Project 1: NPShell

NP TA

3/27 23:55

Project 1 Deadline

Demo: 3/30 Sat.

General Info

- We will announce our Bitbucket organization.
- We will announce the nplinux account.
    - NP projects should run on NP servers.
    - Any abuse of NP server will be recorded.
    - Don't leave any zombie processes in the system.

Project 1: Info

- You are HIGHLY encouraged to publish your questions on Project 1 討論區
- You can contact TAs by e3. (Mails sent to other addresses will NOT be replied)
- TA hours (Tuesday: 15:00 ~ 17:00) on 3/19, 3/26 will be held at online.
- You MUST make a reservation by email in advance.
- TAs will NOT debug for you.

Project 1: Submission

- Create a directory named as your student ID, put all files into the directory.
- You **MUST** use GNU Make to build your project and compile your source code into one executable named npshell. The executable and Makefile should be placed at the top layer of the directory. We will use this executable for demo.
- You are NOT allowed to demo if we are unable to compile your project with a single make command.
- Upload only your code and Makefile. Do NOT upload anything else (e.g. noop, removetag, test.html, .git, __MACOSX)
- zip the directory and upload the .zip file to the E3 platform.
  ATTENTION! We only accept .zip format

4. **zip** the directory and upload the .zip file to the E3 platform

   ATTENTION! We only accept .zip format

   e.g.

   Create a directory 0856053, the directory structure may be:

   ```
   0856053
    ├──── Makefile
    ├──── shell.cpp
    └──── shell.h
   ```

   zip the folder 0856053 into 0856053.zip and upload 0856053.zip onto E3

G. We take plagiarism seriously.

All projects will be checked by a cutting-edge plagiarism detector.

You will get zero points on this project for plagiarism.

Please don't copy-paste any code from the internet, this may be considered plagiarism as well.

Protect your code from being stolen.

