

Systems Programming (Fall, 2021)

Assignment 1 (Due on 10/6)

1.

(a)關於分析 `./a.out < infile 2>&l >outfile` 這個命令，`./a.out`這個是執行a.out這個程式(應該是gcc compile c 程式來的)，`< infile` 這個是把stdin指向open file table的entry指向infile在open file table的entry，因為stdin是./a.out的輸入，所以infile的內容會成為a.out這個程式的輸入。接著`2>&l`是代表把stderr指到stdout在open file table的entry上，也就是終端輸出，之後的`>outfile`也是`l>outfile`，會把stdout指向outfile在open file table的entry。但stderr還是指向stdout(open file table原本stdout指向的地方)，不是指向open file table上outfile指向的地方，所以錯誤訊息不會輸入outfile。

(b)

```
#include <unistd.h>

int main(int argc, char *argv[])
{
    int  fd1, fd2;
```

```

    fd1=open(infile, O_RDONLY);

    fd2=open(outfile, O_WRONLY|O_CREAT, 0666);

    dup2(fd1, 0);

    close(fd1);

    dup2(1, 2);

    dup2(fd2, 1);

    close(fd2);

    execlp( "./a.out" , "./a.out" , (char*)0);

    return 0;

}

```

2.

- (a) `write_to_fd()`需要是atomic operation，因為輸入是fd，如果在這個function外我們用 `int fd1 = dup(fd)`或用 `fork()`產生一個child，就可以使fd1和fd指向同一個open file table 的entry。這樣如果在執行write前發生 context switch，可能因為其他process操作了fd1導致 switch回去時fd跟原fd不一樣，這樣就會出問題。
- (b) `write_to_fn()`不需要是atomic operation。因為fd是在函



式中open的，函式中也沒有fork或dup，所以不會有其他的fd跟這個fd指向同一個open file table 的entry，所以發生context switch後，不會有其他程式對這個entry進行修改，所以不會有問題。

簽名： 許庭瑋

Signature Certificate

Document Ref.: 7RS6Y-6GDRK-KT8KZ-YSBP2

Document signed by:

	<p>許庭瑋</p> <p>Verified E-mail: b09902140@ntu.edu.tw</p>	<p>許庭瑋</p> 
IP: 140.112.16.134	Date: 13 Oct 2021 04:27:38 UTC	

Document completed by all parties on:
13 Oct 2021 04:27:38 UTC

Page 1 of 1



Signed with PandaDoc.com

PandaDoc is a document workflow and certified eSignature solution trusted by 25,000+ companies worldwide.

