TIN Zadanie 2020Z

Początek realizacji: 03. 11 . 2020 Projekt wstępny: 26 . 11 . 2020; Koniec realizacji: . 07. 01. 2021
punktacja: 0 - 50 p.; Zespoły: 4 os

Treść: Napisać program obsługujący uproszczoną wersję protokołu NFS (Network File System). Założenia:

- Należy zaimplementować serwer, bibliotekę kliencką (jako plik .a lub .so) oraz testowe programy klienckie realizujące funkcje operujące na zdalnych plikach (plikach zlokalizowanych na serwerze sieciowym. Zestaw funkcji do implementacji (UWAGA – wariant poniżej może wymagać implementacji dodatkowych funkcji):
  - `int` **`mynfs_open`**`(char *host, char *path, int oflag, int mode);`
    - `host` – nazwa lub adres DNS serwera
    - `path, oflag, mode` – jak w funkcji systemowej open() (dozwolone są modyfikacje lub ograniczenia)
    - zwraca: -1 gdy błąd, deskryptor pliku gdy operacja udana
    - należy zaimplementować co najmniej następujące tryby otwarcia pliku: `O_RDONLY, O_WRONLY, O_RDWR, O_CREAT`
  - `int` **`mynfs_read`**`(...);` **`mynfs_write`**`(...);` **`mynfs_lseek`**`(...);` **`mynfs_close`**`(...);` - jak odpowiedniki systemowych funkcji: read(), write(), lseek(), close()
  - `int` **`mynfs_unlink`**`(char *host, char *path);` – usunięcie pliku, parametry jak dla `mynfs_open()`
  - `mynfs_fstat(int mynfs_fd);`
    - pobiera atrybuty otwartego pliku - analogicznie do funkcji systemowej fstat();
- w razie wystapienia błędu należy kod informacyjny zapisywać w zmiennej globalnej `mynfs_error`
- Uwaga – należy przyjąć, że dozwolone jest wykonanie operacji `mynfs_open()` na katalogu, w trybie tylko do odczytu – wyłącznie w celu przekazania deskryptora do funkcji `mynfs_fstat()` (nie jest dozwolone zwykłe czytanie i inne operacje "plikowe" na katalogach)

Warianty – przypisane do zespołów

W11 – zaimplementować dodatkowo funkcje:
- `int` **`mynfs_opendir`**`( char *host, char *path);`
  - `host` – nazwa lub adres DNS serwera
  - `path` – jak funkcja opendir()
- zwraca: -1 gdy błąd, deskryptor katalogu(!) gdy operacja udana
- `char *`**`mynfs_readdir`**`(int dirfd);` `int` **`mynfs_closedir`**`(int dirfd);`
  - dir_fd – deskryptor rekordu katalogowego zwrócony przez mynfs_opendir()
- Uwaga: systemowe funkcja readdir() zwraca za każdym razem strukturę dirent zawierającą rekordy opisujące kolejne pliki z katalogu, tu wystarczy, że funkcja zwróci samą nazwę.
- `mynfs_closedir(int dirfd);` – analogicznie do closedir()

W12 – zaimplementować blokowanie dostepu do plików w trybie „jeden pisarz albo wielu czytelników” – zob. flock(2)

W13 – użyć protokołu UDP zamiast TCP, można zrezygnowac z implementacji funkcji mynfs_stat() imynfs_lseek()

W21 – implementacja serwera powinna być wielowątkowa

W22 - implementacja serwera powinna być wieloprocesowa

W31 – zaimplementować po stronie serwera możliwość udostępniania wielu niezależnych systemów plików

W32 – zaimplementować po stronie klienta możliwość wykorzystania wielu niezależnych systemów plików (tj. z różnych serwerów)

Na co zwrócić uwagę:
- "Prawdziwy" protokół NFS zaimplementowany jest przy pomocy protokołów ONC RPC/XDR warstw 5 i 6, w tej uproszczonej implementacji nie należy bazować na RPC.
- "Prawdziwe" NFS jest bezstanowe, tj. serwer nie przechowuje deskryptorów plików otwartych przez klientów,

jednak w projekcie należy zastosować rozwiązanie stanowe (jak dużo prostsze w implementacji).