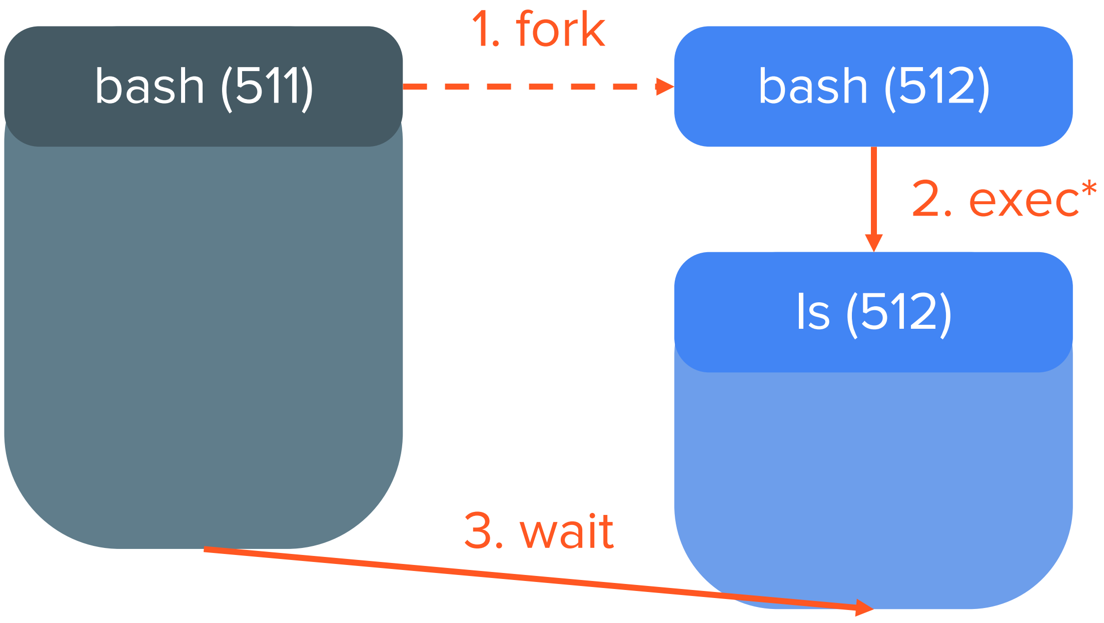
Project 1: Demo

- 3/30 Sat. 09:30 ~ 14:10, 17:30 ~ 22:00.
- We will announce demo slots 2 ~ 3 days before.
- Tasks:
  - (correct format and compile).
  - QA.
  - Pass np-basic test cases.
  - Pass np-hard test cases.
  - Implement 1 or 2 extra functions with limited time.

