קודגורו אקסטרים

מה, איך, כיצד ומתי..

בשיתוף גבהים

יונתן פרי
Jonatan@codeguru.co.il

ברק גונן

דצמבר 2014

מקור: Pen Waggener\Flicker

קודגורו אקסטרים

רקע על התחרות

אתגר (חדש) מבית קודגורו

- **תחרות קבוצתית** בה כל קבוצה מפתחת (בשפת אסמבלי) תוכנה המשמשת כשחקן במהלך סבבי התחרות.
  - מותר להרשם בקבוצות של עד 5 משתתפים
  - אופי התחרות מקנה יתרון לקבוצה על פני משתתף יחיד.
- שחקן זה המכונה בשם "**שורד**" ותפקידו הוא פשוט – לשרוד את מהלך המשחק.
- ישנן טקטיקות רבות להשגת המטרה, נרחיב על כך בהמשך.

קהל היעד

- כולם!
  - בפרט: **אתם ואתן!**
- דרישות הסף: ידע בסיסי באסמבלי, מוטיבציה ורצון להתנסות בחווית משחק לא שגרתית.
- התחרות נערכת אחת לשנה, ובנוסף לה מתקיימות תחרויות וירטואליות
  - תחרות וירטואלית: תחרות בה לא נדרשת הגעה של צוות המשתתפים למקום התחרות. ההשתתפות נעשת ע"ג האינטרנט.

מבנה התחרות

- 25% מהציון - הרצת השורדים שישלחו עד כשבועיים לפני התחרות
- 25% מהציון - הרצת השורדים כפי שהגיעו ערב התחרות.
  - הריצה תתבצע בטקס חגיגי בשעה תשע בבוקר התחרות.
  - בטקס יחשפו כל השורדים.
- 50% מהציון - הרצת השורדים בשעה 12:00 - לאחר מקצה שינויים.
- ארבעת הזוכים בשיקלול שלושת השלבים הללו יעלו לגמר - שם מתחיל הניקוד מאפס.

מבנה התחרות- המשך

- בכל **תור** מנוע המשחק מריץ פקודת אסמבלי אחת מכל שורד, לאחר מכן עובר לשורד הבא וחוזר על כך באופן מעגלי
  - שורד שמנסה להריץ פקודה לא חוקית- נפסל והריצה שלו מופסקת
- **משחקון** מורכב ממספר קבוע של תורות (כמה עשרות אלפים). רק שורדים שהגיעו לסוף המשחקון מקבלים ניקוד. אם נותר שורד יחיד- המשחקון נפסק לפני כמות התורות המוגדרת.
  - מורצים מאות אלפי משחקונים, כאשר בתחילת כל משחקון מאותחלים כל הרגיסטרים ומוגרל מיקום חדש לשורדים.
- ציון ה**משחק** של כל שורד נקבע לפי הניקוד שהשורד השיג על פני מאות אלפי משחקונים.

לתלמידי גבהים

- התחרות מבוססת על חומר הלימוד בכיתות י'
  - ספר הלימוד וסביבת העבודה באתר גבהים
- אנו מעודדים אתכם להשתתף
- אל תצפו לזכות בתחרות בנסיון הראשון... זיכרו: המטרה העיקרית היא הנאה וצבירת נסיון.
- לעזרתכם עומדים מורים ועוזרי ההוראה.
  - התחרות איננה בין המורים ועוזרי ההוראה בבתי הספר :)
  - תקבלו את כל העזרה בהבנת הנושאים שמכוסים במצגת זו ובמסמך "המדריך לקודגורו אקסטרים גרסת גבהים", אך לא מעבר לכך.

קודגורו אקסטרים

מנוע, זירה, שורד

מנוע המשחק

- **זירה וירטואלית** בה מתרחש המשחק בין מספר תוכניות.

