מס' מחברת: _ _

| |
|---|
| **ת"ז:** |

# אוניברסיטת תל-אביב
## מדעי המחשב

# מבוא לאבטחת מידע
## מועד א, תשע"ה

**תאריך: 26.6.2015**
**(נוסח מתוקן)**

**מרצה: ערן טרומר**

## זמן המבחן: 3 שעות

<u>הנחיות:</u>

- מותר להשתמש בחומר עזר כתוב או מודפס על גבי נייר בלבד. עזרים אלקטרוניים אסורים.
- המבחן כולל 5 שאלות. מספר הנקודות עבור כל שאלה מופיע בסוגריים.
- כתבו את תשובותיכם על גבי טופס המבחן במקום המוקצה לכך. מומלץ לכתוב תחילה את התשובה במחברת הטיוטה שקיבלתם ורק אחר כך להעתיק אותה, בצורה ברורה וקריאה, לטופס המבחן. תשובות במחברת הבחינה לא יקראו.
- נמקו בקצרה אך בבהירות את כל טענותיכם. תשובה ללא נימוק לא תזכה בניקוד.
- יתקבלו תשובות המניחות הנחות סבירות ומציינות במפורש את ההנחות.
- ניתן לענות בעברית או באנגלית
- במבחן זה **<u>16</u>** עמודים (כולל עמוד זה). אנא ודאו שכולם ברשותכם.

ב ה צ ל ח ה!

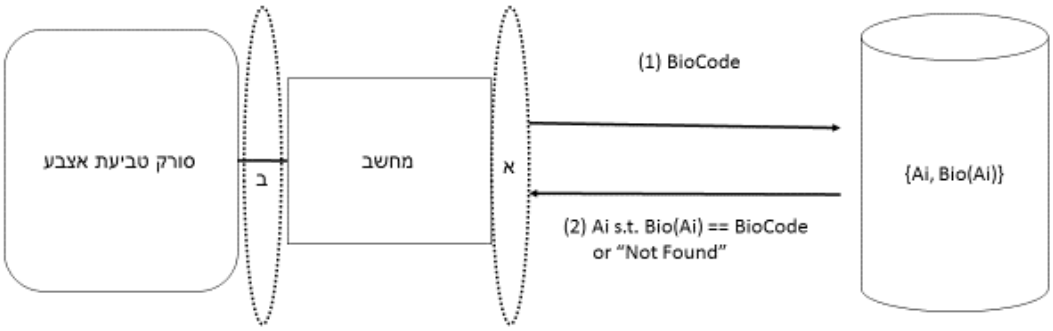
לשימוש הבודקים:

| 1. (א) | 2. (א) | 3. (א) | 4. (א) | 5 (א) |
|---|---|---|---|---|
| (ב) | (ב) | (ב) | (ב) | (ב) |
| (ג) | (ג) | (ג) (ד) | (ג) | (ג) |
| | | (ה) (ו) | | (ד) |
| | | (ז) (ח) | | |

מס' מחברת: _ _

## שאלה 1 (13 נקודות)

במאגר ביומטרי B מקודדים את טביעת האצבע של אזרח על ידי קידוד בשם Bio שהוא רצף של 40 ביט. המאגר בנוי מטבלא של זוגות {Ai, Bio(Ai)} כאשר Ai הוא מס' תעודת הזהות ו-Bio(Ai) הוא הקידוד של טביעת האצבע של האזרח. מספרי תעודת הזהות עצמם אינם סודיים ואין קושי לברר מהו Ai של אזרח מסוים. נניח כי המאגר הוקם ומשתמשים בו לצורך זיהוי אזרחים במעבר גבול. במאגר רשומים $N > 8{,}000{,}000$ אזרחים.
המערכת בגבול בנויה סכמטית כך:




סורק טביעת האצבע מחשב את קידוד Bio בעצמו. המחשב (במרכז השרטוט) מקבל את קידוד Bio מסורק טביעת האצבע, ושולח אותו למאגר. אם הקידוד מתאים לקידוד של אזרח Ai כלשהו אז מספר תעודת הזהות נשלח חזרה, אחרת המאגר עונה "Not Found". בשעת לילה אין פקיד במעבר הגבול והמערכת אוטומטית לחלוטין – אם המחשב מקבל זיהוי של אזרח (ולא "Not Found") המחשב פותח את מעבר הגבול.

