Architektury systemów komputerowych

Pracownia 1: „Rozbrajanie bomby”

Zadanie

Cyfrowa bomba to program wykonywalny składający się z sześciu faz. W każdej fazie należy podać na standardowe wejście «stdin» pewien ciąg znaków. Jeśli jest on poprawny, to następuje przejście do kolejnej fazy. W przeciwnym razie bomba wybucha wypisując na standardowym wyjściu napis "BOOM!!!" kończąc swe działanie. Po przejściu wszystkich faz bomba jest rozbrojona.

Cyberterrorysta przygotował dla każdego studenta po jednej unikatowej cyfrowej bombie. Należy ją rozbroić, a przynajmniej spróbować tego dokonać. Na szczęście wybuch bomby nie ma żadnego wymiernego efektu, oprócz – być może – zranienia *ego* rozbrajającego. Dlatego rozbrajanie można powtarzać do skutku, bądź do upływu terminu zgłaszania rozwiązania. Punktowane są również rozwiązania częściowe – po jednym punkcie za każdą rozwiązaną fazę.

Uwaga! Rozbrajanie bomb innych studentów może mieć nieprzyjemne konsekwencje. Bomba może wtedy zdetonować zadając masowe obrażenia (–10 punktów) osobie rozbrajającej i osobie, która o to poprosiła.

Krok 1: Odbierz swoją bombę

Na stronie przedmioty w systemie SKOS znajduje się plik «bombs.tar.xz». Ściągnij go i rozpakuj. Bomba przeznaczona dla Ciebie znajduje się w katalogu «bomb-*identyfikator*», gdzie *identyfikator* jest nazwą konta studenta w systemie GitHub.

W katalogu tym znajdują się trzy pliki:

- «README» Zawiera dane właściciela bomby.
- «bomb» Bomba właściwa (plik wykonywalny).
- «bomb.c» Plik zawierający fragment kodu bomby w języku C.

Krok 2: Rozbrój bombę

Twoim zadaniem jest rozbrojenie bomby, czyli podanie napisów kończących każdą z sześciu faz programu. Każdy z poprawnych napisów wart jest jeden punkt.

Bombę można uruchomić z parametrem:

```
$ ./bomb psol.txt
```

Wtedy program przeczyta wejście z pliku «psol.txt» aż do jego końca, po czym rozpocznie czytanie reszty z «stdin». Dzięki temu napisy dla rozwiązanych już faz możesz umieścić w «psol.txt» i skoncentrować się na rozbrajaniu kolejnej fazy.

Krok 3: Zgłoś rozwiązanie

Plik zawierający całościowe bądź częściowe rozwiązanie zadania należy zamieścić w systemie GitHub Classroom. Zadanie jest warte 6 punktów. Przejście ukrytego etapu jest wliczone w maksymalną liczbę punktów możliwych do uzyskania.

Wskazówki

Zadanie można rozwiązać na kilka sposobów. Metoda siłowa polegająca na generowaniu wszystkich możliwych napisów wejściowych jest niepraktyczna. Będzie trzeba użyć sprytu i poniższych narzędzi:

- «gdb»

  GNU Debugger umożliwia pracę krokową, wyświetlanie zawartości rejestrów, stosu i pamięci, podglądanie kodu maszynowego, ustawianie punktów wstrzymań – odpowiednio dla zadanego licznika instrukcji (ang. *breakpoint*) i dostępów do zadanej komórki pamięci (ang. *watchpoint*), itp.

  Powstrzymasz bombę przed ciągłymi wybuchami odpowiednio ustawiając punkty wstrzymań. Każde z poleceń gdb ma interaktywny podręcznik – wpisz `help` w wierszu poleceń gdb.

  Dobrym pomysłem jest zapoznanie się z nakładkami debuggera ułatwiającymi poruszanie się w gąszczu informacji – prowadzący przedmiot poleca `GDB Dashboard`[1]. Zapewne przyda Ci się ściąga z poleceniami gdb[2] oraz samouczek[3] autorstwa Richarda Stallmana[4].

- «`objdump -t`»

  Wypisze tablicę symboli pliku wykonywalnego z bombą. Znajdą się tam nazwy wszystkich procedur i zmiennych globalnych, nazwy oraz adresy procedur wywoływanych przez bombę.

  Wskazówka: Czyżby znajomość nazw procedur miała okazać się użyteczna?

- «`objdump -d`»

  Wypisze zdezasemblowany kod bomby z podziałem na procedury. Niewątpliwie będzie to bardzo pomocne – powie Ci jak działa bomba. Nie daj się jednak zwieść! Wywołania procedur bibliotecznych, np. «`sscanf`» mogą w zdezasemblowanym kodzie wyglądać tak:

  ```
  8048c36:  e8 99 fc ff ff  call 80488d4 <_init+0x1a0>
  ```

  Stwierdzenie, że powyższa instrukcja woła procedurę «`sscanf`», jest możliwe tylko dzięki przeprowadzeniu wnioskowania na podstawie analizy uruchomionego programu pod kontrolą gdb.

- «`strings`»

  Wypisze wszystkie drukowalne napisy zawarte w pliku wykonywalnym z bombą.

Pamiętaj, że każde z powyższych poleceń ma odpowiednią stronę podręcznika systemowego, którą można wyświetlić poleceniem «`man`». Bardzo możliwe, że przyda się również przejrzeć stronę podręcznika «`ascii`». Podręcznik GNU Assembler przywołasz poleceniem «`info gas`», oczywiście jeśli masz zainstalowany odpowiedni pakiet z dokumentacją. Prowadzący poleca używać zamiennik «`pinfo`» dla programu «`info`».

Wskazówka: Rzucenie się na wyszukiwarkę internetową nie da tak dobrych efektów jak nauczenie się czytania dokumentacji!

[1] `https://github.com/cyrus-and/gdb-dashboard`
[2] `http://csapp.cs.cmu.edu/2e/docs/gdbnotes-x86-64.pdf`
[3] `http://www.unknownroad.com/rtfm/gdbtut/`
[4] `https://en.wikipedia.org/wiki/Richard_Stallman`