Systemy operacyjne 2		
Informatyka S1	Semestr 4	2015/2016
Laboratorium 8-9		

Wstęp teoretyczny:

- Obsługa wątków PTHREAD (f-cje pthread_...).
- Generowanie skrótów za pomocą kryptograficznych funkcji skrótu (f-cja crypt).
- Precyzyjne pomiary czasu (f-cje clock_gettime, pthread_getcpuclockid).

Zadanie:

- Zapoznać się z problemem generowania skrótów haseł systemowych za pomocą funkcji **`crypt (3)`**. Wykorzystujemy funkcję dostępną w wersji od **glibc 2.7** (np. na blade13), która wspiera kryptograficzną funkcję skrótu **SHA-512**. Napisać testowy program generujący skróty haseł do formy przechowywanej w **/etc/shadow** (kryptograficzna funkcja skrótu SHA-512 z domieszką). Program pobiera jako argumenty hasło i domieszkę (salt), a następnie zwraca skrót sformatowany do postaci przechowywanej w /etc/shadow. Przykładowo dla hasła **`dees`** i domieszki **`5MfvmFOaDU`** powinniśmy otrzymać skrót:

  `CVt7jU9wJRYz3K98EklAJqp8RMG5NvReUSVK7ctVvc2VOnYVrvyTfXaIgHn2xQS78foEJZBq2oCIqwfdNp.2V1`

  który trafiłby do /etc/shadow w formie:

  `$6$5MfvmFOaDU$CVt7jU9wJRYz3K98EklAJqp8RMG5NvReUSVK7ctVvc2VOnYVrvyTfXaIgHn2xQS78foEJZBq2oCIqwfdNp.2V1`

- Drugi program ma służyć do łamania hasła metodą słownikową. Jako argumenty wywołania podajemy:
  - sformatowany skrót hasła, które chcemy złamać (wypreparowany z /etc/shadow bądź otrzymany w wyniku działania pierwszego programu),
  - plik tekstowy zawierający słownik z hasłami (do testów udostępniono plik medium.txt z rzeczywistymi hasłami pochodzącymi głównie z wycieków),
  - liczbę wątków, które będą współpracowały przy obliczeniach.
- Program kończy swoje działanie albo po dopasowaniu hasła (wyświetlając je na ekranie) albo, w przypadku braku dopasowania, po przejściu całego słownika (odpowiednia informacja również powinna pojawić się na ekranie).
- Program powinien sprawdzić aktualnie dostępną w systemie ilość procesorów (funkcja **`sysconf (3)`**) i nie tworzyć większej od niej ilości wątków.
- Program powinien w procentach pokazywać zaawansowanie obliczeń (ilość sprawdzonych haseł ze słownika), uwzględniając wyniki wszystkich współpracujących wątków.
- Zwrócić uwagę na problem sekcji krytycznej w przypadku wykorzystywania przez wątki wspólnych zmiennych i zastanowić się nad optymalną częstotliwością ich aktualizacji.

- Nie wykonywać odczytu haseł z pliku pojedynczymi liniami (zbyt czasochłonne). Rozważyć odczyty wielokrotnościami bloków dyskowych.
- W przypadku braku argumentu określającego ilość wątków program nie ma docelowo łamać hasła a jedynie ustalić optymalną ilość wątków, które mogą zostać użyte. W tym celu program będzie wykonywał obliczenia dla pewnej ilości haseł (np. 1000) i dla różnej ilości wątków, mierząc przy tym dokładny czas obliczeń. Otrzymane wyniki powinny zostać wyświetlone na ekranie.

Uwaga! Kody źródłowe programów (2 pliki) po oddaniu prowadzącemu zajęcia laboratoryjne muszą zostać przesłany na adres so2@zut.edu.pl :

- plik z kodem źródłowym programu testowego musi mieć nazwę:

  ***numerindeksu*.so2.lab08.test.c**

  (np. 666.so2.lab08.test.c),
- plik z kodem źródłowym programu łamiącego hasła musi mieć nazwę:

  ***numerindeksu*.so2.lab08.hack.c**

  (np. 666.so2.lab08.hack.c),
- pliki muszą zostać wysłany z poczty wydziałowej (**wi.zut.edu.pl**),
- nagłówek maila musi mieć postać:

  SO2 IS1 XXXY LAB08

  gdzie XXXY to numer grupy (np. SO2 IS1 210C LAB08),
- w pierwszych trzech liniach plików z kodem źródłowym w komentarzach musi znaleźć się:
  - informacja identyczna z zamieszczoną w nagłówku maila,
  - imię i nazwisko wysyłającego oraz
  - adres email, z którego wysłał wiadomość,
- email **nie powinien** zawierać żadnej treści (tylko załącznik).

Dostarczone kody źródłowe będą analizowane pod kątem wykrywania plagiatów. Nie wysłanie wiadomości lub wysłanie jej w formie niezgodnej z powyższymi wymaganiami będzie traktowane jako brak programu i skutkowało otrzymaniem za niego oceny niedostatecznej.