א) [4 נק] הניחו כי תוקף מסוגל להתערב באזור "א" בשרטוט ולשנות את תוכן ההודעות בשני הכיוונים. תארו התקפה בעלת מספר מינימלי של הודעות שהתוקף יכול ליצר עד שהשער יפתח.

מס' מחברת: _ _

ב) [5 נק] הניחו כעת כי התוקף מסוגל להתערב רק באזור "ב" בשרטוט ולשנות את תוכן ההודעות מן הסורק את המחשב. תארו התקפה שהתוקף יכול לבצע, ותנו הערכה סבירה (מנומקת) למספר הניסיונות שהתוקף צריך לנסות עד שהשער יפתח.

ג) [4 נק] נניח כעת כי כל המאגר הביומטרי נגנב על ידי התוקף. בתנאים של סעיף ב (התוקף יכול להתערב באזור "ב" בשרטוט) – מה התקפה היעילה ביותר שיכול התוקף לבצע כעת?

מס' מחברת: _ _

## שאלה 2 (12 נק')

בתקשורת אינטרנט סלולרית בשיטת MobileX צד אחד יכול לשלוח לצד השני הודעות שנקראות binding-update . ההודעה נשלחת מכתובת ה-IP הנוכחית של התחנה השולחת, ונותנת לצד השני כתובת IP חדשה של התחנה השולחת, כדי שהצדדים יוכלו להמשיך ולתקשר ישירות באמצעות הכתובת החדשה ללא ניתוק (נחוץ למשל כאשר התחנה זזה לאזור רשת אחר ומקבלת כתובת IP חדשה).

א. [4 נק'] אם הודעות ה-binding-update אינן מצריכות שום וידוא (authentication), איך יכול תוקף, המצוייד במחשב המחובר לרשת אך ללא שום סמכויות או גישה מיוחדות, לצותת לתקשורת בין שתי תחנות הפועלות בשיטת MobileX? הניחו כי התוקף יודע את כתובות ה-IP של שני הצדדים והם אינם זזים כרגע.

ב. [4 נק'] אם הודעות ה-binding-update אינן מצריכות שום וידוא (authentication), איך יכול תוקף, המצוייד במחשב המחובר לרשת אך ללא שום סמכויות או גישה מיוחדות, ליצור התקפות Denial-of-Service על יעדים המחוברים לאינטרנט? כדי לעשות זאת עליו לגרום לשליחה של כמויות גדולות של תקשורת אל כתובתו של הנתקף. רמז: התוקף יכול לצפות בסרטים מהאינטרנט.

מס' מחברת: _ _

ג. [4 נק'] נניח כי בגרסא 2 של MobileX הוסיפו מנגנון הגנה כנגד ההתקפה מסעיף ב, כדי לוודא שהתחנה שזזה אכן מחזיקה את הכתובת המדווחת כחדשה. המנגנון כולל 2 הודעות נוספות: כאשר תחנה Z מקבלת הודעת binding-update המדווחת על תזוזה של ip1 ל-ip2, Z שולחת הודעת verify לכתובת החדשה ip2, ומצפה לקבל מ-ip2 הודעת verify-ack שמודיעה "הייתי קודם בכתובת ip1" . עדכון הכתובת מבוצע ע"י Z רק אם הודעת verify-ack תקינה מגיעה תוך 20 מילי-שניות.

האם המנגנון המוצע מונע את ההתקפה מסעיף ב? אם כן נמקו, ואם לא תארו התקפה שמתגברת על המנגנון.

