ASSIGNMENT REPORT 1: PROCESS AND THREAD IMPLEMENTATION

CENG2034, OPERATING SYSTEMS

Emre Taşkın
emretaskin2@posta.mu.edu.tr
github:emretask1n

Thursday 4th June, 2020

Abstract

In this lab, I saw How child process works and How can I download the URLs with Child process. I also learned what is orphan process. I used different python libraries and techniques.

1 Introduction

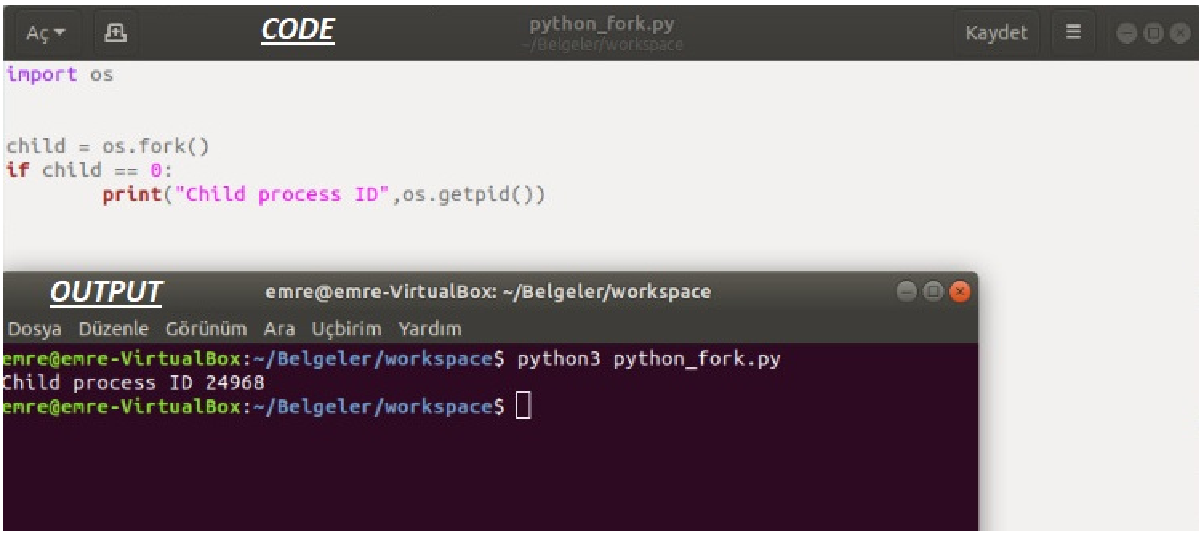
This labs purpose was learning how a child process works and what is the relation between child process and parent process. Using different libraries by using multiprocessing in python.

2 Assignments

I used a virtual machine to do my assignments which has 4GB RAM and 2 cores.This is why I used Pool(2) while doing multiprocessing in Assignment 2.4.

2.1 Assignment 1 Creating Child process

I created a child process with syscall and printed its PID.

