מס' מחברת: _ _

ת"ז:

אוניברסיטת תל-אביב
מדעי המחשב

מבוא לאבטחת מידע
מועד ב', תשע"ה
7.9.2015

(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 4 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.	2.	3.	4.
(א)	(א)	(א)	(א)
(ב)	(ב)	(ב)	(ב)
(ג)	(ג)	(ג)	(ג)
(ד)	(ד)	(ד)	
(ה)	(ה)	(ה)	
(ו)	(ו)		
(ז)			

מס' מחברת: _ _

שאלה 1 [31 נק']

האתר http://example.com מהווה כר פורה להחלפת דוגמאות לשיעורי הבית. הוא משתמש ב- cookie-based authentication. צוות הקורס החליט לפרוץ לחשבונות משתמשים כדי למצוא ראיות לדבר עבירה. הפריצה נחשבת להצלחה אם היא מאפשרת התחזות מלאה למשתמש בפני האתר.

להלן סדרת תרחישים וסוגי התקפות שצוות הקורס ינסה. נתחו כל תרחיש כדלקמן:

- אם יש שם מקובל לסוג זה של התקפה, ציינו אותו.
- אם קיימת התקפה מהסוג המתואר שאכן תצליח עבור מימוש (אולי נאיבי) של האתר:
  - הסבירו כיצד לבצע את ההתקפה ותחת אילו הנחות תצליח.
  - הציעו מימוש משופר של האתר החסין בפני ההתקפה הספציפית, והסבירו מדוע הוא חסין.
- אחרת: הסבירו מדוע.

א. [4 נק'] ה-cookies הם SHA-1 של סיסמת המשתמש. התוקפים מנחשים את ה-cookie.

ב. [4 נק'] התוקפים מאזינים (פסיבית) לתקשורת על רשת האוניברסיטה.

מס' מחברת: _ _

ג. [5 נק'] התוקפים משנים את אתר הקורס http://course.cs.tau.ac.il/infosec15 כך שיכלול קוד JavaScript אשר, כאשר הוא רץ בדפדפן של הסטודנטים, שולח את ה-cookie שלהם באתר הדוגמאות לצוות הקורס.

ד. [4 נק'] התוקפים שולחים לכל הסטודנטים דואל ובו קישור עם הטקסט "הודעה חשובה", אשר מוביל לכתובת http://example.com/change_pwd.php?newpw=12345.

ה. [4 נק'] לתוקפים יכולת לשנות רשומות בשרת ה-DNS של האוניברסיטה.

