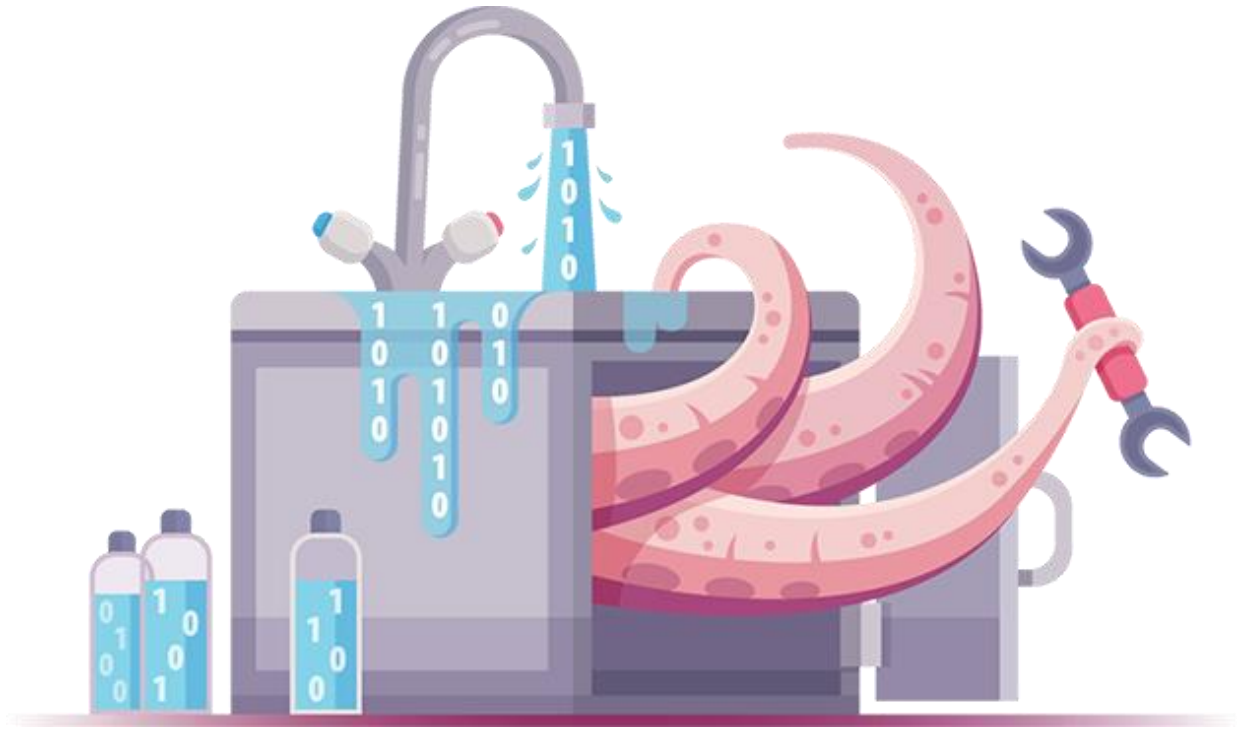
תרגיל בית 4 בהנדסה לאחור

Vulnerabilities

236496

סמסטר אביב תשפ"א

הגשה בזוגות עד 1.7.2021 בשעה 23:59



לשאלות על התרגיל: idan.raz@campus.technion.ac.il

נא לרשום בנושא ההודעה "RES21-EX4" (ללא המרכאות).

הקפידו על הגשת קבצים עם השמות הנכונים כפי שמוגדר במסמך זה.

טעות בשם הקובץ תגרור ציון 0 ע״י הבודק האוטומטי בחלק הרלוונטי.

לפני ההגשה, ודאו שההגשה שלכם תואמת את הנחיות ההגשה בסוף התרגיל.

הנכם מוזמנים לצרף memes להגשה. סוג זה של בידור אהוד במיוחד על בודק התרגילים.