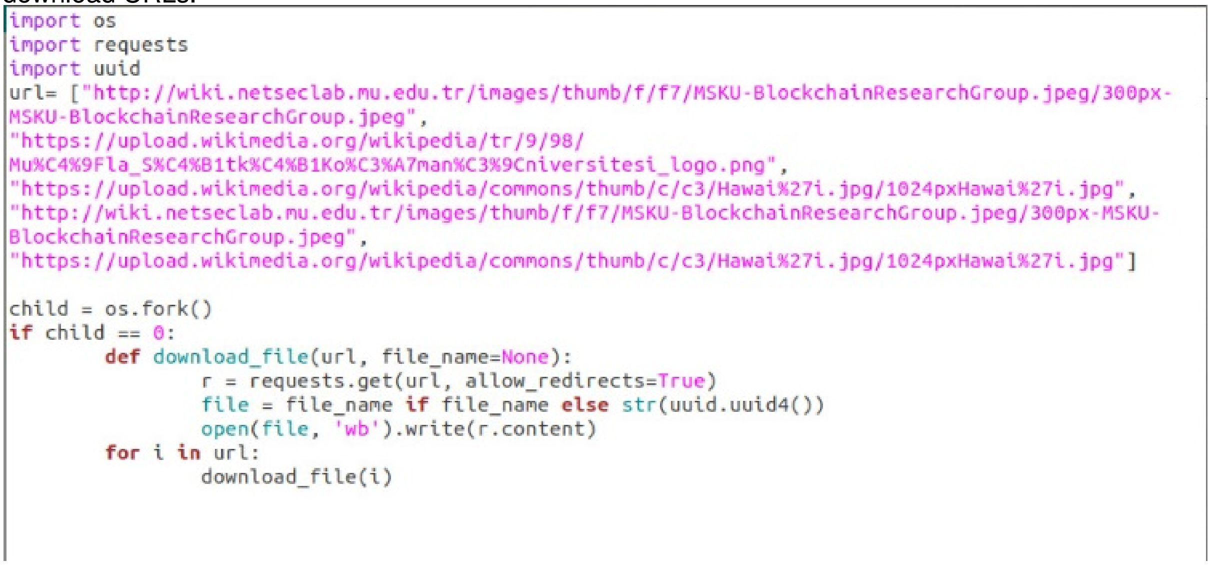

Aç
CODE
python_fork.py
~/Belgeler/workspace
Kaydet
OUTPUT
emre@emre-VirtualBox: ~/Belgeler/workspace
Dosya Düzenle Görünüm Ara Uçbirim Yardım

```python
import os

child = os.fork()
if child == 0:
        print("Child process ID",os.getpid())
```

```
emre@emre-VirtualBox:~/Belgeler/workspace$ python3 python_fork.py
Child process ID 24968
emre@emre-VirtualBox:~/Belgeler/workspace$ 
```

2.2 Assignment 2 Downloading files with the child process

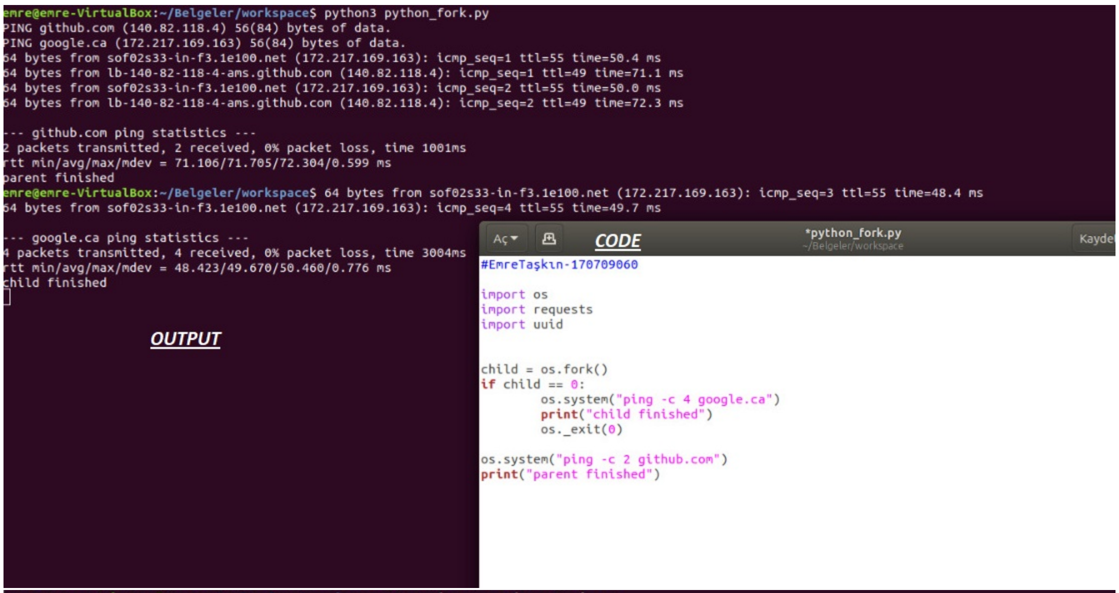
In this assignment I created a child process which is downloads the given URLs This is the code for download URLs.



```python
import os
import requests
import uuid
url= ["http://wiki.netseclab.mu.edu.tr/images/thumb/f/f7/MSKU-BlockchainResearchGroup.jpeg/300px-
MSKU-BlockchainResearchGroup.jpeg",
"https://upload.wikimedia.org/wikipedia/tr/9/98/
Mu%C4%9Fla_S%C4%B1tk%C4%B1Ko%C3%A7man%C3%9Cniversitesi_logo.png",
"https://upload.wikimedia.org/wikipedia/commons/thumb/c/c3/Hawai%27i.jpg/1024pxHawai%27i.jpg",
"http://wiki.netseclab.mu.edu.tr/images/thumb/f/f7/MSKU-BlockchainResearchGroup.jpeg/300px-MSKU-
BlockchainResearchGroup.jpeg",
"https://upload.wikimedia.org/wikipedia/commons/thumb/c/c3/Hawai%27i.jpg/1024pxHawai%27i.jpg"]

child = os.fork()
if child == 0:
        def download_file(url, file_name=None):
                r = requests.get(url, allow_redirects=True)
                file = file_name if file_name else str(uuid.uuid4())
                open(file, 'wb').write(r.content)
        for i in url:
                download_file(i)
```

