WHAT IS FORK? HOW PROCESSES ARE CREATED USING FORK

A process executes the below code:

```
fork();

fork();

fork();
```

How many child processes will be created? What is the meaning

of fork command?

Solution:

When a process forks (or executes a fork command), it creates a copy of itself (a new process is created). The thread of execution is duplicated, creating a child process from the parent process.

```
int main()

{

    printf("1...\n");

    fork();
```

```
    printf("2...\n");
    fork();
    printf("3...\n");
}
```

The first process will print

```
1...
```

then there will be 2 processes and both will start executing from that point (i.e line number 4), hence there will be 2 processes printing 2 (executing the printf on line 5),

```
2...
2...
```

Then, both the processes will create one child process each, hence creating total 4 processes. And each of these 4 processes will print “3…” hence there will be

```
3...
3...
```

```
3...

3...
```

Hence the complete output will be

```
1...

2...

2...

3...

3...

3...

3...
```

The important point to understand is that the child process starts its execution from the point where it is created (and not from the start of the program).

At the end there will be 4 processes running.

Return value of the fork is different for the parent (process creating) and child (process created). And hence the 2 processes can take decisions by observing the return value of fork command. Return value of fork will be

-1: If unable to create a child process.

0: 0 is returned to the child process (if it is created).

Process-ID of newly created process: Process-ID of the newly created process (child process) will be returned to the parent process (In case parent need it for anything)

Separate address space for the child will be created with exact copy of all the memory segments from the parent process. Hence, if the child process want to know the process id of parent process then the parent process can store its id in a variable before forking a process. Hence both the processes can have process-ids of each other in case they want to interact with each other thru inter process communication.

For Example:

```
int main()
{
   printf("Start...\n");
   pid_t pid = fork();

   if(pid == -1)
```

```
        printf("Unable to create child process");
    else if(pid == 0)
        printf("Inside Child Process");
    else
        printf("Inside Parent");
}
```

If process creation is successful, then Parent process will output:

Start…

Inside Parent

and child process will output:

Inside Child Process

Separate address space for the child will be created with exact copy of all the memory segments from the parent process.

Source: http://www.ritambhara.in/what-is-fork-how-are-processes-created-using-fork/#more-36