חלק יבש

יש בידנו קוד עם חולשת buffer overflow אשר מאפשר ניצול של strcpy **בגרסתה עבור Unicode Strings**. בכדי לדרוס את המחסנית ולהשתמש ב-ROP. אנו רוצים להשתמש בשרשרת ROP על מנת לשנות את ההרשאות לעמוד מסוים (בכדי לאפשר: הרצה, כתיבה, וקריאה) ואחר כך לקפוץ לתחילת העמוד (שימו לב: יש כאן שתי פעולות לבצע). ידוע כי בכתובת 0x70707070 יש מצביע (פוינטר) לאובייקט אשר בהיסט 27 בתים (בבסיס עשרוני) מתחילתו יש מצביע אל תחילת העמוד הרצוי. מתוך ניתוח DLL הנמצא בתמונת הזיכרון של הקובץ, נמצאו בצורה אוטומטית ה-gadget-ים הבאים (מופרדים ע"י שורות של ----------):

```
9ad9e700:  01 d8                add     eax,ebx
9ad9e702:  5b                   pop     ebx
9ad9e703:  59                   pop     ecx
9ad9e704:  5b                   pop     ebx
9ad9e705:  c3                   ret
9ad9e706:                ;-------------------------------------
9ad9e706:  31 c0                xor     eax,eax
9ad9e708:  c3                   ret
9ad9e709:                ;-------------------------------------
9ad9e709:  89 c1                mov     ecx, eax
9ad9e70b:  83 c1 0a             add     ecx, 0Ah
9ad9e70e:  89 c8                mov     eax, ecx
9ad9e710:  c3                   ret
9ad9e711:                ;-------------------------------------
9ad9e711:  40                   inc     eax
9ad9e712:  c3                   ret
9ad9e713:                ;-------------------------------------
9ad9e713:  5b                   pop     ebx
9ad9e714:  5d                   pop     ebp
9ad9e715:  c3                   ret
9ad9e716:                ;-------------------------------------

9ad9e716:  8b 40 0f             mov     eax, DWORD PTR [eax+0Fh]
9ad9e719:  c3                   ret
9ad9e71a:                ;-------------------------------------
9ad9e71a:  5f                   pop     edi
9ad9e71b:  c3                   ret
9ad9e71c:                ;-------------------------------------
9ad9e71c:  60                   pushad
9ad9e71d:  c3                   ret
```

1. ה gadget האחרון (מכתובת 0x9ad9e71c) מהווה רצף פקודות אשר לא הגיוני למצוא בתוכנית לגיטימית. בהנחה שה DLL אכן לגיטימי, כיצד ייתכן כי חיפוש אוטומטי מצא כזה gadget? הדגימו.
2. בנו שרשרת ROP המבצעת בזו אחר זו את שתי הפעולות הנדרשות לעיל (שינוי הרשאות ולאחר מכן קפיצה לתחילת הדף). ניתן להניח:

a. כתובות פונקציות המערכת (פונקציות בDLL-ים של Windows) הנדרשות בעבור הפעולות ידועות וקבועות.
b. ניתן לכתוב כל ערך לאזור הכתובות 0x200000-0x300000 (כלומר, יש לשם הרשאות כתיבה, וכתיבה לשם לא תפגע בביצוע התקין של התוכנית).
c. על שרשרת ה-ROP להיות מוצגת בתצוגת מחסנית – כמו התצוגה שראיתם בסדנה / בתרגול.

3. כעת, לאחר כתיבת שרשרת ה-ROP בסעיף הקודם, ניסינו להחדירה לתוכנית דרך חולשת ה buffer overflow ולהריץ. באופן מפתיע, לא קרה מה שציפינו. מדוע שרשרת ה-ROP לא עובדת במקרה ספציפי זה?

4. כעת מסופק לכם גם ה gadgetים הבא:

```
6ad6e71d:                    ;-------------------------------------
6ad6e71d: f7 5c 24 04                neg     [esp + 04h]
6ad6e71f: c3                         ret

6ad6e720:                    ;-------------------------------------
6ad6e720: f7 54 24 0c                not     [esp + 0Ch]

6ad6e724: c3                         ret
```

תארו כיצד תשנו את שרשרת ה rop כך שהתוכנית כן תעבוד.

חלק רטוב – כיבוי שריפות

לאחר שברח השודד משביו, הבינה הקבוצה כי מישהו מנסה לעצור אותם. לכן, החליטה קבוצת המורדים המתוסכלת לצאת למבצע אחד אחרון. הם גרמו לאבירי המשחק לפנות כנגדו ולהצית את שדות המשאבים. נכון לעכשיו הובערו כל שדות החצץ והעץ ואין לדעת מה עוד הם מתכננים! עליכם מוטלת האחריות לעצור את השריפות!

בתרגיל זה נשיג את היכולת להריץ קוד על אחת המכונות שבשימוש הקבוצה ונחפש רמזים כיצד ניתן לעצור את ההצתות. משימתכם תושלם כאשר שדות המשאבים בלוח המשחק יחזרו למצב המקורי (לא יהבהבו באדום).