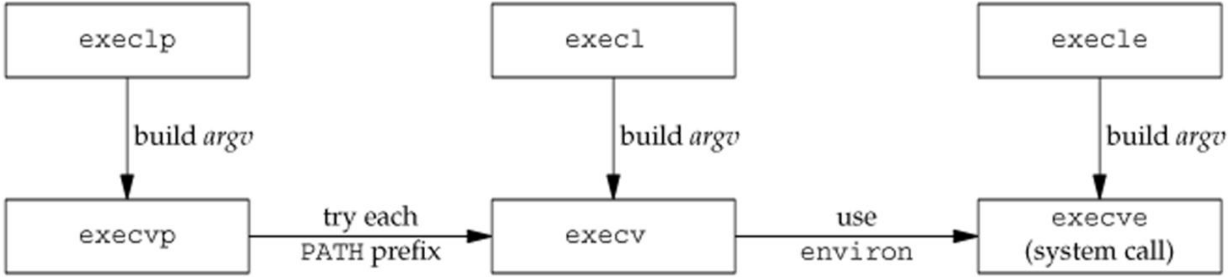
Implementation

Process Lifecycle: fork-exec-wait

1. Creation: fork
2. Execution exec*
3. Termination: waitpid

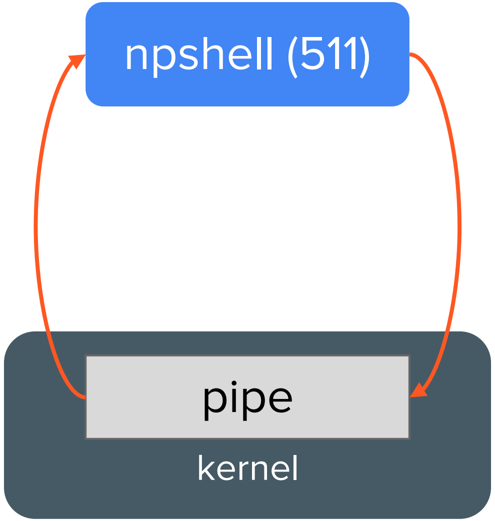


exec*

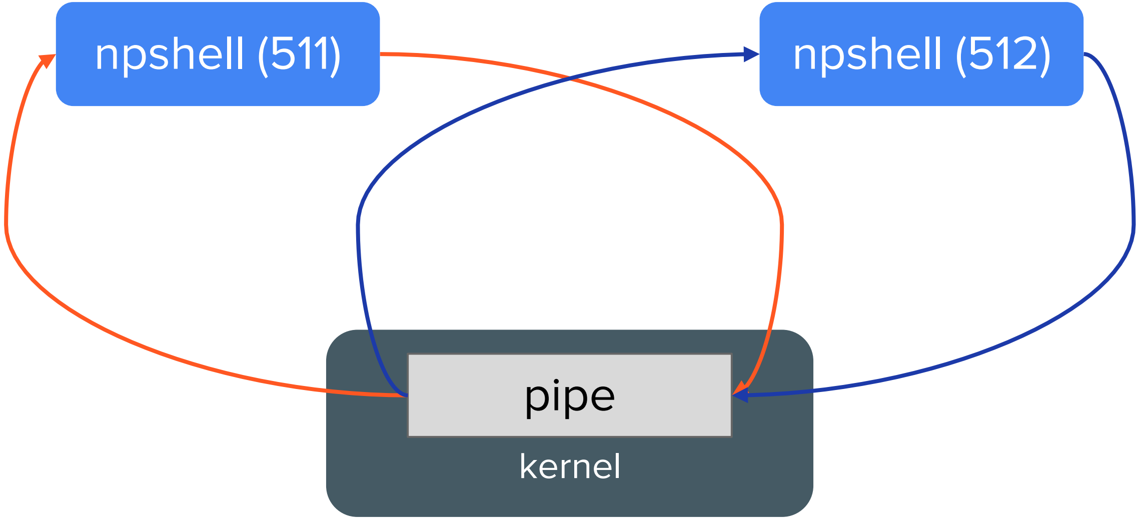

execlp
execl
execle
build *argv*
build *argv*
build *argv*
execvp
try each
PATH prefix
execv
use
environ
execve
(system call)

