System programming MP4 report

B03902071 葉奕廷

1. 假設我們可以知道每個 client 在 upload file 的哪一部分，就能使用 fcntl()去 lock 該 file 特定的區域而不是去 lock 整個 file。如果不行的話就如這次作業我們必須 lock 整個 file。

   當一個 client 試著去 upload 一個 file 時若 server 端可以成功 lock file(用 fcntl(fd , GETLK)檢查)，則 lock file 後上傳。若 server 發現該 file 已經先被 lock 住了則先上傳到一個備份檔後並回傳一個”Conflict”的 message 回去給該 client。當先 lock file 的 client 上傳結束後 server 端多去檢查是否存在這樣的一個備份檔，如果存在的話則 fork 一個 process 並在該 process 用 exec()去 call file_merger(如果有函式庫的話也可以直接 call file_merger 的 function)來 merge 該 file 和備份檔(可能需要調整一下 file 的名稱)，merge 完後同樣回傳”Conflict”的 message 回去給 client。Client 在收到”Conflict”的 message 後可以自行決定該如何處理 conflict 並上傳新版本的 file。
2. 在 server 端，每次收到一個 client(conn_fd)的 request，就 fork 出一個 process 去 handle request，並先暫時在 select 部分無視該 conn_fd 以免多個 process 同時處理同一個 conn_fd 會造成奇怪的事情。Fork 出的 child process 在處理完一個 request 就 return 回去。

   Process 和 Thread 兩個實作方法最大的不同就在他們 memory 的處理方法，fork 出來的 child process 會有它自己的 process memory space 而各個 Thread 只是在一個 process memory space 的 stack 部分有自己的 frame 而已。因為每個 fork 出來的 child process 都要先複製 parent process 除了 text segment 外的 memory space(儘管是 copy-on-write)，使用 fork 來實作的 delay time 會大於以 Thread 來實作。

   在穩定性的部分來說，用 fork 來實做的話 fork 出來的 child process 如果在 handle request 的時候掛掉，其不會影響到整個 server 的運作。如果是用 Thread 實作，若有一個 thread 在 handle request 時掛掉，會是整個 server 一起掛掉，所以當寫出來的 code 不太穩定時可能用 fork 來實作會比較好。