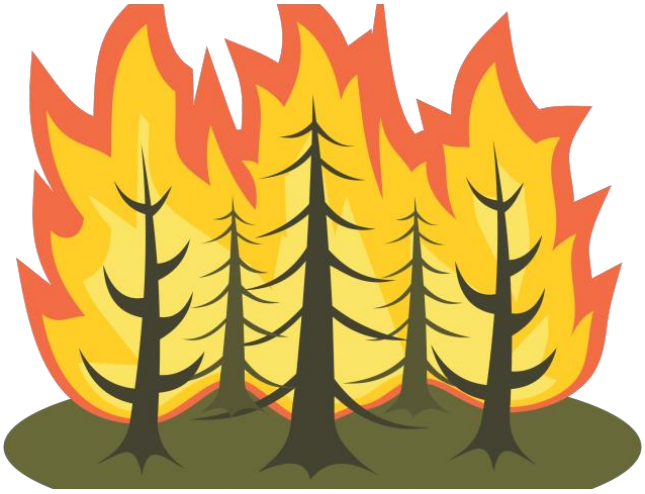
בעקבות הצלחתכם במשימות קודמות מישהו ערך את ה client שמצאתם באתר בדף ה tools (הגרסה החדשה מופיעה תחת השם hw4_client.exe אך לאורך התרגיל נתייחס אליו בשם client.exe). כעת הוא מוגן ע"י שם משתמש וסיסמא. נרצה להשתמש בקובץ client.exe כדי להגיע אל השרת ולהריץ עליו קוד כרצוננו.

שימו לב!

אין להתחבר לשרת שלא דרך client.exe בשום שלב של הפתרון.

קלפי פיתוח ואבירים

במשחק "קטאן" ניתן לרכוש קלפי פיתוח תמורת משאבים. ישנם מספר סוגי קלפי פיתוח ולכל אחד השפעה אחרת על המשחק. ישנם קלפים התורמים למחזיק בהם נקודת ניצחון, ישנם כאלה המאפשרים בנייה בחינם ועוד. רוב קלפי הפיתוח במשחק הם מסוג אביר. ניתן להשתמש באביר על מנת להבריח את השוד מהאריח עליו הוא עומד באותו הרגע, ולמקמו מחדש על הלוח. כאשר השודד ממוקם מחדש, הוא כמובן מבצע שוד, על פי אותם הכללים הרלוונטיים להזזת השודד.



שלב ראשון – התחברות

התחילו בחקירת התכונית client.exe שברשותכם. תוכלו לראות כי בשלב הראשון התוכנית מבקשת פרטים מזהים על מנת להתחבר לשרת. פרטי ההתחברות לשרת הם זהים לפרטי המשתמשים של האתר, עם הוספת u לכל שם משתמש. לדוגמא, שם המשתמש goblin יהיה כעת uGoblin.

התחברו עם הפרטים השונים, שחקו מעט עם הפקודות השונות המוצעות לכם לביצוע וענו על השאלות הבאות (ביבש): איזה מן המשתמשים בעל ההרשאות הגבוהות ביותר? מה האינדיקציה לכך?

שלב שני – כתיבת shellcode

כפי ששמתם וודאי לב, לאחר הדפסת התפריט השרת מחכה לקלט, בנוסף, לאחר קבלת הקלט והתכנית מגיבה ויוצאת מיד. מטרתכם בשלב זה היא לנצל זאת על מנת להריץ קוד כרצונכם בצד הלקוח (כלומר על המחשב שלכם). נרצה להשיג את היכולת להריץ פקודות שוב ושוב מבלי שהתוכנית תצא. על מנת לעשות זאת תכתבו shellcode אשר יאפשר לכם להריץ פקודות **PEEK** שוב ושוב על גבי השרת.

עליכם לכתוב קובץ **attack_shell.py** המבצע את ההתקפה (**בפייתון 3.6 בלבד**, שימו לב כי אתם עובדים עם גרסת bit32). ייתכן כי שורת הפקודה תהיה שונה מעט, כתלות במחשב עליו תריצו:

```
python3.6 attack_shell.py
```

- חשבו כיצד למצוא את ה socket דרכו מתקשרים עם השרת. הסבירו איך זה משפיע על מיקום ה shellcode שלכם ביחס למיקום של הshellcode שראינו בסדנה.
- ענו על השאלה הבאה: האם ה shellcode שלכם צריך להשתמש במחסנית כמבנה נתונים? אם לא הסבירו מדוע. אם כן, הסבירו כיצד תשתמשו במחסנית הן להרצת הקוד והן לשמירת מידע.
- שימו לב כי עליכם לקבל את הפקודות לביצוע מהקלט הסטנדרטי בתוכנית ה python ולהעביר אותן ל shellcode ב client.exe.
- בסדנה חלקכם נתקל בבעיה: עבור הזנה של ערכים מעל 0x7f נקלטו במפתיע זוג בתים כאשר השני הוא C2 או C3. הדבר נגרם כיוון ש python מקודדת את התווים הללו ב utf-8 (משום שהם חורגים מטווח הערכים של ascii). על מנת להתמודד עם הבעיה, כאשר אתם שולחים לתוכנית client.exe רצף של בתים (bytearray) צרו אותו מרשימה של ערכים מספריים **(ולא מערך מטיפוס string)**.