Pipe

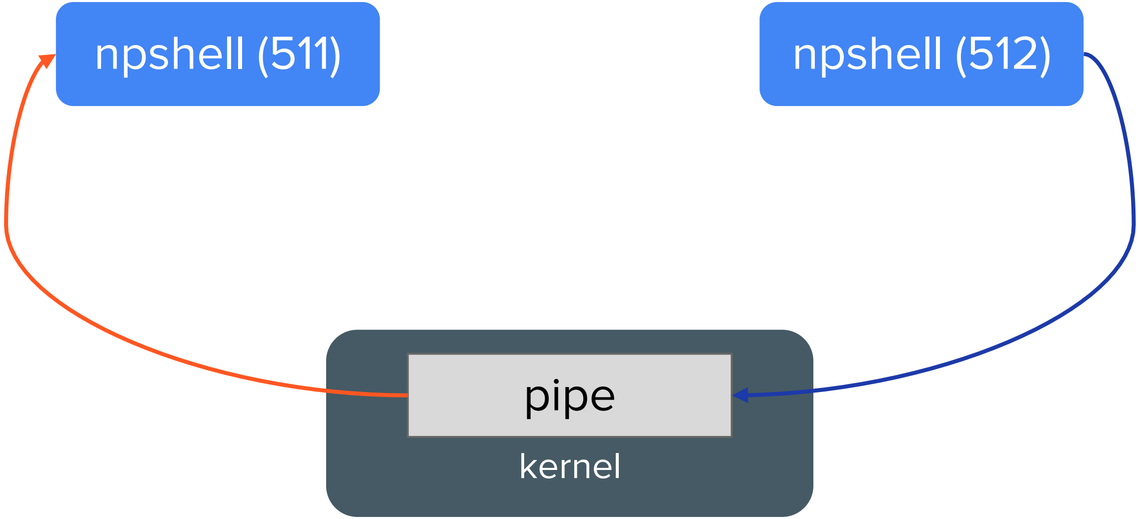
0. pipe



1. fork

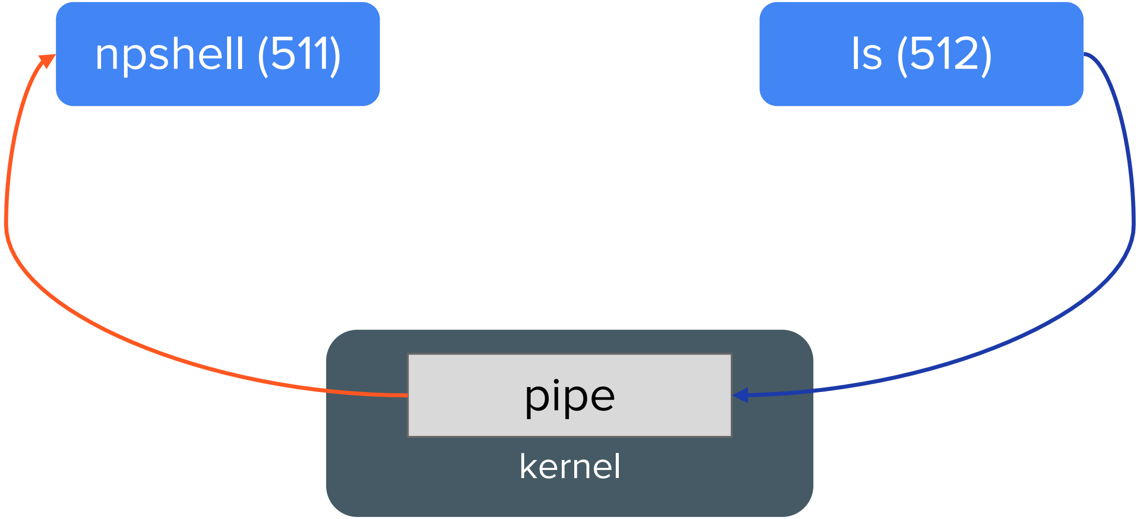


2. close



(because pipe is half-duplex)

