

## Assignment #4

### just group it!

#### 1. seeing cgroups

##### (a) visual #1

For this visual to be understood, follow the following steps ...

- (i) Create a new CPU cgroup
- (ii) Add a **CPU intensive** process to the new group
- (iii) Change its period, shares and quota
- (iv) Measure the process behavior (cgroups stats and execution time)  
You can also get some metrics from the `cpu.stat` cgroups file
- (v) Repeat these steps from step (ii) with different settings.
- (vi) Report findings via a table or graphs as an experiment — question, hypothesis, data representation, observations.

##### (b) visual #2

- (i) Repeat 1a with two CPU cgroups with at least one process each.
- (ii) Configure each cgroup to use differentiated CPU resources (via one or more of share, quota, period etc.)
- (iii) Measure the behavior of each process
- (iv) Repeat these steps from step (ii) with different settings.
- (v) Report findings via a table or graphs as an experiment — question, hypothesis, data representation, observations.

#### 2. namespaces or spaces with names?

##### (a) Recreate and understand the two examples described [here](#).

The link describes two programs/techniques to use the system call interface to work with namespaces — one is to create a new process in a new namespace and the other is to add a process to an existing namespace.

##### (b) Repeat the above process using the command line tools `unshare` and `nsenter` to produce the same effect.

Demonstrate the correctness via shell tools (`ps`, `ls` etc.)

##### (c) Modify the program in part (a) to accept as argument the PID of a process whose namespace a new process will join via `setns`.

Hint: The first parameter of `setns` can be a `pidfd` or `fd` of a namespace.

#### 3. 1+2 = 3

##### (i) Create a PID namespace and add 5 tasks to it, with its own root directory.

- Will need to write a program that creates 5 child processes — the corresponding programs of these process are stored in a directory relative to its root directory.

You can use the same program and its variations from part 1.

- Execute the main program with `chroot` to enable the correct execution.
- Add the main program to a separate PID namespace (via `clone` or `setns`)

##### (ii) Add the processes of the new namespace to a new/separate cgroup.

- Change cgroup configuration parameters to observe changes in behavior of the processes.

- Report findings of change in resource settings.

## Submission

Submit code files, data and outputs for each part.

Create a folder <your-roll-no>-CS695-a4 and add the above in three directories one for each part.

Create a .tar.gz of the folder and submit a single <your-roll-no>-CS695-a4.tar.gz file in Moodle. You may use the following command to create the tarball.

```
tar -czvf <your-roll-no>-CS695-a4.tar.gz <your-roll-no>-CS695-a4/
```

**Due date: 7th Apr. 5pm**