מס' מחברת: _ _

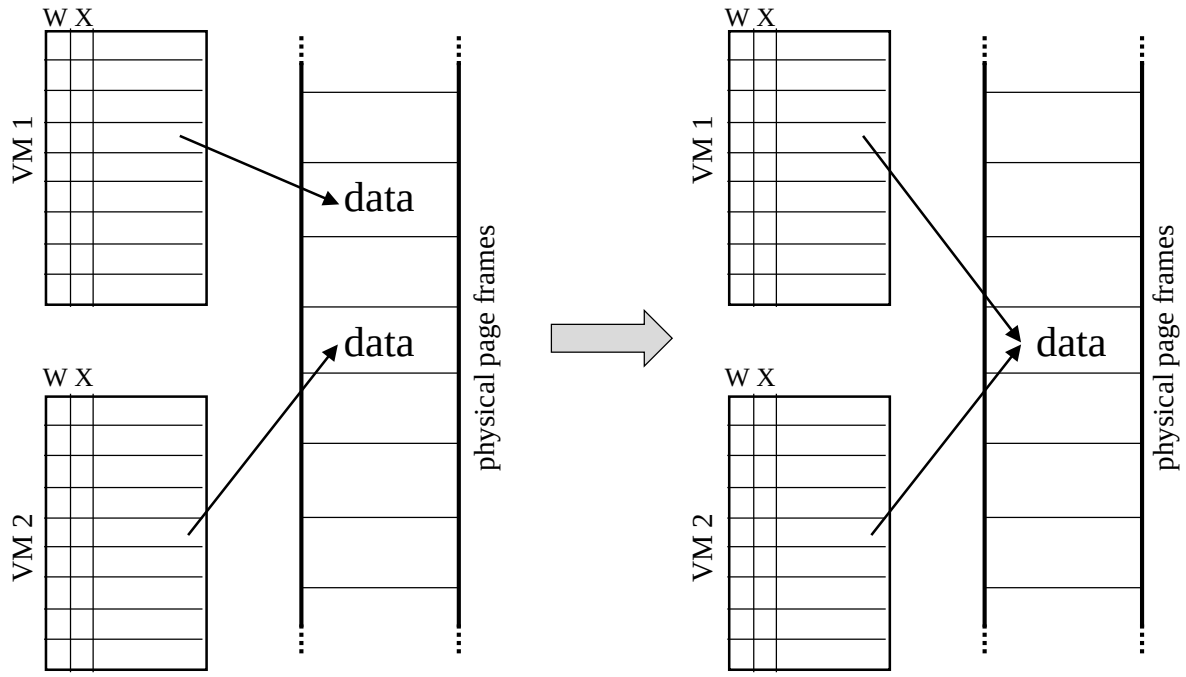
ו. [5 נק'] התוקפים יוצרים משתמש פיקטיבי באתר, ושולחים בשמו הודעות. הם מגלים ש-<סוגריים משולשים> בהודעות מועברים כלשונם, ומחליטים לנצל זאת.

ז. [5 נק'] מנהלי האתר הוסיפו תמיכה ב-HTTPS, אבל השתמשו בגרסה ישנה מאד של ספריית OpenSSL.

מס' מחברת: _ _

שאלה 2 [28 נק']

במחשבים המריצים מכונות וירטואליות, קיים לעיתים בזבוז גדול של זיכרון פיזי כי כל מכונה וירטואלית כוללת עותק משלה של מערכת ההפעלה, ספריות, אפליקציות וכו'. מימוש מתקדם של מכונות וירטואליות כולל מנגנון איחוד דפים משוכפלים (page deduplication), הפועל כך. מדי שנייה ה-hypervisor מתעורר ומחפש זוגות של דפי זיכרון אשר מכילים בדיוק אותו תוכן אבל שייכים למכונות וירטואליות שונות. כאשר הוא מזהה זוג כזה, הוא מאחד את הדפים, כך שבזכרון הפיזי נשמר דף יחיד עם התוכן, ומצביעים אליו מכמה מקומות בטבלאות הדפים הממפות את מרחב הכתובות של המכונות הווירטואליות למרחב הזיכרון הפיזי. בטבלאות הדפים האמורות, דפים מסומנים בביט Writable ובביט eXecutable. ראו איור.



בכל התרחישים שלהלן, תוקף זדוני מסוגל להריץ קוד כרצונו במכונה וירטואלית VM1 , והוא מנסה להתקיף קוד במכונה וירטואלית VM2 , הרצה על אותה מכונה פיזית. הניחו שהתוקף מכיר את הקוד המותקף.

א. [5 נק'] נאמר שמנגנון איחוד הדפים משמר את הביטים X ו-W בטבלאות הדפים האמורות. כיצד יכול התוקף לבצע התקפת control hijacking על מכונה וירטואלית אחרת, דרך מנגנון איחוד הדפים? הניחו שבמכונה המותקפת אין אמצעי הגנה כגון W^X או ASLR.

ב. [4 נק'] כעת הניחו שמנגנון איחוד הדפים מוודא שהביטים X ו-W תואמים בכל השורות בטבלאות הדפים המצביעות לדף המאוחד (ולעולם לא ישונו). כמו כן הניחו שהמכונה המותקפת ממשת X^W לכל התהליכים. כיצד, עדיין, יכולה מכונה וירטואלית אחת לבצע התקפת control hijacking על מכונה וירטואלית אחרת, דרך מנגנון איחוד הדפים?