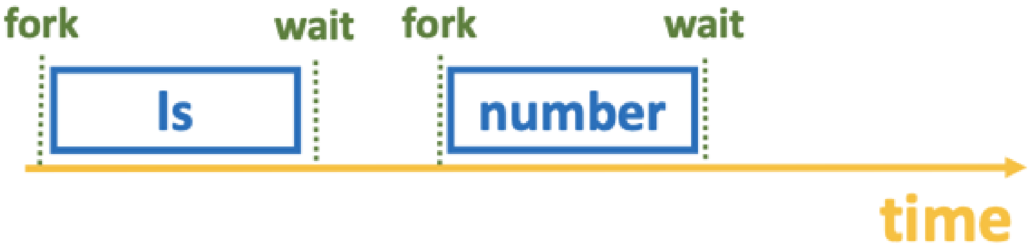
3. exec



Issues

Impl 1 : Wait for each child

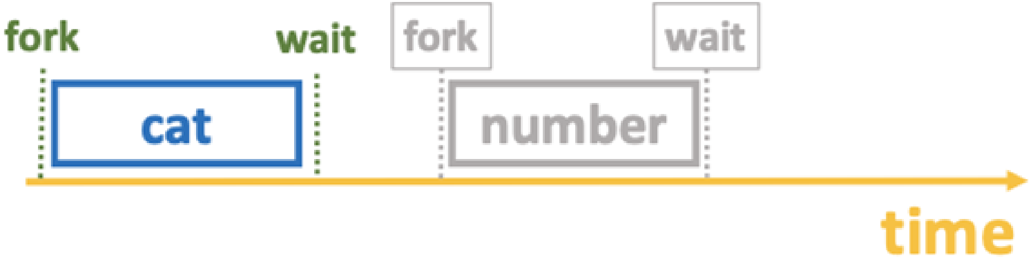
% ls | number



Problem 1 : Unable to process large data

```
% cat largeFile.txt | number
```

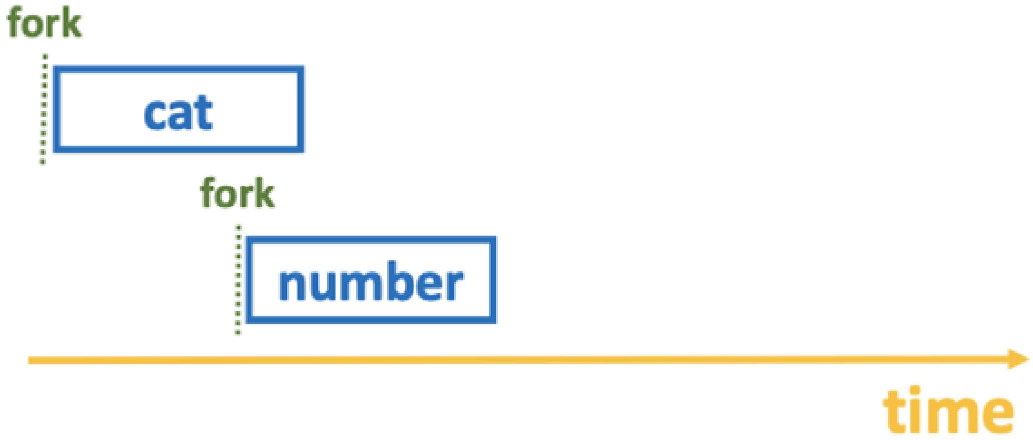
```
% cat largeFile.txt |1
% number
```



The process will hang forever !

