운영체제 3차 Coding 과제: AdvMysh2

201420891 유동균 (소프트웨어학과)

구현 목록 및 진행 사항

운영체제 3차 과제는 2차 과제인 mysh2에 추가 고급 기능을 구현하는 것이다. 기존 2차 과제의 코드에 덧붙혀 parent-child IPC를 위한 Pipelining, job control을 위한 signal processing, Background processing 기능을 구현하여 과제 문서에 주어진 test case를 통과할 수 있도록 했다.

과제의 요구 사항은 아래와 같으며, 구현 여부와 구현 방법은 [Table 1]과 같다.

[Table 1] Assignment coverage

Requirements	Coverage	Implementation	Note
Pipe	100%+	System calls(dup, dup2), Unix socket, pthread	**Implemented** **multiple pipe**
Signal handling	100%	System calls(signal, sigact)	Implemented internal kill command
Background processing	100%	System calls(tcsetpgrp…), pthread	Implemented internal fg command
Path resolution	100%	System calls(getenv…)	**Submitted in** **last assignment**

Pipe의 경우, 초기부터 multiple pipe 구현을 목표로 두고 있었다. 그래서 socket file descriptor를 넘길 수 있는 procedure를 개발했고, 이를 생성할 때 다양한 방법 중에서도 Unix socket을 사용하여 socketpair를 구현하도록 했다.

unix socket을 통해 구현한 socketpair를 생성하고, 생성된 socketpair의 descriptor들을 필요한 곳에서 불러올 수 있는 function을 작성했다. socketpair library는 universal하게 사용할 수 있도록 modularization하게 구현했다.

해당 socketpair의 양 단을 stdout/stdin으로 활용하고, system call인 dup2를 사용하여 해당 스트림을 stdin/stdout에 copy했고, 마지막에 사용된 socketpair들을 close함으로써 stdout/in이 정상적으로 terminal로 돌아오게 구현했다.

해당 내용은 코드의 주석으로 작성하여 자세한 설명을 대체한다.

Signal handling은 간단하였다. Ctrl+C시 signaling되는 SIGINT와 Ctrl+Z시 signaling되는 SIGTSTP를 무시하도록 구현하고, 마지막으로 mysh process가 종료 될 시 zombie alert를 내는 것이었다. signal system call을 사용하여 SIGINT, SIGTSTP에 대한 catch-up을 ignoring하는 방법으로 진행하였다.

초기에는, SIGCHLD를 ignoring하여 별도의 wait이나 waitpid system call을 통해 직접 child process dispose를 할 필요가 없다고 생각했으나, 이러한 방식의 programming은 code에 잠재적인 오류가 있을 것이라 생각했다. 예를 들어, server programming의 경우 이와 같이 처리할 경우 여러 SIGCHLD가 들어온 경우에 child가 정상적으로 dispose가 되지 않는 문제점이 발생한다.

이 문제점을 해결하기 위해 signal flag인 SA_NODEFER, SA_NOCLDWAIT와 sigaction을 사용하여 구현했었으나, 이는 나중에 아무 생각 없이 재사용될 수 있는 불완전한 code 문제를 해소하기 위해 모두 synchronous한 code로 변경하였다. (coding convention)

zombie alert의 경우 exit을 호출하여 종료 될 시 synchronous하게 구현하였고, internal command인 kill command 또한 kill system call을 통해 구현하였다.

Background processing을 구현하는 것은 process group에 대해 이해할 수 있는 좋은 기회가 되었다. process group이 foreground process group에서 detach될 경우(setpgid을 이용한 분리), 해당 pg는 더 이상 stdin/out을 할 수 없고 Signal을 통한 제어를 해야 한다. 해당 기능을 구현하기 위해 SIGTTIN과 SIGTTOU를 controlling해야 하며, signal을 normal하게 처리하는 SIG_DFL 플래그와 tcsetpgrp를 통해 background processing을 구현하였다.

mysh의 internal command인 fg command도 구현했다. 이는 SIGTTOU를 normal하게 처리하도록 하는 SIG_DFL, 다시 ignoring하는 SIG_IGN, stdin을 가져오는 tcsetpgrp를 통해 구현할 수 있었다. SIGTTOU를 ignoring할 경우 제어권을 넘겨 받지 못해 stdout이 출력되지 않는데, default로 변경하면 child process의 stdout이 redirection된다. waitpid를 통해 작업을 기다리고, 작업이 완료되면 SIGTTOU를 다시 ignoring하는 방법으로 background processing 구현을 완료했다.