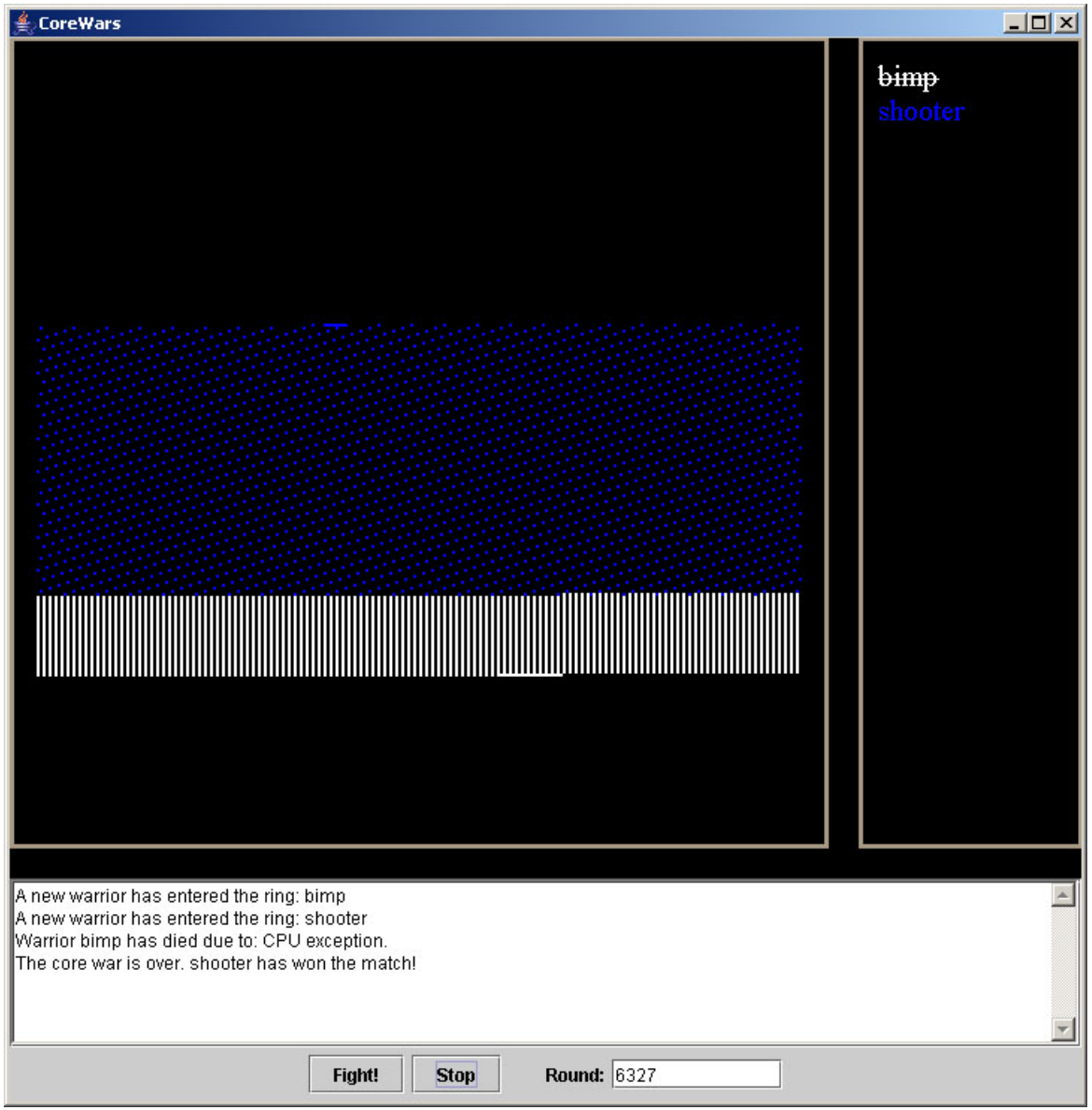
  כל תוכנית (הנקראת גם "שורד") נכתבת בשפת אסמבלי 8086, נטענת לזירת המשחק ומורצת במקביל לשאר התוכניות.

  פגיעה בשורד אחר נעשת על ידי כתיבת פקודה לא חוקית לאזור בו נמצא אותו שורד.

  כאשר מנוע המשחק ינסה להריץ את הפקודה השמורה באותו אזור – הוא יריץ פקודה לא חוקית. אותו שורד אשר עבורו הורצה הפקודה – יפסל.

- מנוע המשחק מריץ את השורדים בזירה הוירטואלית מספר רב של הרצות, בכל הרצה מחולק הניקוד בין השורדים שהגיעו לסוף ההרצה "חיים".

זירת המשחק



- 256 שורות- 00 עד FF
- 256 פיקסלים בשורה
- סה"כ 65,536 פיקסלים

	00	01			
00	0000	0001			
01	0100	0101			

שורד

- מטרת כל שורד להגיע לסוף הסיבוב, לשלב חלוקת הנקודות, ולמנוע משורדים אחרים לעשות זאת.
- טקטיקה: לאורך השנים נכתבו שורדים רבים, ניתן לסווג אותם:
  - **התקפיים** – שורדים אשר יעודם לכתוב כמות גדולה של פקודות לזיכרון המשחק במספר תורות מצומצם ככל שניתן, וזאת ע"מ להביס את היריבים.
  - **הגנתיים** – שורדים אשר יעודם להתחמק מפגיעה של יריבים על ידי טכניקות שונות.