מס' מחברת: _ _

## שאלה 3 (40 נקודות)

נתקיף שרת לינוקס מרוחק הרץ על מעבד אינטל 32 ביט אשר גרסאות התוכנה והספריות שלו ידועות בדיוק מוחלט. מופעל DEP ו-ASLR. השרת מריץ תוכנה המכונה "לב מלבלב" המכיל את קטע הקוד הבא:

```
void handle_echo_command(unsigned char *msg, unsigned char len) {
      char buf[224]; // 224 == 0xE0
      if (len >= sizeof(buf))
            return;
      snprintf(buf, sizeof(buf), "%s", msg);
      send_outgoing_message(buf, len);
}

void handle_log_command(unsigned char *msg, unsigned char len) {
      char buf[112]; // 112 == 0x70
      sprintf(buf, "msg: %s", msg);
      send_outgoing_message("OK", 2);
      puts(msg);
}


void handle_incoming_message(unsigned char *msg) {
      unsigned char type = msg[0];
      unsigned char len = msg[1];

      if (type == 'E') {
            handle_echo_command(msg, len);
      } else if (type == 'L') {
            handle_log_command(msg, len);
      }
}
```

לנוחותכם בסוף השאלה יש תיאור של פונקציות הספריה sprintf, snprintf, puts.

אנחנו הלקוח (המתקיף את השרת). עומדת לרשותנו ספריית פיתון בשם `remoteproto` אשר מכילה את הפונקציה הבאה:

```
answer = remoteproto.send_and_recv(msg)
```

הפונקציה רצה אצל הלקוח. מקבלת כפרמטר מחרוזת בינארית (רצף בתים שרירותי) `msg` ושולחת אותה אל השרת המרוחק.

השרת המרוחק מטפל בהודעה בעזרת פונקצית `handle_incoming_message` שלמעלה. המחרוזת `msg` מגיעה כלשונה כפרמטר בקריאה ל- `handle_incoming_msg..`

השרת המרוחק עונה ע"י קריאה לפונקציה `send_outgoing_message` אצלו. הפונקציה `send_and_recv` אצל הלקוח מקבלת את הודעה התשובה הזו, ואז מסיימת ומחזירה את הודעת התשובה כ-return value.

מס' מחברת: _ _

א. [5 נק] זהו חולשה בקוד אשר באמצעותה ניתן לגרום לקריסה של הקוד, וענו: מאיזה סוג החולשה? נמקו, וכיתבו קוד פייתון אשר גורם לקריסה.

ב. [5 נק] תחת ההנחה שיש לנו רק ניסיון אחד בלבד לגרום לקריסת התוכנה, ושאין בה חולשה נוספת, הסבירו מדוע לא תוכל לנצל את החולשה על מנת לבצע `Control Hijacking`.