이번 개발 시 고려했던 사항은 최대한 전역 변수를 사용하지 않으면서, 유기성 있는 간결한 코드로 작성하는 것이었다. callback 사용을 최소화하면서, 누가 보더라도 따라가기만 하면 쉽게 이해할 수 있는 코드를 작성해보려 노력했다.

그 결과로 individual socket생성을 위한 전역 변수, background process가 이미 실행되고 있는지에 대한 전역 변수, pthread내 static 변수로 총 4개의 전역 변수를 사용하여 AdvMysh2 구현에 성공하였다.

이 전역변수들은 차후 mutex를 통해 공유 위반 제어를 해야 할 것이다.

배우게 된 점

pthread 라이브러리를 통해 생성된 thread가 종료될 때, thread내에서 생성된 local variable들이 모두 n+1 scope or n+2 procedure line 등 일정 범위에 도달하면 GC되는 것을 알게 되었다. 이것이 socketpair의 구현에 지대한 영향을 미치기도 했다. 해결 방법은 pthread내의 variable을 static으로 정의해주어 compile time의 변수로 만들어 주는 것으로 해결했다.

pointer를 이용한 value pass(call-by-reference)에 대한 이해는 저번 과제에서 익혀 진행에 문제가 없었지만, 정말 특이한 점을 하나 발견하였다. pthread function들에 대한 wrapper를 작성하여 운영하였는데, macOS에서는 value passing이 잘 되어 구현에 문제가 없는 반면 arch linux에서는 전혀 작동하지 않는 문제가 발생했다. (문제의 commit tag: AdvMysh2_macOS)

macOS에서는 pthread 라이브러리를 사용할 경우 valgrind가 항상 오류를 발생시킨다는 점을 알게 되었다. 그러하여 valgrind를 통한 memory leak 해결은 arch linux에서 진행하여 해결했다.

피드백

3차 과제는 지난 1차 과제와 2차 과제에 비해 운영체제 강의에서 배웠던 내용이 과제에 많은 부분 반영되었다. 비록 중간고사 및 다른 일 때문에 많이 바빴지만, delay token 2개를 사용하여 과제를 마무리 할 수 있었고 과제의 난이도는 충분하다고 생각한다.

최대한 shell처럼 구현하려고 했는데, multi-background process의 경우 thread_pool을 만들기에 어렵기도 하고 내게 주어진 시간이 짧기도 해서 구현하지 못한 점이 정말 아쉽다. 그래도 구현한 부분에서 대부분은 사용하고 있던 zsh와 유사하게 제작해 보았고, 흥미로웠다.