קודגורו אקסטרים

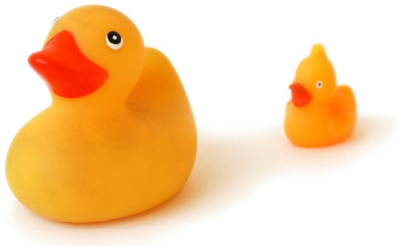
כתיבת שורד

קובץ שלד לכתיבת שורד

```
1  IDEAL
2  MODEL tiny
3  CODESEG
4  org 100h
5  start:
6  ;----- sored code -----;
7  END start
```

- נביט בקטע הקוד הבא.
  בשונה מקובץ שלד "רגיל"
  (base.asm) בקובץ זה:
  - מודל הזיכרון הוא tiny- מתאים לקבצי com
  - אין DATASEG
  - הגדרת org 100h- כנדרש מקובץ com

כתיבת השורד הראשון



מקור: Robert Francis\Flicker

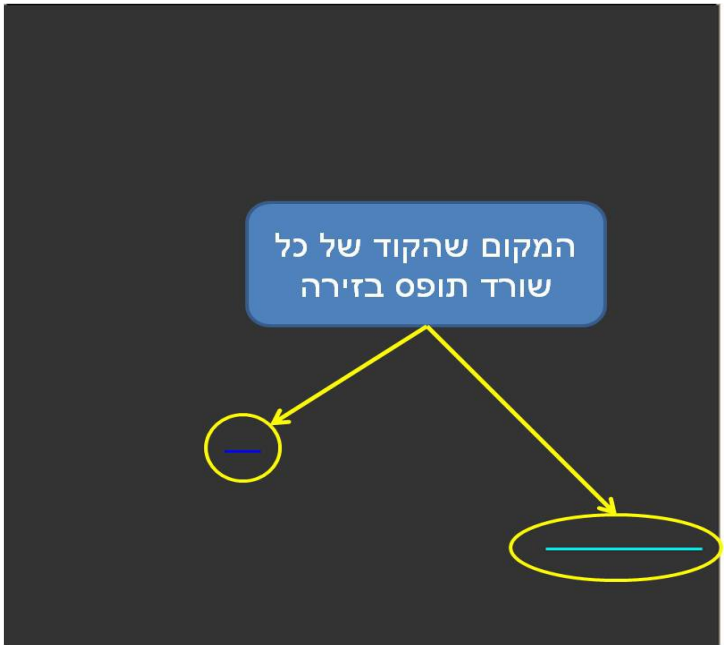
- נביט בקטע הקוד הבא

```
start:
    jmp start
```

- שורד זה יבצע פעולה אחת בכל תור, לאורך כל המשחק – הוא "יעמוד במקום"
- התוכלו לחשוב על יתרונותיו של שורד זה? על חסרונותיו?

מיקום התחלתי בזירה - כללים

- מיקומו ההתחלתי של כל שורד בזירה הוא אקראי, ונקבע על ידי מנוע המשחק לפני כל סיבוב
- ניתן לראות את המיקום של השורדים בזירה כ"קווים אופקיים" בזירה-המקומות שהם תופסים בזיכרון
- מנוע המשחק דואג לכך ששני שורדים לא ייטענו למקום זהה בזירה
- מנוע המשחק מאפשר לכל שורד להיות באורך של עד 512 בתים



תפיסת מקום בזירה

- נבצע בשורד שלנו שינוי, שיגרום לו לתפוס את המקום המקסימלי בזירה

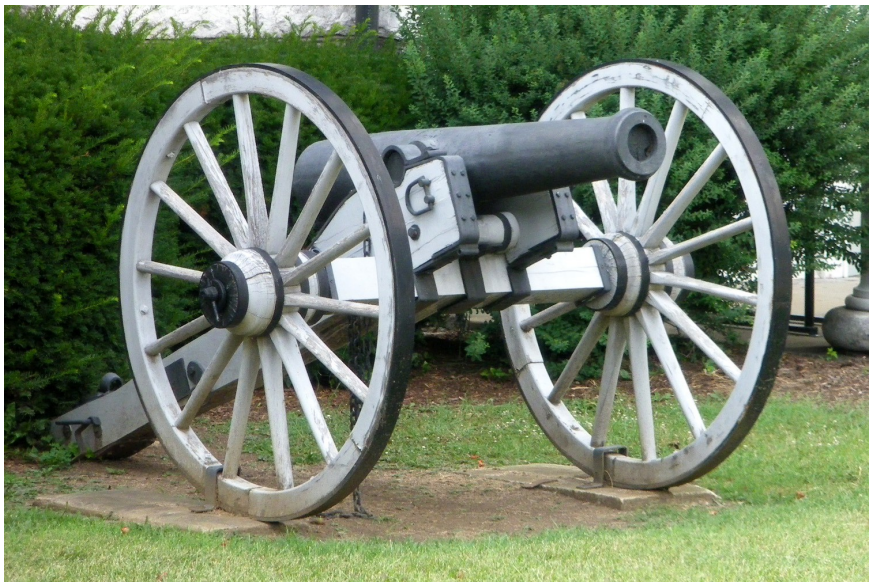
```
1  IDEAL
2  MODEL tiny
3  CODESEG
4  org 100h
5  start:
6      jmp start
7  exit:
8      db 512-(exit-start) dup (0cch)
9  END start
```