בכל המודולים למעט ה client.exe עצמו מופעל ASLR. על מנת לקפוץ חזרה למחסנית נשתמש בפקודה jmp esp אשר נמצאת בclient.exe. אתם יכולים למצוא את המיקום של פקודה זו באמצעות ropper כפי שראינו בסדנה.

שלב שלישי – הרצת קוד על השרת

בשלב זה נרצה להשיג יכולת להריץ קוד בצד השרת. שחקו מעט עם הפקודה PEEK שביכולתכם להריץ כעת. הסבירו בדיוק מה היא מבצעת, מה היא מאפשרת לכם לגלות על השרת? השתמשו במידע שמצאתם על מנת למצוא חולשה בפקודה זו. הסבירו כיצד חולשה זו מאפשרת לכם להריץ קוד על גבי השרת.

כעת עדכנו את הקובץ **attack_shell.py** כך שיקבל בקלט הסטנדרטי רק את הפקודות לביצוע על השרת. בנוסף דאגו שעבור הפקודה exit ה shell שלכם ייסגר. כמוצג בתמונה:

```
Sunday, Jan 12, 2020 23:59:43 → tools [238]: Error: 1 › python .\attack_shell.py
Enter username: Enter password: Welcome [redacted]

What would you like to do?
[1] ECHO - ping the server with a custom message, receive the same.
[2] TIME - Get local time from server point of view.
[3] 2020 - Get a a new year greeting.
[4] USER - Show details of registered users.
[5] DMSG - Download message from the server.
[6] PEEK - peek into the system.
[7] LOAD - Load the content of the last peeked file.
your choice (4 letters command code): send failed with error: 10038
echo echoing from the server!
echoing
from
the
server!

For ($i=1; $i -le 10; $i++) { \"10 * $i = $(10 * $i)\" }
10 * 1 = 10
10 * 2 = 20
10 * 3 = 30
10 * 4 = 40
10 * 5 = 50
10 * 6 = 60
10 * 7 = 70
10 * 8 = 80
10 * 9 = 90
10 * 10 = 100

date

Monday, January 13, 2020 12:00:08 AM



exit
Monday, Jan 13, 2020 00:00:11 → tools [239]:
```

הערות חשובות:

- וודאו שאתם עובדים עם גרסת פייתון 3.6.
- וודאו שהסקריפט שלכם עובד עם input redirection, כלומר קבלת פקודות מקובץ ולא מהcmd.

שלב חמישי – כיבוי השריפות

מעולה! השגתם יכולת הרצת קוד על השרת. כעת נרצה לנצל זאת! חפשו במערכת הקבצים ומצאו דרך לעצור את האבירים ולכבות את השריפות. הסבירו כיצד ביצעתם זאת.

הוראות הגשה

עליכם לתאר **במפורט** את הדרך שבה פעלתם על מנת להשלים את המשימה. על בודק התרגילים להיות מסוגל לשחזר את הדרך שעברתם מבלי להיתקע. בנוסף הגישו כל קובץ שרלוונטי לפתרון התרגיל שלכם. בקטגוריה זו נמנים בין היתר קבצי קלט עבור תכניות, ארגומנטים בשורת הפקודה, וקבצי קוד שכתבתם לכל מטרה שהיא. שימו לב! בתרגיל זה תממשו את הפתרונות בשפת Python.

קבצים להגשה

- קובץ **dry.pdf** המכיל את הפתרונות שלכם לחלק היבש ואת תיאור הפתרון של החלק הרטוב.
- **attack_shell.py** אותו התבקשתם ליצור במהלך התרגיל.
- כל קובץ נוסף ניתן להוסיף תחת תיקיית extras.
- ניתן להניח כי הקובץ hw4_client.exe נמצא בתיקייה הנוכחית בעת הרצת הטסטים.

שומעים חופשיים

שומעים חופשיים בקורס אשר אין להם גישה לתרגיל מוזמנים לפנות אל עידן במייל, ולקבל עותק.

בהצלחה!