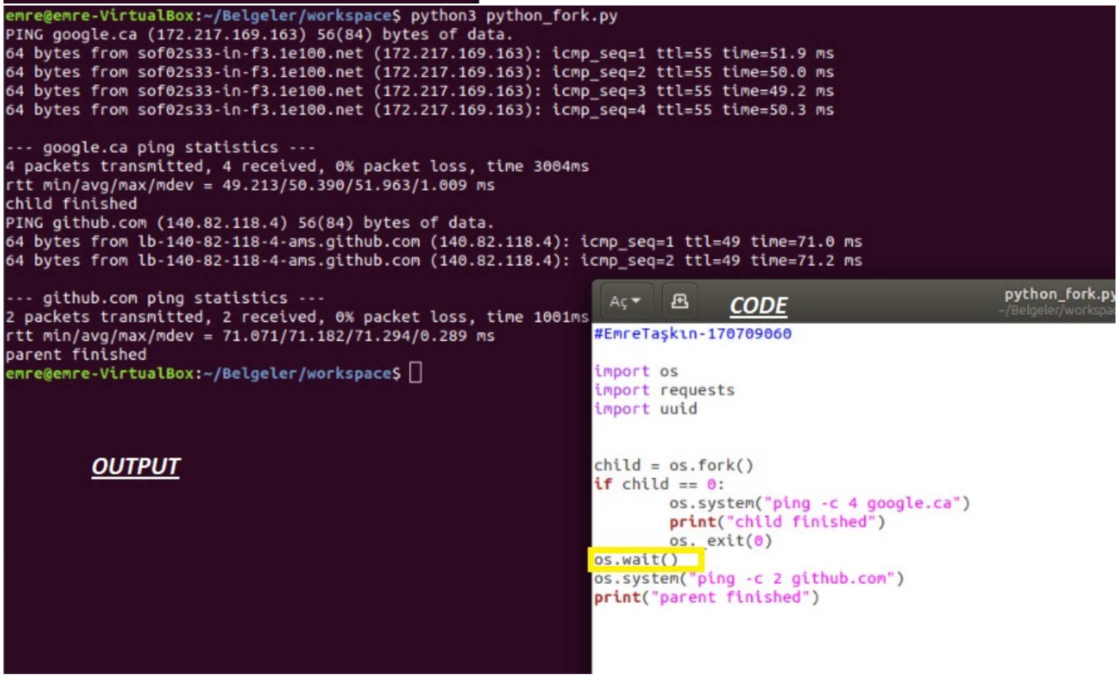
2.3 Assignment 3 How we can avoid from the orphan process situation?