Impl 2 : Don’t wait for processes

```
% cat largeFile.txt | number
```



fork

cat

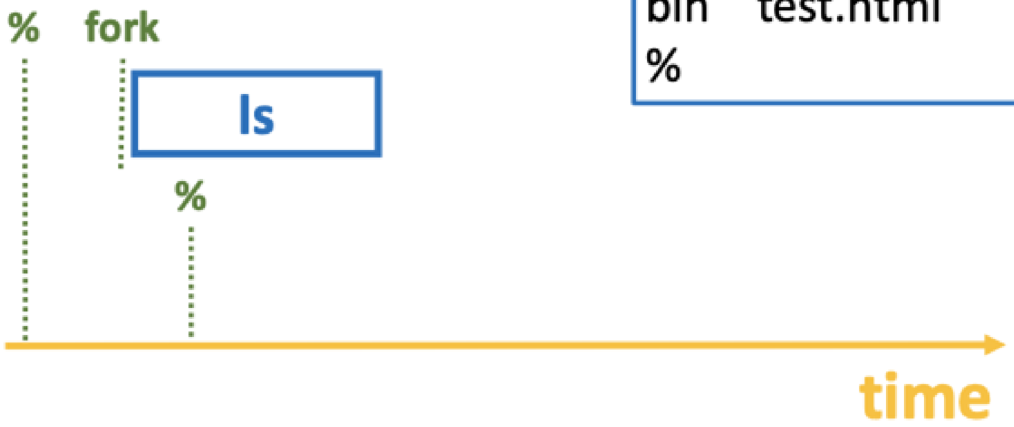
fork

number

time

Problem 2 : % ordering

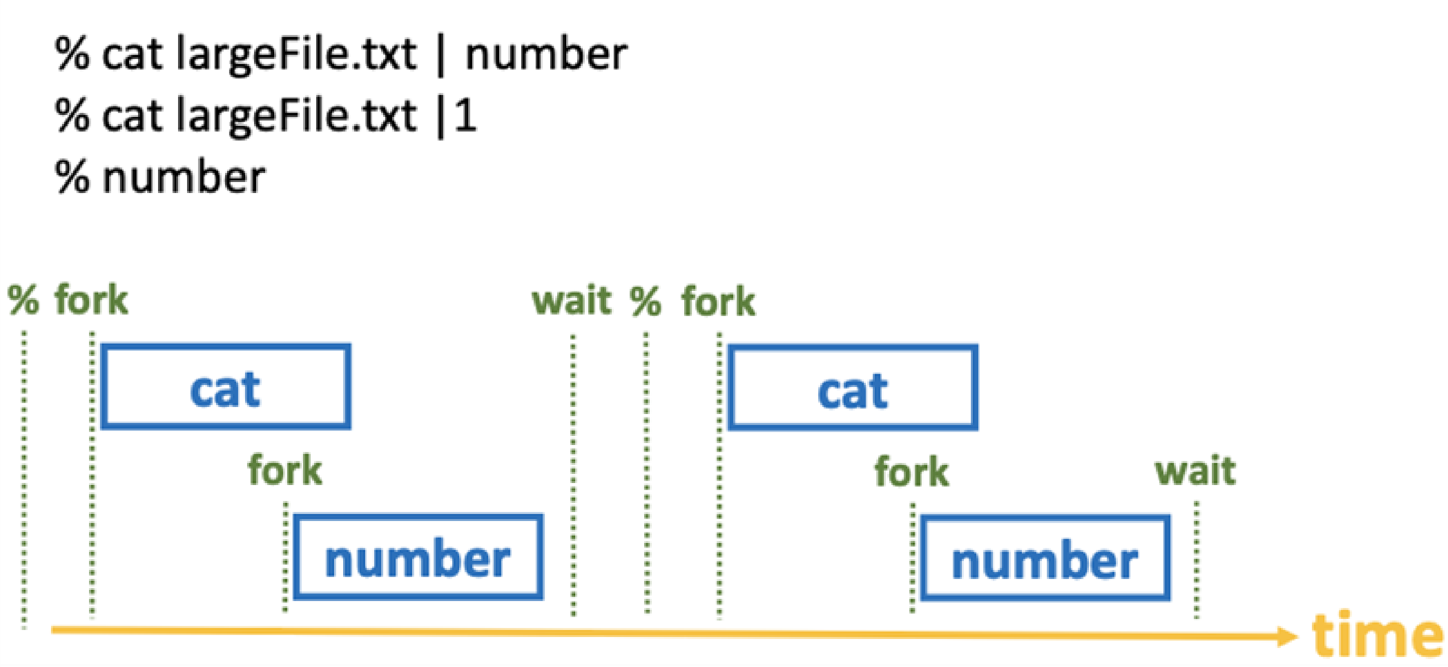
```
% ls
% bin    test.html
```



Correct:

```
% ls
bin    test.html
%
```

Impl 3 : Wait for a line if not pipeN



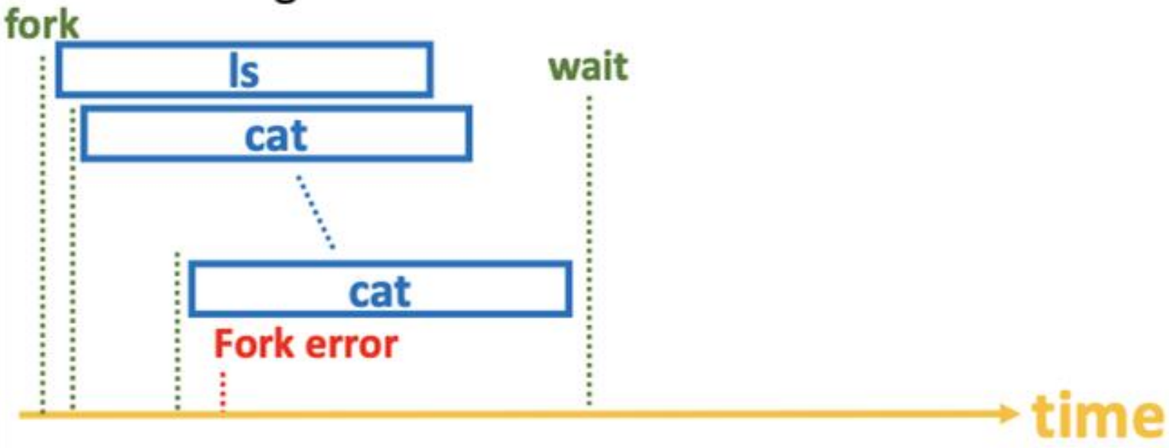
```
% cat largeFile.txt | number
% cat largeFile.txt |1
% number
```

Problem 4 : Process limitation

ls | cat | cat | cat | cat .... cat | cat

Process limit is 512 on NP server.

Fork error might occur around the 510th cat.



Hint

- Functions you may use
  - fork
  - pipe
  - dup, dup2
  - exec*
  - wait, waitpid
- Handle failure
  - See man page for more information
- Debug
  - gdb
  - lsof