In the last laboratory we learned how to replace a library to generate an attack. In this laboratory we will try to see how we can prevent the installation of malicious files.

There are a lot of ways you can get infected:

- Files with malware
- Files specially crafted to take advantage of known vulnerabilities of popular applications (for example a pdf specially crafted to use a buffer overflow from the Acrobat Reader parser).
- Important operating system or popular applications files which are patched with malicious code
- Malicious web content (like javascripts)
- Malicious web content which take advantage of a browser vulnerability
- Etc

As we can see in many cases a source of malware spreading is represented by files. In this laboratory we will try to monitor each file from a certain folder, to verify its level of trustiness and to block infected files.

We will use inotify to monitor each file access. Inotify API provides a mechanism through which we can monitories filesystem events. Inotify can be used to monitor files and folders. When we monitor a folder, inotify will provide events for any changes done to the folder or to the files it contains.

API inotify:

```
#include <sys/inotify.h>
int inotify_init(void);
```

Function inotify_init is used to create an instance of inotify in the calling process and it returns a file descriptor to a new inotify events queue.

```
#include <sys/inotify.h>

int inotify_add_watch(int fd, const char *pathname,
uint32_t mask);
```

Function inotify_add_watch is used to add a watcher for certain events. The function parameters are file descriptor previously obtained with inotify_init, pathname which is the path we want to monitor, and mask which represents the mask of events which we want to monitor, we can set a watch on a file or a folder and depending on what we set on the mask we can watch multiple events like :

- IN_CREATE – we get notifications when a file or folder was created in the monitored path.
- IN_ACCESS – a file or a folder from the monitored path was accessed.

The events can be read using function read and using as file descriptor the file descriptor we obtained initially with inotify_init.

In the laboratory archive you will find a folder inotify which contains a sample where you can see how you can use inotify api to monitor a certain path.

When we want to stop monitoring a certain path we can call the following function:

```
#include <sys/inotify.h>

int inotify_rm_watch(int fd, int wd);
```

where fd is the initial file descriptor obtained with inotify_init, and wd represents the file descriptor returned by inotify_add_watch.

To close the file descriptor obtained with inotify_init you will have to use normal close function used to close a normal file descriptor. For example:

```
close (fd);
```

Practic. Please create a new folder, lab3, in your home directory. Change the sample provided in the archive, to allow you to count how many times was accessed each file from lab3 folder. Please create another folder in the lab3 folder, lets call it lab3-1 and see if your application is counting how many times were accesssed each file from lab3-1 folder. Please explain the obtained results.

(To compile inotify.c file you will run this command: gcc -o in inotify.c)

In sha256 folder you will find a small program which will help you to compute sha256 hash for a file. You can compile it using this command:

gcc -o t test.c sha-256.c

In the same folder you will find also another file, called eicar.txt.

Practic. Please compute the sha256 hash for eicar.txt file. Now enter on this site https://www.virustotal.com/gui/home/upload and select search tab. Please insert the computed sha256 hash and see the results you obtain for that hash. Try similar tests with other files and see the results.

In folder curl from the archive you will find a small sample which uses curl to send a http(s) request to a server and to get the server response. Please look at the code. Change the requested url with different other urls you are usually accessing and see if you obtain those pages.

Practic Using the sample you have implemented with inotify please monitor a folder and for every created or accessed file do compute its sha256 hash and using virus total api https://developers.virustotal.com/reference and curl send a request to virus total api to see what it knows about the files computed hashes. If you obtain the answer the hash correspond to an infected file, change your code to delete the file.