IT-Sicherheit
WS 2019/20
Prof. Dr. Elmar Böhler

Informatik Bachelor
Hochschule Kempten

Übungsblatt 5

1. Im Moodle finden Sie eine kleine C-Datei `a1.c`. Diese enthält drei Funktionen, nämlich `main`, `f` und `g`. Für diese Aufgabe modifizieren Sie bitte ausschließlich die Funktion `g` und zwar so, dass dadurch die Schleife in Funktion `f` zwar endlich oft, aber mehr als 2 Milliarden mal durchlaufen wird.

2. Im Moodle finden Sie eine kleine C-Datei `a2.c`. Diese enthält vier Funktionen, nämlich `main`, `harmless`, `q` und `evil`. Für diese Aufgabe modifizieren Sie bitte ausschließlich die Funktion `q` und zwar so, dass nach Abarbeitung der Funktion `q` die Funktino `evil` ausgeführt wird. Bewerkstelligen Sie das ohne einen expliziten Aufruf von `evil` nur durch direkte Manipulation des Stacks! Es ist kein Problem, wenn nach Abarbeitung von `evil` ein Segmentation Fault auftritt.

3. Im Moodle finden Sie eine kleine C-Datei `a3.c`. Zur Lösung dieser Aufgabe verwenden Sie bitte eine unveränderte, compilierte Version dieser Datei. (Sie dürfen schon an der Datei rumspielen und den Code erforschen und verändern, aber am Ende müssen Sie mit einer unveränderten Version arbeiten.) Wenn Sie das Programm starten, geschieht das folgende:

   - Es wird eine neue Instanz vom Typ `funcRef` erzeugt.
   - Der Benutzer wird dazu aufgefordert einen neuen Namen für diese `funcRef` einzugeben.
   - Anschließend wird die in der `funcRef` gespeicherte Funktion aufgerufen.

   Aufgabe: Sorgen Sie dafür, dass alleine durch Ihre Eingabe am Ende nicht die gute, sondern die böse Funktion ausgeführt wird.

   Hinweis: Möglicherweise ist die Aufgabe nicht oder nur sehr schwer lösbar, wenn Sie mit einem Betriebssystem arbeiten, bei dem ASLR aktiviert ist. Am besten arbeiten Sie also auf einem System, bei dem ASLR nicht aktiviert ist. Ich empfehle dringend, nicht Ihr eigenes System zu verändern, oder irgend eines, mit dem ernsthaft gearbeitet wird! Laden Sie sich stattdessen die freie Software "VirtualBox" von Oracle herunter und installieren Sie darauf Betriebssystem Ihrer Wahl, bei dem Sie wissen, wie man ASLR deaktiviert. Unter Ubuntu Linux lässt sich das mit dem folgenden Befehl bewerkstelligen:

   ```
   sudo bash -c 'echo 0 > /proc/sys/kernel/randomize_va_space'
   ```

   (Falls der obige Befehl per copy-und-paste nicht auf Anhieb funktioniert: Spielen Sie ein bisschen mit den Anführungszeichen herum, vielleicht wurden die ja falsch kopiert.)

4. Systemadministrator Gernot Notgern stellt fest, dass die Mitarbeiter in seiner Firma für ihre Passwörter nur sehr wenige Symbole verwenden und zwar die folgenden mit den jeweils angegebenen Wahrscheinlichkeiten:

Symbol	**Wahrsch.**	**Symbol**	**Wahrsch.**
`t`	$1/32$	`u`	$1/16$
`r`	$1/16$	`z`	$3/32$
`n`	$1/8$	`k`	$1/8$
`w`	$1/8$	`b`	$3/8$

   Gernot will die Passwörter der Mitarbeiter gegen Brute-Force-Angriffe schützen und den Suchraum für so einen Angriff hinreichend groß gestalten.

   (a) Davon ausgehend, dass prinzipiell nur die obigen 8 Symbole in Passwörtern benutzt werden, wie viele Stellen sollten die Passwörter mindestens haben, damit Angreifer, die zwar über die verwendeten Symbole bescheid wissen, aber nicht von ihrer Wahrsch.-Verteilung, per Brute-Force einen Suchraum von mindestens der Größe 2^{60} durchforsten müssen? Begründen Sie Ihre Antwort!

(b) Wie lang sollten die Passwörter mindestens sein, wenn die Angreifer (mit dem selben Wissen wie in der vorherigen Aufgabe) per Brute-Force genau 10000 Passwörter in der Sekunde durchprobieren könnten und ein Angreifer erwartungsgemäß mindestens ein Jahr zum Knacken eines Passworts brauchen soll? Begründen Sie Ihre Antwort!

(c) Gernot will auch auf den Fall vorbereitet sein, bei dem die Angreifer über die Symbolwahrscheinlichkeiten Bescheid wissen. Wie lang sollten die Passwörter sein, wenn eine zeichenbasierte Passwortentropie von mindestens 60 Bits erreicht werden soll? Begründen Sie Ihre Antwort!