Tema 3

- Deadline Soft: 20.01.2016 23:55
- Deadline Hard: 20.01.2016 23:55
- Data publicării: 09.01.2016 01:15
- Data ultimei modificări: 19.01.2016 10:55
- Responsabili:
  - Vladimir Diaconescu
  - Adrian Bogatu

Enunț

Să se compileze un program scris în limbaj C și să se rezolve următoarele:

1. Identificați posibilele vulnerabilități ce s-ar putea găsi în program (analiză statică).
2. Determinați userul și parola pentru login.
3. Creați o suită de intrări pentru program astfel încât:
   1. Programul să ajungă să execute funcția `secret_func1`.
   2. Programul să "apeleze" funcția `secret_func2` cu argumente valide de intrare.
   3. Programul să "apeleze" din biblioteca standard C funcția `fopen('<prenume>', 'w')` pentru a crea un fișier local având prenumele vostru.

Setup

Pentru dezvoltarea temei va trebui să folosiți mașina virtuală de Windows descrisă în secțiunea Mașini virtuale din pagina de resurse. Mașina virtuală de Windows este numită `SSS - Windows 7 32bit`. Această mașină virtuală este folosită și pentru verificarea temei pe vmchecker.

Puteți testa anumite funcționalități pe alt sistem, dar implementarea voastră trebuie să meargă pe mașina virtuală. Este posibil ca, datorită mediului diferit, anumite payload-uri să meargă pe un alt sistem dar nu pe mașina virtuală. Folosiți, ca referință, mașina virtuală.

În arhiva de resurse a temei veți vedea că există fișierul `seed.c`, precum și scriptul Python `mangler.py`.

Pentru a genera fișierul sursă C plecând de la `seed.c`, rulați comanda:

```
python mangler.py <nume> <prenume> <a treia cifră din numărul grupei>
```

unde:

- <nume> este numele vostru de familie (începeți cu majusculă)
- <prenume> este prenumele vostru (unul dintre ele dacă aveți două, începeți cu majusculă)
- <a treia cifră din numărul grupei> este ceea ce ce spune 😀; pentru grupa 324CA este cifra 4, iar pentru grupa 322CA este cifra 2

De exemplu, pentru `Ionescu Anca-Monica` de la grupa 323CA, modul de rulare a comenzii de generare a fișierului sursă C va fi

```
python mangler.py Ionescu Anca 3
```

Comanda de mai sus va genera fișierul `tema3.c`. Acesta este fișierul propriu-zis al temei care trebuie compilat.

Înainte de toate, recomandăm să adăugați în variabila PATH calea pentru gcc/gdb și alte utilitare, precum și pentru Python. Pentru aceasta, în `Command Prompt`, rulați comanda

```
set PATH=%PATH%;C:\Python27;C:\MinGW\bin
```

Acum puteți să compilați fișierul temei generat mai sus:

```
gcc -o tema3.exe tema3.c
```

Simplificat

Toți pașii de mai sus sunt realizați de scriptul de verificare (`checker.py`) din arhiva de resurse a laboratorului. Prin rularea scriptului de verificare se configurează variabila de mediu PATH se generează fișierul `tema3.c` și apoi se obține fișierul executabil aferent și se validează conținutul payload-urilor.

Dezactivare ASLR

Punctul 3 de exploatat (apelarea funcției `fopen`) necesită dezactivarea ASLR (*Address Space Layout Randomization*); altfel biblioteca standard C (`msvcrt.dll`) va fi încărcată la adrese diferite iar funcția `fopen` va avea la fiecare sesiune în sistem o altă adresă.

Pentru dezactivarea ASLR (așa cum este indicat aici) va trebui să editați o intrare în Windows Registry în mașina virtuală de Windows (vom face actualizarea mașinii). Pentru acesta urmați pașii:

- Folosiți combinația de taste `Windows + R` și rulați comanda `regedit`.
- În fereastra proaspăt deschisă (`Registry Editor`), navigați în `Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management`.
- Folosiți click dreapta pe intrarea `Memory Management` și selectați din meniul contextual `New → DWORD (32-bit) Value`.
- Numiți noua intrare `MoveImages` și lasăți-o inițializată la (`0x00000000`). Este inițializată implicit.
- Reporniți sistemul.