כדי לפתור את ההתקפות שלעיל, מימושים אמיתיים של מנגנון איחוד דפים כוללים מגנון Copy on Write (CoW), כדלקמן.

כאשר ה-hypervisor מאחד דפים, הוא תמיד מסמן את השורות הרלוונטיות בטבלאות הדפים כ- Writable=0. אם בעתיד אחת המכונות הוירטואליות מנסה לכתוב לדף, תיוצר פסיקת תוכנה אשר תטופל על ידי ה-hypervisor. בהנחה שהדף היה Writable לפני האיחוד, ה-hypervisor מקצה עותק נפרד של הדף עבור המכונה הזו, מעדכן את טבלת הדפים (כלומר מבטל את האיחוד), מסמן את העותק כ-Writable=1, ומבצע את הכתיבה רק לעותק. עם השלמת התהליך, פסיקת התוכנה מסתיימת והמכונה הכותבת ממשיכה את ריצתה.

התוקף משער שבמכונה וירטואלית השנייה, המותקפת, יש דף המכיל תוכן ספציפי (נאמר המחרוזת "מתקיפים עם שחר" מרופדת באפסים לגודל דף). הוא רוצה לבדוק השערה זו, תוך ניצול מנגנון איחוד הדפים.

ג. [5 נ'] כיצד ניתן לממש התקפה זו?

מס' מחברת: _ _

(על הסעיפים הבאים אפשר לענות גם אם לא מצאת מימוש להתקפה.)

ד. [3 נ'] מה המונח הכללי להתקפה מסוג זה?

ה. [5 נק'] המכונה הוירטואלית המותקפת מריצה מערכת המממשת Mandatory Access Control האוכף מדיניות Information Flow Control. מה ההשפעה של ההתקפה האמורה על אכיפת המדיניות?

ו. [6 נק'] המכונה המותקפת משתמשת במנגנון ASLR, כדי להקשות על התקפות control hijacking. כיצד התוקף (המסוגל להריץ קוד במכונה וירטואלית אחרת על אותה מכונה פיזית) יכול לנטרל את האפקטיביות של ה-ASLR? כמה זמן זה יקח, בערך?

מס' מחברת: _ _

שאלה 3 (25 נק')

בשרת לינוקס מרוחק (המבוסס מעבד Intel x86 32bit) התגלתה חולשה בשירות בעל הרשאות root. החולשה קיימת בפונקציה הבאה, אשר אנו יכולים לשלוט בקלט שלה ע"י שליחת נתונים:

```
void skip_n_bytes_from_fd(int fd, int n)
{
      char buf[100];
      // if (n > 100) return;
                  read(fd, buf, n);
}
```

א. [5 נק'] שרטטו את מבנה המחסנית (יחסית לרגיסטר EBP) רגע אחרי הקריאה ל read בקוד האמור, באופן הכי מדויק שתוכלו. הניחו שאין הגנות מחסנית מכל סוג שהוא.

הנתון הנמצא בכתובת	גודל הנתון הנמצא בכתובת	כתובת בסיס יחסית ל-EBP (נא למיין: כתובות גדולות למעלה)

ב. [3 נק'] מהו ה- n המינימלי שעלינו לדרוש כדי שנוכל לבצע control hijacking? מדוע?

מס' מחברת: _ _

ג. [5 נק'] בהנחה שהערך של EBP לפני הקריאה ל-read נמצא בין `0xbfff0000` ל-`0xbfff0020`, רשום רצף בתים ב-hex שעלינו לשלוח, כך שה-shellcode הבא (שאורכו 30 בתים) ירוץ בצורה אמינה ככל שניתן.

```
'\xeb\x0f\x5b\x31\xc0\xb0\x0b1\xc9\xcd\x80\x31\xc0\xb0\x01\xcd\x80\xe8\xec\xff\xff\xff
/bin/sh\x00'
```

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

לנוחותך, ניתן לסמן ע"י קו רציף בתים חוזרים .לדוגמא תיאור מחרוזת HELLLLLLLLLLLLLLLOO:

'H'	45	'L'	→	→	→	→	→	→	→	→	→	→	→	→	→
→	→	'O'	'O'												

מס' מחברת: _ _

ד. [6 נק'] גילינו כי בשרת מופעל DEP והדרך היחידה לפעול היא באמצעות ROP. כתוב shellcode (כרצף בתים) מסוג ROP אשר משתמש ב-system כדי להוסיף לקובץ /etc/passwd משתמש בעל הרשאות root.