고품질의 과제로 프로그래밍에 흥미를 돋게 해주셔서 고맙습니다. -동균

Bitbucket repository: https://bitbucket.org/sokdakino/mysh2

Appendix. valgrind with implemented AdvMysh2 in archlinux

```
sokdak@sokdak ~/dev/mysh2 ubuntu valgrind --leak-check=full --show-leak-kinds=all  ./mysh
==19403== Memcheck, a memory error detector
==19403== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==19403== Using Valgrind-3.14.0.GIT and LibVEX; rerun with -h for copyright info
==19403== Command: ./mysh
==19403==
ls
Makefile  WDc.1        answer2.txt  answer4.err  answer5.txt  answer7.err  docs     lib   sample1.input
WDc       answer1.txt  answer3.txt  answer4.txt  answer6.txt  answer7.txt  include  mysh  sample2.input
ls &
Makefile  WDc.1        answer2.txt  answer4.err  answer5.txt  answer7.err  docs     lib   sample1.input
WDc       answer1.txt  answer3.txt  answer4.txt  answer6.txt  answer7.txt  include  mysh  sample2.input
[1] 19405
[1]  + 19405 done       ls
pwd &
[1] 19407
/home/sokdak/dev/mysh2
==19407==
==19407== HEAP SUMMARY:
==19407==     in use at exit: 0 bytes in 0 blocks
==19407==   total heap usage: 37 allocs, 37 frees, 13,079 bytes allocated
==19407==
==19407== All heap blocks were freed -- no leaks are possible
==19407==
==19407== For counts of detected and suppressed errors, rerun with: -v
==19407== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
[1]  + 19407 done       pwd
wget ka.do/WDc &
[1] 19410

Redirecting output to 'wget-log.2'.
fg 19410
--2018-11-04 03:20:55--  http://ka.do/WDc
Resolving ka.do (ka.do)... 222.122.205.147
Connecting to ka.do (ka.do)|222.122.205.147|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/snapshot/linux-4.19-rc7.ta
--2018-11-04 03:20:56--  https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/snapshot/li
Loaded CA certificate '/etc/ssl/certs/ca-certificates.crt'
Resolving git.kernel.org (git.kernel.org)... 147.75.46.191, 2604:1380:4080:c00::1
Connecting to git.kernel.org (git.kernel.org)|147.75.46.191|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/x-gzip]
Saving to: 'WDc.2'

WDc.2                                                       [           <=>
C[1]  + 19410 interrupted       wget ka.do/WDc
ls | grep s | grep a | grep m | grep p | grep l | grep e | grep 3
sample3.input
```

```
Redirecting output to 'wget-log.2'.
fg 19410
--2018-11-04 03:20:55--  http://ka.do/WDc
Resolving ka.do (ka.do)... 222.122.205.147
Connecting to ka.do (ka.do)|222.122.205.147|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/snapshot/li
--2018-11-04 03:20:56--  https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.g
Loaded CA certificate '/etc/ssl/certs/ca-certificates.crt'
Resolving git.kernel.org (git.kernel.org)... 147.75.46.191, 2604:1380:4080:c00::1
Connecting to git.kernel.org (git.kernel.org)|147.75.46.191|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/x-gzip]
Saving to: 'WDc.2'

WDc.2                                                       [           <=>
C[1]  + 19410 interrupted       wget ka.do/WDc
ls | grep s | grep a | grep m | grep p | grep l | grep e | grep 3
sample3.input
exit
==19403==
==19403== HEAP SUMMARY:
==19403==     in use at exit: 0 bytes in 0 blocks
==19403==   total heap usage: 227 allocs, 227 frees, 69,366 bytes allocated
==19403==
==19403== All heap blocks were freed -- no leaks are possible
==19403==
==19403== For counts of detected and suppressed errors, rerun with: -v
==19403== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
 sokdak@sokdak  ~/dev/mysh2  ubuntu
```

```
 sokdak@sokdak  ~/dev/mysh2  ubuntu  valgrind --leak-check=full --show-leak-kinds=all  ./mysh
==20301== Memcheck, a memory error detector
==20301== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==20301== Using Valgrind-3.14.0.GIT and LibVEX; rerun with -h for copyright info
==20301== Command: ./mysh
==20301==
wget ka.do/WDc &
[1] 20303

Redirecting output to 'wget-log.2'.
kill 20303
[1]  + 20303 killed     wget ka.do/WDc
exit
==20301==
==20301== HEAP SUMMARY:
==20301==     in use at exit: 0 bytes in 0 blocks
==20301==   total heap usage: 34 allocs, 34 frees, 7,914 bytes allocated
==20301==
==20301== All heap blocks were freed -- no leaks are possible
==20301==
==20301== For counts of detected and suppressed errors, rerun with: -v
==20301== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
 sokdak@sokdak  ~/dev/mysh2  ubuntu
```