Wydział Informatyki Politechniki Białostockiej	Data: 9.V.2022 Przedmiot: Systemy Operacyjne
Projekt SO Piotr Zalewski, Maurycy Zdanowicz, Jakub Mirończuk	Prowadzący: dr inż. Wojciech Kwedlo

Treść Zadania

Program który otrzymuje co najmniej dwa argumenty: ścieżkę źródłową oraz ścieżkę docelową . Jeżeli któraś ze ścieżek nie jest katalogiem program powraca natychmiast z komunikatem błędu. W przeciwnym wypadku staje się demonem. Demon wykonuje następujące czynności: śpi przez pięć minut (czas spania można zmieniać przy pomocy dodatkowego opcjonalnego argumentu), po czym po obudzeniu się porównuje katalog źródłowy z katalogiem docelowym. Pozycje które nie są zwykłymi plikami są ignorowane (np. katalogi i dowiązania symboliczne). Jeżeli demon (a) napotka na nowy plik w katalogu źródłowym, i tego pliku brak w katalogu docelowym lub (b) plik w katalogu źrodłowym ma późniejszą datę ostatniej modyfikacji demon wykonuje kopię pliku z katalogu źródłowego do katalogu docelowego - ustawiając w katalogu docelowym datę modyfikacji tak aby przy kolejnym obudzeniu nie trzeba było wykonać kopii (chyba ze plik w katalogu źródłowym zostanie ponownie zmieniony). Jeżeli zaś odnajdzie plik w katalogu docelowym, którego nie ma w katalogu źródłowym to usuwa ten plik z katalogu docelowego. Możliwe jest również natychmiastowe obudzenie się demona poprzez wysłanie mu sygnału SIGUSR1. Wyczerpująca informacja o każdej akcji typu uśpienie/obudzenie się demona (naturalne lub w wyniku sygnału), wykonanie kopii lub usunięcie pliku jest przesłana do logu systemowego. Informacja ta powinna zawierać aktualną datę. [12p.]

a) [10p.]

Dodatkowa opcja -R pozwalająca na rekurencyjną synchronizację katalogów (teraz pozycje będące katalogami nie są ignorowane). W szczególności jeżeli demon stwierdzi w katalogu docelowym podkatalog którego brak w katalogu źródłowym powinien usunąć go wraz z zawartością.

b) [12p.]

W zależności od rozmiaru plików dla małych plików wykonywane jest kopiowanie przy pomocy read/write a w przypadku dużych przy pomocy mmap/write (plik źródłowy) zostaje zamapowany w całości w pamięci. Próg dzielący pliki małe od dużych może być przekazywany jako opcjonalny argument.

Uwagi

(a)

Wszelkie operacje na plikach należy wykonywać przy pomocy API Linuksa a nie standardowej biblioteki języka C

(b)

kopiowanie za każdym obudzeniem całego drzewa katalogów zostanie potraktowane jako poważny błąd

(c)

podobnie jak przerzucenie części zadań na shell systemowy (funkcja system).

Zrealizowane Punkty

Zrealizowano punkty a) oraz b)

Opis Najistotnijszych algorytmów

Jedynym algorytmem obecnym w projekcie jest rekurencyjne kopiowanie plików z katalogu do katalogu oraz rekurencyjne usuwanie katalogu. Algorytm zaimplementowany jest w funkcji *NativeSync*(...).

```
static void NativeSync(
    const char* srcPath,
    const char* dstPath,
    const uint64_t threshold,
    const bool recursive
);

static void RecursiveRemove(const char* dirPath);
```

W danym wywołaniu pobierana jest lista plików, z podkatalogów katalogu/katalogów źródłowego i docelowego po czym w zależności od dat modyfikacji i istnienia plików następuje wykonanie kopii z zachowaniem metadanych tj. data modyfikacji i praw dostępu. W razie gdy dany katalog/plik nie istnieje w katalogu źródłowym uruchamiany jest *RecursiveRemove*(...).

Opis najważniejszych modułów

Kod źródłowy można podzielić w następujący sposób.

Parsowanie argumentów przekazanych do programu, wyświetlanie pomocy

- Command.h/c
- ParseCommand.h/c

Operacje na plikach tj. listowanie, kopia

- FileSystem.h/c
- FileInfo.h/c

Uruchomienie demona, log systemowy

- Daemon.h/c
- Log.h/c

Główny algorytm oraz pętla programu

- Synchronize.h/c
- main.c

Funkcje pomocnicze

- Utils.h/c

Instrukcja obsługi

Instrukcja zostanie wyświetlona po uruchomieniu programu w następujący sposób

```
$ ./build/src/CpSync -h
```