Securitatea sistemelor de calcul

Vulnerabilități software și programare defensivă

Marius Minea

19 octombrie 2012

Rezumat: atacuri tip buffer overflow

– buffer overflow “clasic”
    detectabil protecția stivei (canar/copie)
    îngreunat de randomizarea adreselor
    eliminat prin protecția memoriei la execuție
    variantă: depășirea în heap (nu în stivă)

– atac “return into libc”
    execută cod legitim; detectabilă prin canar/copie RET
    permite înlănțuirea mai multor apeluri

– suprascriere de pointeri: pointeri la funcții, pointer din `longjmp`, pointeri la funcții de bibliotecă (PLT: procedure linkage table)

Condiții pentru atac și protecție împotriva lor

1) se poate depăși zona de memorie alocată
   secure programming
   analiză statică pentru detectarea depășirilor
   demonstrații că programul e corect

2) se poate insera conținut arbitrar
   de regulă da (intrarea e controlată de utilizator)
   eventual: filtrarea datelor de intrare

3) se știe adresa care trebuie suprascrisă
   Address Space Layout Randomization (încarcă programul / bibliotecile la adrese aleatoare în memorie)

Condiții pentru atac și protecție împotriva lor

4) se poate suprascrie adresa de return
    plasarea de compilator a variabilelor *după* RET
    plasarea variabilelor critice (pointeri) *înainte* de tablouri

5) se poate executa RET suprascris $\Rightarrow$ verificarea înainte de RET
    cu "canar" (terminator/XOR/random) înainte de adresa de RET
    prin copierea adresei de RET într-o zonă dedicată

6) se poate executa codul injectat
    protecția segmentelor de memorie (W xor X):
un segment scris (accesibil la scriere) nu poate fi executat
    tehnică implementată și în PAX (pt. Linux)