Acum ASLR este dezactivat și biblioteca standard C (`msvcrt.dll`) va fi încărcată la aceeași adresă (`0x6ff51000`). Adresa funcției `fopen` va fi mereu aceeași (`0x6ff6b2c4`). Puteți verifica acest lucru prin inspectarea executabilului `tema3.exe` la fel ca mai jos:

```
C:\Users\student\Downloads\iocla-tema3-resurse>gdb -q ./tema3.exe
```

```
Reading symbols from C:\Users\student\Downloads\iocla-tema3-
resurse\tema3.exe...
(no debugging symbols found)...done.
(gdb) start
Temporary breakpoint 1 at 0x4015fa
Starting program: C:\Users\student\Downloads\iocla-tema3-resurse/./tema3.exe
[New Thread 2388.0x9e4]

Temporary breakpoint 1, 0x004015fa in main ()
(gdb) p fopen
$1 = {<text variable, no debug info>} 0x6ff6b2c4 <msvcrt!fopen>
(gdb) info sharedlibrary
From        To          Syms Read   Shared Object Library
0x77ec1000  0x77ffbc3c  Yes (*)     C:\Windows\system32\ntdll.dll
0x77de1000  0x77eb30f8  Yes (*)     C:\Windows\system32\kernel32.dll
0x0dce1000  0x0dd2a668  Yes (*)     C:\Windows\system32\KernelBase.dll
0x6ff51000  0x6fffb2c4  Yes (*)     C:\Windows\system32\msvcrt.dll
(*): Shared library is missing debugging information.
(gdb)
```

Implementare

La nivelul implementării, aveți de scris câteva fișiere de intrare pentru programul compilat care să rezolve câte una din cerințele din enunț (de la punctul 3).

Recomandăm să generați aceste fișiere de intrare folosind un script Python sau orice alt limbaj de programare.

Trimitere și notare

Temele vor trebui încărcate pe platforma vmchecker (în secțiunea IOCLA) și vor fi testate automat. Arhiva încărcată va fi o arhivă `.zip` care trebuie să conțină:

- fișierele cu intrările pentru program: `payload_a`, `payload_b`, `payload_c`
  - Dacă ați folosit un script atunci adăugați în arhiva submisă și scriptul cu care ați obținut acele payload-uri.
    - Dacă nu ați folosit un script atunci adăugați în README modul de obținere a payload-urilor.
- fișierul `credentials.txt` care conține numele, prenumele și a 3-a cifră din numărul grupei, fiecare pe câte o line, astfel:

```
<Nume>
<Prenume>
<A 3-a cifră din numărul grupei>
```

- fișier README ce conține descrierea implementării
  - analiza statică (pe scurt)
  - cum ați determinat userul și parola de login

    - cum v-ați gândit să faceți programul să ajungă în `secret_func1`, `secret_func2`, `fopen`
- scriptul (sau scripturile) de generare a fișierelor payload

Punctajul este repartizat în felul următor:

- 20% README, cu pașii urmați pentru a rezolva cerințele, explicați în detaliu
- 25% analiză statică
- 25% determinare user și parola, și trecerea de procesul de login
- 30% cele trei exploatări

Ca să compensăm pentru întârzierea anunțării temei, tema valorează **1.5 puncte** față de **1 punct** cât valorează celelalte două teme. Punctajul total obținut pe teme se trunchiază la **3 puncte**.

Precizări suplimentare

În cadrul corectării, fișierul `tema3.c` este generat în mod automat pornind de la `seed.c` și datele voastre din fișierul `credentials.txt`. Acest lucru este realizat de scriptul de verificare `checker.py` care verifică și corectitudinea conținutului fișierelor de tip payload.

Datele din fișierul `credentials.txt` trebuie să fie corecte, cu excepția motivelor bine întemeiate.

Din moment ce programul de intrare scris în C este dat, analiza statică presupune să interpretați cod C, nu assembly.

Arhivă temă

Arhiva de resurse a temei, ce conține `mangler.py` si `seed.c` (în curând și checker-ul) o puteți descărca de aici.

Resurse ajutătoare

- Laborator 9: Gestiunea bufferelor. Buffer overflow
- Laborator 10: Exploatarea memoriei. Shellcodes
- Performing a ret2libc attack