Programowanie Niskopoziomowe
Jądro systemowe Linux
Strace

Wykonał:
Rafał Hudaszek
WfiIS

Wywołania systemowe

Wywołanie systemowe – stanowią interfejs między wykonywanym programem a jądrem systemu operacyjnego. Funkcje systemowe wywoływane są przez specjalny mechanizm, wspierany przez dany procesor, na przykład z użyciem wyznaczonego przerwania lub instrukcji skoku dalekiego

Śledzenie wywołań systemowych

strace — narzędzie do analizy kodu badające interakcję programu z jądrem systemu operacyjnego. Śledzi wywołania systemowe oraz sygnały w procesie. Może też zliczać i mierzyć czas poszczególnych wywołań.

Polecenie strace można zastosować do śledzenia procesu na dwa sposoby: uruchamiając program pod kontrolą strace bądź podłączając się do już działającego procesu o zadanym identyfikatorze.

Przykład użycia:

strace ls - Wynikiem instrukcji będzie wypisanie na standardowe wyjście, wszystkich wywołań systemowych wykorzystanych przy wykonaniu instrukcji „ls”

Przydatne flagi:

- -r wyświetla czas wykorzystany na pojędyńcze wywołanie systemowe liczone w sekundach
- -t wyświetla godzine wywołania każdego wywołania systemowego
- -c tworzy statystyki które zawierają między innymi:
  1. ilość wywołań danego wywołania systemowego
  2. sumaryczną ilość czasu spędzonego na dany typ wywołania
  3. ilość errorów dla danego wywołania
- -e trace=[arg] ogranicza wyswietlanie wywołań systemowych do konkretnego/konkretnych podanych w argumencie
- -o [arg] zapis do pliku (zapis do pliku można uzyskać także poprzez: 1>, 1>>, 2>, 2>> itp.)
- -p [PID] wywołania systemowego dla aktualnie działającego procesu
- -f śledzenie procesu w trakcie jego działania

Wywołania systemowe które mogą nas interesować:

- execve – wskazuje wywołanie konkretnej instrukcji/funkcji oraz podaje argumenty
- read – czyta dane z pliku/standardowego wejścia
- write – wypisuje dane na standardowe wyjście
- close – zamyka czytane pliki
- mmap – rezerwuje pamięć
- mprotect – chroni pamięć
- munmap – usuwa pamięć

Po bardziej szczegółowe informacje odsyłam do manuala strace -a.

Opis zadań

W części zadaniowej skupimy się na funkcjonalności programu strace.

Zad 1.

Zadanie polega na śledzeniu terminala innym terminalem w którym zostanie uruchomiony plik wykonywalny. Naszym zadaniem będzie ustalenie jakie instrukcje linuxowe zostają wywołane w pliku wykonywalnym, wraz z ich argumentami. Plik wykonywalny uruchamia 3 instrukcje.

Dostępne materiały:
- program strace
- skompilowany program

Cel zadania
- poznanie podstaw programu strace
- poznanie sposobu na badanie działającego programu, nie zawierając jego kodu

Przebieg zadania:
- uruchomić terminal X i terminal Y
- włączyć śledzenie terminala Y na terminalu X (użyteczna może być instrukcja *ps -ef | grep bash)*
- uruchomić skompilowany plik w terminalu Y
- przeczesać output w terminalu X w poszukiwaniu wywołania systemowego odpowiedzialnego za wywołanie instrukcji, zczytać instrukcje oraz jej argumenty

Zad 2.

Zadanie polega na stworzeniu statystyk które dotyczą pamięci. Programy stworzone są do wielokrotnego alokowania znacznej ilości pamięci.

Dostępne materiały:

- dwa programy napisane w języku C++, zadanie nie wymaga ich modyfikacji
- program strace

Cel zadania:

- nauka badania i formatowania wywołań systemowych
- poznawanie sposobu na wykrycie mankamentów programu
- uświadomienie sobie problemu marnowania czasu na zbędnę alokowanie pamięci

Przebieg zadania

- skompilować programy
- uruchomić plik wykonywalny stworzony przez kompilacje programu main.cpp
- opracować komende która stworzy statystyki wywołań systemowych które dotyczą wyłącznie pamięci. Statystyki mają być wykonane kilkukrotnie dla tej samej wielkości tablicy.
- Statystyki powinny dotyczyć tablicy wielkości: 1000, 100000, 10000000
- Po utworzeniu statystyk należy je luźno zinterpretować oraz wyciągnąć wnioski
- Przyklad poprawnego wyjścia w katalogu głównym.

Dziękuje za uwage
x mam nadzieje że rozwiązanie zadań nie zajeło wam duże czasu D