להלן מסך IDA של הפונקציות handle_echo_command ו-handle_log_command:

```
08048614 ; ¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦ S U B R O U T I N E ¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦
08048614 ; Attributes: bp-based frame
08048614                 public handle_echo_command
08048614 handle_echo_command proc near ; CODE XREF: handle_incoming_message+1Ap
08048614 var_EC          = byte ptr -0ECh
08048614 var_E8          = byte ptr -0E8h
08048614 var_4           = dword ptr -4
08048614 arg_0           = dword ptr  8
08048614 arg_4           = byte ptr  0Ch
08048614                 push    ebp
08048615                 mov     ebp, esp
08048617                 push    ebx
08048618                 sub     esp, 0F4h       ; Integer Subtraction
0804861E                 mov     dl, [ebp+arg_4]
08048621                 cmp     dl, 0DFh        ; Compare Two Operands
08048624                 ja      short loc_804865A ; Jump if Above (CF=0 & ZF=0)
08048626                 push    [ebp+arg_0]
08048629                 lea     ebx, [ebp+var_E8] ; Load Effective Address
0804862F                 mov     [ebp+var_EC], dl
08048635                 push    (offset aMsgS+5) ; char *
0804863A                 push    0E0h            ; size_t
0804863F                 push    ebx             ; char *
08048640                 call    _snprintf       ; Call Procedure
08048645                 pop     edx
08048646                 mov     dl, [ebp+var_EC]
0804864C                 pop     ecx
0804864D                 movzx   edx, dl         ; Move with Zero-Extend
08048650                 push    edx             ; __int16
08048651                 push    ebx             ; void *
08048652                 call    send_outgoing_message ; Call Procedure
08048657                 add     esp, 10h        ; Add
0804865A loc_804865A:                            ; CODE XREF: handle_echo_command+10j
0804865A                 mov     ebx, [ebp+var_4]
0804865D                 leave                   ; High Level Procedure Exit
0804865E                 retn                    ; Return Near from Procedure
0804865E handle_echo_command endp

0804865F ; ¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦ S U B R O U T I N E ¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦¦
0804865F ; Attributes: bp-based frame
0804865F                 public handle_log_command
0804865F handle_log_command proc near ; CODE XREF: handle_incoming_message+2Dp
0804865F
0804865F var_88          = dword ptr -88h
0804865F var_78          = byte ptr -78h
0804865F var_4           = dword ptr -4
0804865F arg_0           = dword ptr  8
0804865F                 push    ebp
08048660                 mov     ebp, esp
08048662                 push    ebx
08048663                 sub     esp, 78h        ; Integer Subtraction
08048666                 mov     ebx, [ebp+arg_0]
08048669                 lea     eax, [ebp+var_78] ; Load Effective Address
0804866C                 push    ebx
0804866D                 push    offset aMsgS    ; "msg: %s"
08048672                 push    eax             ; char *
08048673                 call    _sprintf        ; Call Procedure
08048678                 pop     eax
08048679                 pop     edx
0804867A                 push    2               ; __int16
0804867C                 push    offset aOk      ; "OK"
08048681                 call    send_outgoing_message ; Call Procedure
08048686                 mov     [esp+88h+var_88], ebx
08048689                 call    _puts           ; Call Procedure
0804868E                 add     esp, 10h        ; Add
08048691                 mov     ebx, [ebp+var_4]
08048694                 leave                   ; High Level Procedure Exit
08048695                 retn                    ; Return Near from Procedure
08048695 handle_log_command endp
```

מס' מחברת: _ _

ג. [5 נק'] בהינתן שיופעל קוד הפיתון הבא, הסבירו כיצד תראה המחסנית בעת הכניסה לפונקציה `send_outgoing_message`:

```
import remoteproto
result = remoteproto.send_and_recv("E\xdfHello\x00")
print result
```

| תוכן | אורך בבתים | Offset from ebp |
|---|---|---|
| arg_0: msg | 4 | +8 |
| return address | 4 | +4 |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

ד. [5 נק] מה תציג הפעולה `print result` (ניתן לתאר במילים) ?

מס' מחברת: _ _

ה. [5 נק] כתבו רצף פקודות Pythonאשר יחלץ מידע כלשהו על מרחב הכתובות של התוכנית (ראשית, הבן את תוכן המחסנית בעת הקריאה ל-puts).

ו. [5 נק] בהנחה שברשותך ספרית libc באותה הגרסא כמו בשרת המרוחק, ותוך הידיעה כי puts מסתיים בקוד הבא

```
call _write
add esp, 0ch
pop ebp
retn
```

תארו בצורה מדויקת באמצעות קוד פייתון כיצד תוכל לקבל כתובת כלשהיא אשר תשמש כ"עוגן" לתוך libc

ז. [5 נק] השתמשו בכתובת העוגן על מנת להגיע לכתובות של system, exit על מנת ליצור קוד פייתון המעצב את הקלט כך שיכיל ROP shellcode אשר מריץ את המחרוזת:
`"echo user::0:0::/:/bin/sh >> /etc/passwd"`
הניחו כי המחרוזת הרצויה נמצאית מבעוד מועד בכתובת `0xDEADBEEF`.
לצורך השאלה השתמשו במשתני פייתון שיקראו diff_ret_system, diff_ret_exit שיחזיקו הפרש בקבועים – תנו הסבר מה צריכים להכיל המשתנים הללו.