- השתמשו בפקודה : echo "myuser:0:0:::/:/bin/sh" >> /etc/passwd
- עומדים לרשותכם הגדג'טים ברשימה שבסוף השאלה.
- עומד לרשותכם כל המידע מהסעיפים קודמים, ובפרט אודות EBP.
- אין צורך לצאת בצורה מסודרת בסוף ה-shellcode.

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

מס' מחברת: _ _

ה. [6 נק'] כתבו רצף בתים אשר מנצל את ה-file descriptor הידוע לנו ומחבר את התוקף לshell על השרת המרוחק.

- חשוב: הניחו ש-EBX מכיל את הfile descriptor בזמן היציאה מהפונקציה skip_n_bytes_from_fd
- לרשותכם שוב רשימת גדג'טים שתוכלו להסתייע בהם.
- אין צורך לצאת בצורה מסודרת בסוף ה-shellcode.

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

מס' מחברת: _ _

רשימת גדג'טים זמינים

```
.text:B7523F01 5C | pop     esp
.text:B7523F02 C3 | ret
```

```
.text:B752B025 59    | pop     ecx
.text:B752E026 C3    | ret
```

```
.text:B752E030 58 | pop     eax
.text:B752E031 C3 | ret
```

```
.text:B752E040 5B    | pop     ebx
.text:B752E041 C3    | ret
```

```
.text:B752F152 5A | pop     edx
.text:B752F153 C3 | ret
```

```
.text:B7523868 CD 80 | int     0x80
.text:B752386A C3    | ret
```

```
.text:B752EC77 54 | push     esp
.text:B752EC78 C3 | ret
```

```
.text:B752EC87 54    | push     ebx
.text:B752EC88 C3    | ret
```

```
.text:B752599D 50 | push     eax
.text:B752599E C3 | ret
```

```
.text:B752CEA0 5A    | pop    edx
.text:B752CEA1 5E    | pop    esi
.text:B752CEA2 5F    | pop    edi
.text:B752CEA3 C3    | ret
```

```
.text:B752DEB4 03 DC | add  ebx, esp
.text:B752DEB6 C3    | ret
```

```
.text:B752DEC0 83 EB 10 | sub ebx,
0x10
.text:B752DEC3 C3       | ret
```

```
.text:B752FFD4 89 5c 24 10 | mov dword ptr [esp+0x10], ebx
.text:B752FFD8 C3          | ret
```

```
.plt:080486C0                     ; int system(char *string)
.plt:080486C0                     system      proc near
.plt:080486C0 FF 25 44 B0 04 08                 jmp     ds:off_804B044  ; Indirect Near Jump
.plt:080486C0                     system      endp
```

```
.plt:08048610                      ; int execv(char *file, char *argv[] /* can be null */)
.plt:08048610                      execv     proc near               ;
.plt:08048610 FF 25 18 B0 04 08                 jmp     ds:off_804B018  ; Indirect Near Jump
.plt:08048610                      execv     endp
```

```
.plt:080486A0                      ; int dup2(int oldfd, int newfd)
.plt:080486A0                      dup2      proc near
.plt:080486A0 FF 25 3C B0 04 08                 jmp     ds:off_804B03C  ; Indirect Near Jump
.plt:080486A0                      dup2      endp
```

פסאודו קוד של ה- system call של dup2()

```
int dup2(int old_file_descriptor_num, int new_file_descriptor_num) {
__asm__("mov eax,0x3f");
__asm__("mov ebx, old_file_descriptor_num");
                                // ebx,ecx contain a number, not a pointer to a number.
__asm__("mov ecx, new_file_descriptor_num");
__asm("int 0x80");
}
```