An Orphan process is a running process whose parent process has finished or terminated.In the first picture child process is finished after parent process, this is called Orphan process. To avoid this situation we should use os.wait() as shown in second picture.



```
emre@emre-VirtualBox:~/Belgeler/workspace$ python3 python_fork.py
PING github.com (140.82.118.4) 56(84) bytes of data.
PING google.ca (172.217.169.163) 56(84) bytes of data.
64 bytes from sof02s33-in-f3.1e100.net (172.217.169.163): icmp_seq=1 ttl=55 time=50.4 ms
64 bytes from lb-140-82-118-4-ams.github.com (140.82.118.4): icmp_seq=1 ttl=49 time=71.1 ms
64 bytes from sof02s33-in-f3.1e100.net (172.217.169.163): icmp_seq=2 ttl=55 time=50.0 ms
64 bytes from lb-140-82-118-4-ams.github.com (140.82.118.4): icmp_seq=2 ttl=49 time=72.3 ms

--- github.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 71.106/71.705/72.304/0.599 ms
parent finished
emre@emre-VirtualBox:~/Belgeler/workspace$ 64 bytes from sof02s33-in-f3.1e100.net (172.217.169.163): icmp_seq=3 ttl=55 time=48.4 ms
64 bytes from sof02s33-in-f3.1e100.net (172.217.169.163): icmp_seq=4 ttl=55 time=49.7 ms

--- google.ca ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 48.423/49.670/50.460/0.776 ms
child finished
```

```python
#EmreTaşkın-170709060

import os
import requests
import uuid


child = os.fork()
if child == 0:
        os.system("ping -c 4 google.ca")
        print("child finished")
        os._exit(0)

os.system("ping -c 2 github.com")
print("parent finished")
```

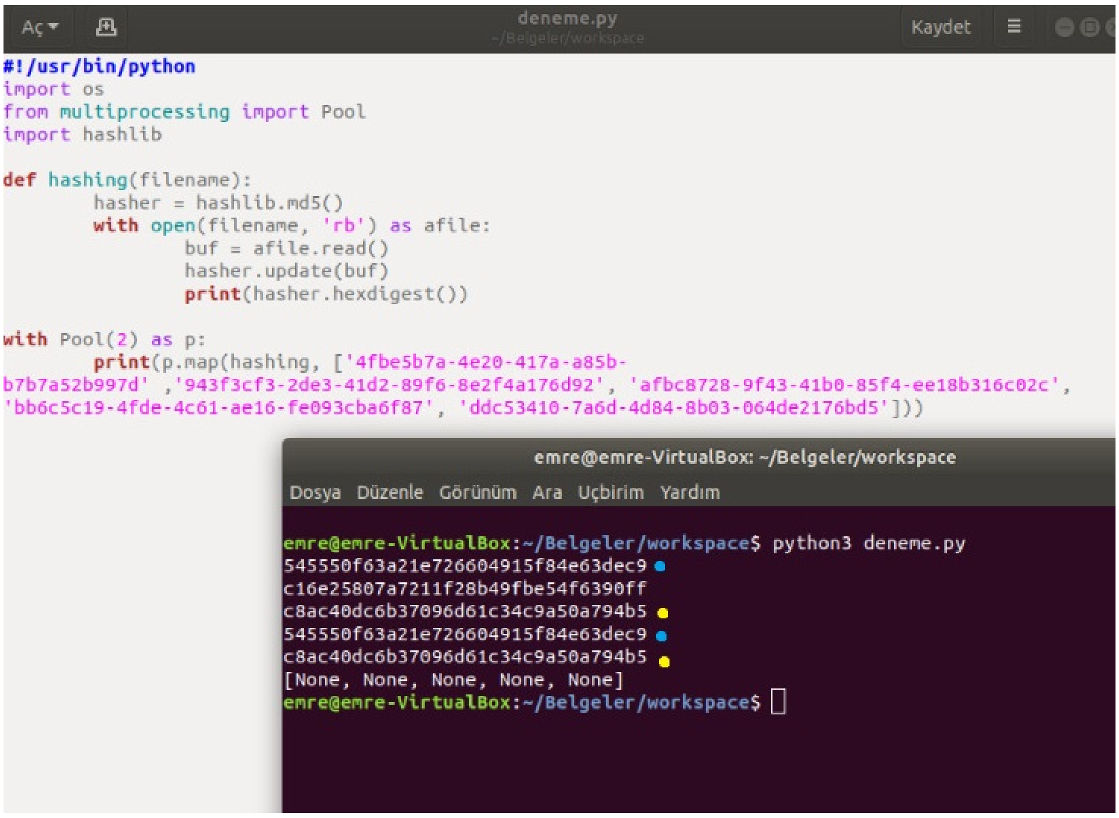


```
emre@emre-VirtualBox:~/Belgeler/workspace$ python3 python_fork.py
PING google.ca (172.217.169.163) 56(84) bytes of data.
64 bytes from sof02s33-in-f3.1e100.net (172.217.169.163): icmp_seq=1 ttl=55 time=51.9 ms
64 bytes from sof02s33-in-f3.1e100.net (172.217.169.163): icmp_seq=2 ttl=55 time=50.0 ms
64 bytes from sof02s33-in-f3.1e100.net (172.217.169.163): icmp_seq=3 ttl=55 time=49.2 ms
64 bytes from sof02s33-in-f3.1e100.net (172.217.169.163): icmp_seq=4 ttl=55 time=50.3 ms

--- google.ca ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 49.213/50.390/51.963/1.009 ms
child finished
PING github.com (140.82.118.4) 56(84) bytes of data.
64 bytes from lb-140-82-118-4-ams.github.com (140.82.118.4): icmp_seq=1 ttl=49 time=71.0 ms
64 bytes from lb-140-82-118-4-ams.github.com (140.82.118.4): icmp_seq=2 ttl=49 time=71.2 ms

--- github.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 71.071/71.182/71.294/0.289 ms
parent finished
emre@emre-VirtualBox:~/Belgeler/workspace$
```

```python
#EmreTaşkın-170709060

import os
import requests
import uuid


child = os.fork()
if child == 0:
        os.system("ping -c 4 google.ca")
        print("child finished")
        os._exit(0)
os.wait()
os.system("ping -c 2 github.com")
print("parent finished")
```

