system call PARI/GP. GP like GrandPa! It is already installed on the computer and `/usr/bin/gp` is a link to `/usr/bin/gp-2.11`. If you create a file `/usr/bin.gp-2.10` it will look like an uninstallation script forgot it! This is decided, that will be the name of your daemon called by `systemd`.

To trick `tripwire` you will copy all the new files in `/var/log/` using the usual log filename pattern such as `gp`, `gp.1`, and tt gp.2.gz.[1] As you realise that all your traces have to be covered before `tripwire` starts you will need to ensure your `cron` script is run first.

Although your idea in clear in theory you need to refresh your memory on the following points:

- What is `systemd`, where are service files stored and how to write one?
- How to get a `systemd` service to autostart?
- What is the difference between running `tmux` from the `systemd` service or from the `gp-2.10` daemon?
- What is `dbus` and how to listen to all the system events from the command line?

[1] To prevent log files from growing too large always use `logrotate` which will automatically handle them.

- What is `tmux`, when is it especially useful, and how to run a detached session?
- What is `tripwire`, what are some alternatives, and why should the configuration files also be encrypted and their corresponding plaintext deleted?
- What is `cron` and how to use it in order to run tasks at a specific time?

Now that you are fully ready you take your chance. If you are not caught in the next few days it means you succeeded, otherwise you will probably need to find a new place to live...