- Deskryptor zwracany przez mynfs_open() nie jest oczywiście deskryptorem plikowym – takim jak zwraca open(), creat(), itd.
- Należy wyspecyfikować (w proj. wstepnym), które z informacji dot. otwartego pliku są przechowywane po stronie serwera, a które po stronie klienta i jak lokalny deskryptor zwracany przez mynfs_open() mapuje się na zdalny plik. Istotne w tym konteście atrybuty pliku to: bieżąca pozycja i tryb otwarcia.
- Jak rozwiązana zostanie kwestia praw dostępu i autoryzacji użytkowników? Do rozważenia i opisania w proj. wstępnym (wskazówka – nie są wymagane skomplikowane rozwiązania)
- Co się stanie gdy usuniemy otwarty plik? (co się dzieje gdy usuwamy otwarty plik w systemie lokalnym? – łatwo można to sprawdzić...)

Uwaga: należy starannie zaprojektować protokół. Już w sprawozdaniu wstępnym należy szczegółowo go opisać

Instrukcje dot. realizacji projektu:

Co powinien zawierać projekt wstępny (c.a. 4 strony)

Sprawozdanie musi być dostarczone mailem w formacie pdf. Ocena ze sprawozdania wstępnego zostanie wystawiona w ciągu 1-2 tygodni od dostarczenia, w przypadku oceny <= 12 p. konieczne jest omówienie i poprawienie sprawozdania. Konsultowanie się przed przekazaniem sprawozdania nie jest jest wymagane, ale w razie jakichkolwiek wątpliwości zapraszam.

Sprawozdanie **wstępne** powinno zawierać:

1. temat zadania, treść zadania, skład zespołu, data przekazania
2. interpretację treści zadania (doprecyzowanie)
3. krótki opis funkcjonalny – "black-box", najlepiej w punktach
4. opis i analizę poprawności stosowanych **protokołów komunikacyjnych** (wskazane - z rysunkami, np. zależnosci czasowych przy wymianie komunikatów, oraz postać/formaty komunikatów – np. W postaci tabelek lub rozpisanych w C struktur/obiektów)
5. planowany podział na moduły i strukturę komunikacji między nimi (być może z rysunkiem)
6. zarys koncepcji implementacji (język, biblioteki, narzędzia, etc.)

Nie należy opisywać kwestii znanych i omawianych na wykładzie, np. zasady funkcjonowania API gniazd, funkcji systemowych, standardowych narzędzi programistycznych, itp.

Co powinien zawierać projekt ostateczny (6-15 stron)

1. to co projekt wstępny
2. opis najważniejszych rozwiązań funkcjonalnych wraz z uzasadnieniem (opis protokołów, struktur danych, kluczowych funkcji, itp.)
3. szczegółowy opis interfejsu użytkownika
4. postać wszystkich plików konfiguracyjnych, logów, itp.
5. Opis wykorzystanych narzędzi, itp.
6. Opis testów i wyników testowania

Uwaga:

- **Kodowanie**: język C/C++, środowisku Linux (lub inny Unix: MacOS, BSD, Solaris, ...)
- **Pokaz** powienien odbywać się w środowisku obejmującym co najmniej 2 połączone siecią komputery, np. przez VPN, dozwolone jest środowisko zwirtualizowane, ale preferowane fizycznie oddzielne komputery.
- **eKonsultacje** w trakcie realizacji projektu nie wymagają obecności całego zespołu.
- **Sprawozdanie końcowe i pokaz** funkcjonowania **musi** odbywać się w obecności całego zespołu, pozostałe warunki formalne – patrz spr. wstępne.
- B. ważne jest precyzyjne opisanie obsługi **sytuacji wyjątkowych i reakcji na błędy**
- B. ważne jest szczegółowe opisanie przeprowadzonych testów – UWAGA: testy nie mają na celu wykazania, że program **działa** poprawnie. Test ma na celu wykazanie, że program **nie działa** poprawnie!
- kod źródłowy proszę przesyłać wyłącznie w formacie tar.gz na adres: GBlinowski@ii.pw.edu.pl dopiero po zaaprobowaniu programu i sprawozdania, w temacie wiadomości nalezy wpisać: "TIN.A 2020Z nazwisko"
- punktacja: proj. wstępny: 15p; ogólna ocena realizacji projektu: 15 p.; sprowozdanie końcowe: jakość i kompletność: 10 p, jakość kodu z punktu widzenia inżynierii oprogramowania (właściwe wykorzystanie dostępnych funkcji i mechanizmów systemowych, komentarze, czytelny podzial funkcjonalny i na moduły, nazewnictwo funkcji i zmiennych, poprawne skonstruowanie plików nagłówkowych (.h), logowanie, itd.): 10 p.; w sumie: 50 p.