2.4 Assignment 4 Control duplicate files within the downloaded files of my python code

In this assignment I used multi processing technique and hashlib library of python. My function is hashing the files and output the results. We can see the duplication from the results. In my output I saw the same hash codes and colored them.


Aç
deneme.py
~/Belgeler/workspace
Kaydet
emre@emre-VirtualBox: ~/Belgeler/workspace
Dosya Düzenle Görünüm Ara Uçbirim Yardım

```python
#!/usr/bin/python
import os
from multiprocessing import Pool
import hashlib

def hashing(filename):
        hasher = hashlib.md5()
        with open(filename, 'rb') as afile:
                buf = afile.read()
                hasher.update(buf)
                print(hasher.hexdigest())

with Pool(2) as p:
        print(p.map(hashing, ['4fbe5b7a-4e20-417a-a85b-
b7b7a52b997d' ,'943f3cf3-2de3-41d2-89f6-8e2f4a176d92', 'afbc8728-9f43-41b0-85f4-ee18b316c02c',
'bb6c5c19-4fde-4c61-ae16-fe093cba6f87', 'ddc53410-7a6d-4d84-8b03-064de2176bd5']))
```

```
emre@emre-VirtualBox:~/Belgeler/workspace$ python3 deneme.py
545550f63a21e726604915f84e63dec9 ●
c16e25807a7211f28b49fbe54f6390ff
c8ac40dc6b37096d61c34c9a50a794b5 ●
545550f63a21e726604915f84e63dec9 ●
c8ac40dc6b37096d61c34c9a50a794b5 ●
[None, None, None, None, None]
emre@emre-VirtualBox:~/Belgeler/workspace$ 
```

3 Results

Fork system call use for creates a new process, which is called child process, which runs concurrently with process (which process called system call fork) and this process is called parent process. If parent process does not wait the child process this situation causes the orphan process. To avoid from the orphan process we can use os.wait(). Hashlib library implements a common interface to many different secure hash and message digest algorithms. It was really efficient to control duplicate files.

4 Conclusion

In conclusion, we can create child process with fork system call. While using child process we should check the child processes and parent processes times because if parent process takes shorter time than child this means there is orphan process situation in your code. We should fix it. Im happy to learn hashlib library, it was really fast and good way to check duplicates by using multiprocessing with hashlib. This lab improved my virtual machine skills and also I feel more confident about how operating systems works.