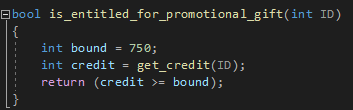
תכנות מערכות דפנסיבי – ממ"ן 12

מאת: יאיר חריט 207282955

1. א. החולשה הקיימת בקטע הקוד הנתון היא חולשת גלישה נומרית המתרחשת כאשר משווים בין credit וbound. מאחר והאחד מסוג int והשני מסוג unsigned int כאשר הביט הגבוה ביותר יהיה 1 יציג כל משתנה ערך שונה עבור אותם ביטים בזכרון, זאת משום שבמשתנים מסוג int מסמנים מספרים שליליים בעזרת הדלקת הביט הגבוה. על כן, כאשר ישוו בין int שלילי לunsigned int תתקבל תוצאה כי credit >= bound

על כן, ניתן לנצל חולשה זו ולתקוף את המערכת ע"י כך שנכנסים למינוס בבנק!  
לקוח אשר יהיה סכום שלילי בחשבונו יקבל גם הוא את המתנה (בנק מאוד נחמד)

ב. ניתן לתקן חולשה זו ע"י הגדרת המשתנה bound כint (credit יכול להיות שלילי ולכן אינו יכול להיות מטיפוס unsigned int).

כאשר משווים בין 2 int-ים אנו נמנעים מהשוואות שגויות כמתואר בסעיף א', שכן, שני המשתנים נשארים באותו "קנה מידה".

1. החולשה:

החולשה המתחבאת בקטע הקוד הנתון הינה חולשת גלישת חוצץ (Buffer Overflow):

בפונקציה handle\_escape, כאשר מעתיקים את הפרמטר str לl.buffer אין דבר המונע מאיתנו לדרוס את הזכרון של h (Handler), כלומר לשנות את הטבלה הוירטואליות של האובייקט.

על כן, כאשר נזין מחרוזת ארוכה מספיק נדרוס את ראש הטבלה הוירטואלית כך שתצביע על פונקציות אחרות מכפי שהגדיר המשתמש. כאשר ינסה המשתמש לעשות שימוש בפונקציות שהגדיר, יקראו פונקציות אלו **במקום** פונ' המשתמש ההתחלתיות.

ההתקפה:

תחילה, על מנת לאפשר שימוש ב'\' עלינו להגדיר משתנה סביבה בשם "ECHOUTIL\_OPT\_ON" ולתת לו ערך כלשהו (אפילו 0) על מנת שהמשתנה env לא יהיה NULL וכך allow\_options = True.

שנית, על מנת להגיע לקטע הקוד הפגיע עלינו להכניס את הארגומנטים הבאים לתכנית הראשית:

* ארגומנט 1: הדגל '-e'

כך שdo\_escape יהיה True.

* ארגומנט שני:

\x123456789012345 \*

* יתחיל ב'\' כך שהתכנית תכנס לתנאי בשורה 154
* התו השני יהיה 'x', על מנת שהתכנית תכנס לפונ' l.h.interpret
* כעת נכניס 15 תווים נוספים על מנת למלא את הbuffer
* לאחריהם נכניס את ערך הascii של הבית הנמוך של כתובת הטבלה החדשה (אין צורך לשנות את שאר הבתים במקרה שלנו):
  + באמצעות הdebugger נמצא את הכתובת הקיימת ונשמור את הבית הנמוך ביותר (lsb), במקרה זה LSB = 0x10
  + מאחר וגודל כל כתובת הינו 4 בתים נחסר מהLSB שמצאנו 4 - כך שכאשר ננסה לגשת למקום 1 (helper) בטבלה נגיע למקום 0 (unreachable). במקרה זה הערך החדש יהיה 0x0C.
  + מאחר והקלט נעשה כמחרוזת נמיר את התוצאה לערכי ascii ונוסיף אותה לסוף הארגומנט.

כעת כאשר תנסה המערכת לגשת לפונק' l.h.helper על מנת לעבד את הארגומנט השני תקרא הפונק' l.h.unreachable.

ההגנה:

ניתן למנוע התקפה מהסוג המתואר לעיל ע"י עטיפת הלולאה המעתיקה בפונקצייה handle\_escape בתנאי הבודק את אורך המחרוזת.

