Zadanie 6
I-UPB: Úvod do počítačovej bezpečnosti

November 15, 2021

1 Úvod

Cieľom zadanie je oboznámiť sa s existujúcimi zraniteľnosťami programov a ošetrenie týchto zraniteľností.

2 Format string injection

Štandardné knižnice jazyka C obsahujú funkciu *gets* ktorá načítava vstup zo štandardného vstupu. Uvažujme následovný príklad:

```
int main(){
    char in[255];
    char pwd[] = "safeheslo\0";

    while (1) {
        printf("Zadaj heslo\n");
        gets(in);
        printf("skusam heslo: ");
        printf(in);
        printf("\n");
        if (strcmp(&in, pwd) == 0) {
            printf("spravne heslo!");
        } else {
            printf("nespravne heslo");
        }
        return 0;
    }
}
```

Keď zadáme format string napr "%p %p %p" a vypíšeme heslo ktoré sme zadali do štandardného výstupu tak dokážeme odhaliť adresy v stacku ktoré môžeme ďalej využiť pre útok.

```
Zadaj heslo
%p %p %p
skusam heslo: 0x68206d6173756b73 (nil) (nil)
nespravne heslo
```

Pre ošetrenie tejto zraniteľnosti je nutné použiť funkciu fgets ktorej je nutné špecifikovať max. dĺžku načítavaného vstupu spolu s referenciou na vstup z ktorého sa číta. Zároveň je dôležité pri výpise zadávaného hesla použiť formátovací znak "%s" ktorý nám neumožní čítať adresy na stacku.

```
fgets(in, 20, stdin);
printf("skusam heslo: ");
printf("%s", in);
```

Po týchto zmenách dostaneme výstup:

```
Zadaj heslo
%p %p %p
skusam heslo: %p %p %p
nespravne heslo
```

3 Buffer overflow

```
int main() {
  char in[4];
  char isIn = 0;

  gets (in);

  if (strcmp (in, "abcd"))
    {
      printf ("nespravne heslo\n");
    }
  else
    {
      printf ("spravne heslo, ");
      isIn = 1;
    }

  if (isIn)
    {
      printf ("u in boi");
```

```
    }

  return 0;
}
```

Pri tomto útoku nám ide o prepísanie hodnoty v premennej isIn tým že zadáme reťazec vačší ako 4 znaky ktoré sú vyhradené v poli *in*. Táto zraniteľnosť sa mi nepodarila nasimulovať pretože kompilátor mi to nedovoľoval. Kompilátor zaznamenal že sa snažím použiť tento útok a pri jeho detegovaní program okamžite ukončil.

```
abcde
nespravne heslo
*** stack smashing detected ***: terminated
```

Pre ochranu proti tomuto útoku (pri potencionálne starších kompilátoroch ktoré spomínanú ochranu nemajú) e nutné znova použiť funkciu *fgets* v ktorej špecifikujeme dĺžku načítavaného reťazca. Tak isto je nutné špecifikovať dĺžku porovnávaného reťazca vo funkcií strcmp

```
  fgets (in, 4, stdin);
  if (strcmp (in, "abcd", 4))
```

Po týchto úpravách sa program po zadaní dlhšieho reťazca správa tak isto ako pri zadaní nesprávneho hesla teda, korektne sa ukončí s hláškou že bolo zadané zlé heslo.