ח. [5 נק] לולא היתה ידועה לנו הכתובת `0xDEADBEEF`, הסבירו עקרונית כיצד אפשר היה לעצב את הקלט כך שתוכל בכל זאת להריץ את המחרוזת באמצעות `system`

מס' מחברת: _ _

## man snprintf / man puts

```
PRINTF(3)                 Linux Programmer's Manual
PRINTF(3)

NAME
       printf,   fprintf,  sprintf,  snprintf,  vprintf,  vfprintf,
vsprintf,
       vsnprintf - formatted output conversion

SYNOPSIS
       #include <stdio.h>

       int printf(const char *format, ...);
       int fprintf(FILE *stream, const char *format, ...);
       int sprintf(char *str, const char *format, ...);
       int snprintf(char *str, size_t size, const char *format, ...);

DESCRIPTION
       The functions in the printf() family produce output according to a
forâ
       mat as described below.  The functions  printf()  and  vprintf()
write
       output  to stdout, the standard output stream; fprintf() and
vfprintf()
       write  output  to  the  given  output  stream;  sprintf(),
snprintf(),
       vsprintf() and vsnprintf() write to the character string str.

       The  functions  snprintf()  and  vsnprintf()  write  at most size
bytes
      (including the terminating null byte ('\0')) to str.
```

```
PUTS(3)                   Linux Programmer's Manual
PUTS(3)
NAME
       fputc, fputs, putc, putchar, puts - output of characters and strings

SYNOPSIS
       #include <stdio.h>

       int fputc(int c, FILE *stream);

       int fputs(const char *s, FILE *stream);

       int putc(int c, FILE *stream);

       int putchar(int c);

       int puts(const char *s);

DESCRIPTION
       fputc() writes the character c, cast to an unsigned char, to stream.

       fputs()  writes  the  string  s to stream, without its terminating null
       byte ('\0').

       putc() is equivalent to fputc() except that it may be implemented as  a
       macro which evaluates stream more than once.

       putchar(c); is equivalent to putc(c, stdout).

       puts() writes the string s and a trailing newline to stdout.

       Calls  to the functions described here can be mixed with each other and
       with calls to other output functions from the  stdio  library  for  the
       same output stream.
```

מס' מחברת: _ _

## שאלה 4 (15 נק')

במטוס הותקנה רשת WiFi לרווחת הנוסעים. כאשר מתחברים ממחשב נייד לרשת ומנסים לגשת לאתר HTTP בדפדפן, מופיע תפריט המציע גלישה באינטרנט בתשלום למשך שעה, וכן מבחר מוצרי דיוטי-פרי, וטופס למילוי פרטי כרטיס אשראי לתשלום. מי שרוכש את שירות הגלישה יכול, במשך השעה, להתחבר לאינטרנט בכל פרוטוקול – גם לאתרים דרך הדפדפן וגם לשירותים אחרים כמו משחקי און-ליין שאינם משתמשים בפרוטוקול HTTP. לאחר השעה, כל התקשורת מאותו מחשב נייד נחסמת, וניסיון לגלוש בדפדפן לאתר HTTP כלשהו מעלה שוב את התפריט, עד לתשלום נוסף.

א. [5 נק'] כיצד, לדעתך, מערכת המטוס מזהה מי זכאי לגלוש ומחליטה למי להציג את תפריט התשלום?

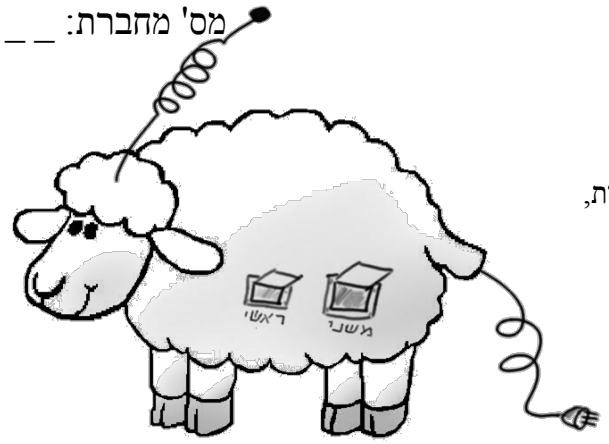
ב. [5 נק'] מישהו במחלקה הראשונה שילם על החיבור, ואחרי כמה דקות כיבה את המחשב שלו ונרדם. כיצד אפשר "לגנוב" את החיבור שלו ולגלוש ללא תשלום (בלי לקום מהכיסא)?