פסאודו קוד של ה- system call של execve()

```
int execve(char *file, char *argv[] /*can be null */, char *envp[] /*can be null*/) {
__asm__("mov eax,0xb");
__asm__("mov ebx, file");// ebx contains a pointer to a file string.
__asm__("mov ecx, argv");// ecx contains pointer to a string array containing the arguments
__asm__("mov edx, envp");// ecx contains pointer to a string array containing the env vars.
__asm("int 0x80");
```

מס' מחברת: _ _

שאלה 4 [16 נק']

בהמשך לשאלה הקודמת: עם היוודע החולשה למתכנת של השירות הפגיע, שונה הקוד בשרת.
הקוד המקורי:

```
client_fd = accept(server_fd, &remote_addr, sizeof(remote_addr), 0);
.
.
.
skip_n_bytes_from_fd(client_fd, n);
```

גרסה שנייה:

```
client_fd = accept(server_fd, &remote_addr, sizeof(remote_addr), 0);
if (remote_addr.sin_addr.s_addr != 0x08080808) {
    close(client_fd);
    continue;
}
.
.
.
skip_n_bytes_from_fd(client_fd, n);
```

א. [3 נק'] מה משמעות השינוי שעשה המתכנת?

מס' מחברת: _ _

ב. [7 נק'] כיצד יכול תוקף להתגבר על השינוי ולנצל את החולשה מהשאלה הקודמת (לפחות בהסתברות לא-זניחה)?

יש לכלול הסבר מילולי של תהליכי התקיפה והסתברות ההצלחה, וגם לספק קוד Scapy אשר מאפשר ניצול של החולשה בתרחישים אלו.

בקוד ה-Scapy, ניתן להניח כי כי רצף בתים לניצול החולשה עצמה כבר נמצא במשתנה בשם `payload`.
תיעוד קצר של Scapy מצורף בהמשך השאלה.

מס' מחברת: _ _

Scapy תיעוד

Layers:

```
Ether(dst,src,type)
ARP(hwtype,ptype,hwlen,plen,op,hwsrc,psrc,hwdst,pdst)
IP(version,ihl,tos,len,id,flags,frag,ttl,proto,chksum,src,dst,option
s)
ICMP(type,code,chksum,id,seq)
TCP(sport,dport,seq,ack,dataofs,reserved,flags,window,chksum,urgptr,
options)
UDP(sport,dport,len,chksum)
DNS(id,qr,opcode,aa,tc,rd,ra,z,rcode,qdcount,ancount,nscount,arcount
,qd,an,ns,ar)
```

Functions:

sr(x, filter=None, iface=None, nofilter=0, *args, **kargs)

Send and receive packets at layer 3

srp(x, iface=None, iface_hint=None, filter=None, nofilter=0, type=3, *args, **kargs)

Send and receive packets at layer 2

sniff(count=0, store=1, offline=None, prn=None, lfilter=None, L2socket=None, timeout=None, opened_socket=None, *arg, **karg)

Sniff packets

send(x, inter=0, loop=0, count=None, verbose=None, realtime=None, *args, **kargs)

Send packets at layer 3

sendp(x, inter=0, loop=0, iface=None, iface_hint=None, count=None, verbose=None, realtime=None, *args, **kargs)

Send packets at layer 2

מס' מחברת: _ _

ג. [6 נק'] המפתח הוציא גרסה שלישית לקוד השירות. בגרסה זו בחר המפתח להגביל את הגישה לשירות כך שתתאפשר רק מהרשת המקומית: מבוצע סינון בקוד המאפשר תקשורת נכנסת רק מכתובות מקומיות, אשר לא ניתנות לניתוב באינטרנט.

כיצד יכול תוקף מרוחק (אבל מוכשר, בעל משאבים ובעל מוטיבציה) לנצל את החולשה בכל זאת? תארו שלושה ווקטורי התקפה שונים מהותית. ניתן להניח הנחות סבירות, בתנאי שהן מצוינות במפורש.