System Programming – Project 2

For this project, you are required to write a system call which sets a flag in the task descriptor of a process. When this flag is set to `1`, the process is not listed in the `/proc` file system and cannot be seen using "`ps`" or "`top`". Also, if a process has its flag set to `1`, it cannot fork any new processes. The prototype for the system call will be

```
long set_hidden(pid_t pid, int flag);
```

The `hidden` flag can be `0` (for OFF) or `1` (for ON). The system call changes the value of this flag. Only processes having root privileges can successfully execute this system call. On error, the `set_hidden` system call returns an appropriate error message. Otherwise, it returns `0`.
If a process which has `hidden=1` makes a `fork` system call, no new processes will be created and the `fork` system call will return an appropriate error message.

To achieve this, you need to:

1. Add a new field to the task descriptor. The name and type of the field is:
   ```
   int hidden;
   ```
   Note: This field should be added to the end of the task descriptor. If `hidden=1`, the process is not listed in the `/proc` file system and thus cannot be seen using "`ps`" or "`top`". It cannot also fork any new processes.
2. Modify the code used by the kernel when creating and initializing new processes. A newly created process should have its `hidden` field initialized to `0`. (Note: Learn how the process with `pid=0` is created and initialized in Linux.)
3. Write a system call which changes the value of the `hidden` field in the task descriptor if the caller process has root privileges. Add your system call to the kernel.
4. Modify the code that generates the `/proc` filesystem so that if the `hidden` field of a process is set to `1`, the process is not included.
5. Write a short test program that accepts the pid of the process and the flag value as input and makes the `set_hidden` system call. The test program should output the return value of the system call. Experiment by running the program with and without root privileges.
6. Write a short test program that makes a `fork` system call. The test program should output the return value of the `fork` system call. Experiment by running the program with the two flag values.

References:

- Chapters 3, 7 and 10 of the book "Understanding the Linux Kernel, 3rd Edition" by Daniel P. Bovet, Marco Cesati (Publisher: O'Reilly Pub, 2005) which is freely accessible from the ITU Library through Safari e-books.

Hints:

- `fs/proc/array.c: proc_pid_stat()` defines the format and fields to write in the `/proc/pid/stat` file.
- `fs/proc/base.c: proc_pid_readdir()` is used to read the `/proc/pid` directories.
- To see a list of default error codes, refer to the manual pages using "man errno".