מס' מחברת: _ _

בדף התפריט יש שדה "כינוי משתמש", והמשתמש יכול להזין לתוכו כל כינוי שיחפוץ, נאמר "Moose", שנשמר בשרת המערכת לאורך זמן. כאשר המשתמש יישלח מחדש לדף התשלום כעבור שעה, הוא יבורך בברכת "Welcome Back, Moose". המימוש נאיבי ואינו בודק תווים מיוחדים בכינוי אשר הוזן.

ג. [5 נק'] כיצד ניתן לגרום לכך שנוסע המחלקה הראשונה מסעיף ב', כאשר יתעורר וינסה לשלם על שעת גלישה נוספת, בעצם (שלא בכוונה) ירכוש פריט דיוטי-פרי מהתפריט?

מס' מחברת: _ _




## שאלה 5 (20 נק')

מערכת הפעלה חדשה, אשר פותחה עבור כבשים חשמליות, תומכת בשני התקני אחסון: ראשי ומשני (ראו תרשים).

ההתקן הראשי הוא חלק מלוח האם של מחשב הכבשה, בעוד שההתקן המשני הוא תוספת הניתנת לפירוק. הכבשה תחזיק בהתקן האחסון המשני מידע רגיש (למשל תמונות אישיות מביכות).

א. [4 נק'] התעורר חשש שהתקן האחסון המשני יפורק וייגנב. כיצד מערכת ההפעלה יכולה להגן על פרטיות הנתונים במקרה זה, באמצעי קריפטוגרפי פשוט ושקוף למשתמש? הסבירו באיזה פרימיטיב קריפטוגרפי תשתמשו ומה יאוכסן בהתקן הראשי ובהתקן המשני.

ב. [6 נק'] התעורר חשש שהכבשה החשמלית תיחטף ותתוכנת מחדש, באופן שכאשר תוחזר לבעליה היא תעלה לאתר אינטרנט את כל המידע המאוכסן בהתקן האחסון המשני. בעת החטיפה כבשה תנותק מהחשמל ולכן המחשב שבה יכבה, אבל הגנבים יכולים להפעילה מחדש, ואפילו להחליף ולקרוא כרצונם את תוכן התקני האחסון. לכבשה יש עור עבה, ולכן הגנבים לא יכולים לעשות בה שינויי חומרה אחרים.

כיצד ניתן בכל זאת להגן על פרטיות המידע שבהתקן המשני?

ג. [7 נק'] למניעת גניבות וחטיפות, הכבשים החשמליות הועברו לדירים מאובטחים היטב. אבל הפושעים מצאו שיטה חדשה לגניבת מידע: פיתוח אפליקציות זדוניות למערכת ההפעלה של הכבשים. כך התגלה, לדוגמה, שאפליקציה פופולרית, אשר התיימרה רק להשמיע קולות פעייה, בעצם אוספת את כל הקבצים מהתקני האחסון המשני ושולחת אותם לפושעים דרך הרשת. נדרש מנגנון המפריד בין האפליקציות, כך שמידע הנכתב על ידי אפליקציה אחת לא יהיה ניתן לקריאה על ידי אף אפליקציה אחרת.

התקן האחסון הראשי תומך במערכת קבצים בסגנון Unix מסורתי. התקן האחסון המשני תומך רק במערכת הקבצים FAT, אשר אינה כוללת שום מנגנון הרשאות גישה: כל תהליך וכל משתמש מקומי יכול לגשת לכל קובץ במערכת הקבצים הזו.

כיצד ניתן לבצע את ההפרדה, בלי לשנות את ליבת מערכת ההפעלה (ובפרט את מערכות הקבצים)? מה יאוכסן והיכן, ואיזה רכיב מערכת ישתנה?

ד. [3 נק'] האם המנגנון שתארת בסעיף הקודם הוא Mandatory Access Control או Discretionary Access Control, ומדוע?