- Exit ו-start מצביעים על מקומות בקוד: start על ההתחלה ו-exit על הסיום. ההפרש ביניהם הוא גודל הקוד. הפקודה האחרונה מחשבת כמה בתים צריך "לרפד" עד הגודל המקסימלי של 512 בתים ומקצה מקום זה בסוף השורד.

כתיבת שורד "תותח"

- עתה נרחיב מעט את יכולותיו של השורד:
  - נסו להבין מה יבצע כעת השורד?
  - האם נראה לכם כי הוא ינצח את השורד הקודם?
  - כיצד אפשר לשפר את פעולותיו?

```
1  IDEAL
2  MODEL tiny
3  CODESEG
4  org 100h
5  start:
6      mov cx, 0cccch
7      mov bx,0
8  write_word:
9      mov [bx],cx
10     add bx, 2
11     jmp write_word
12 exit:
13 END start
```



מקור: Renee\Flicker

קודגורו אקסטרים

יכולות מיוחדות של שורדים

כתיבת שורד "מפציץ כבד"

- בשלב זה נציג יכולת נוספת של מנוע המשחק: **הפצצה כבדה**.
- הפקודה להפעלת הפצצה כבדה: **int 86h** (מוכרת רק למנוע המשחק, אל תנסו בסביבה אחרת :))
- הפצצה כבדה מאפשרת בפקודה אחת לכתוב 256 בתים לזירה המשחק! ניתן להשתמש פעמיים בלבד בכל סיבוב בהפצצה כבדה (עבור כל שחקן)
- בדוגמה זו, נשתמש ביכולת זו בכדי להפציץ את האזור הקרוב ביותר אלינו, ולאחר מכן נעבור לשיטה הישנה. כך נכתוב 512 בתים במהירות
  - לכמה מהלכי כתיבה רגילה שקולה "הפצצה כבדה"?

הפצצה כבדה- המשך

- "הפצצה כבדה" של איזור בזיכרון: כאשר הפקודה נקראת, מנוע המשחק יכתוב 64 פעמים את DX:AX לכתובת המתחילה ב-ES:DI (ובסה"כ יכתבו 256 בתים).
- הבתים יכתבו לפי הסדר הבא: AL, אח"כ AH, אח"כ DL ולבסוף DH - וחוזר חלילה.
- כיוון הכתיבה (קדימה או אחורה) נקבע לפי ערך ה-direction flag, וערכם של ES:DI לאחר הקריאה משתנה בהתאם.

- פרמטרים:
- ax,dx – הערך אשר יכתב.
- Es –המקטע אליו נכתוב, הערך משתנה בהתאם.
- di – ההיסט אליו נכתוב, הערך משתנה בהתאם
- Direction Flag – כיוון החיפוש
  - CLD –
  - STD –
- Int 86h – פקודת ההפעלה. ניתן לקרוא לה עד פעמים בסיבוב.

דוגמה לשורד "מפציץ כבד"

```
1  IDEAL
2  MODEL tiny
3  CODESEG
4  org 100h
5  start:
6      push cs
7      pop es
8      mov di,ax
9      add di, offset exit
10     mov dx, 0cccch
11     mov ax,dx
12     int 86h
13     int 86h
14 write_word:
15     mov [di],dx
16     add di, 3
17     jmp write_word
18 exit:
19 END start
```

כתיבת שורד "מפציץ חכם"

מקור: Pascal\Flicker

- לעיתים נראה כי ישנם שורדים יריבים המהווים איום ממשי על נצחוננו.
  - נוכל לטפל בהם בצורה "כירורגית"
- מנוע התחרות מקצה לכל שורד "**פצצת חכמה**" אחת בכל סיבוב.
- פצצה חכמה: פקודה המחפשת אחר מבנה נתון של 4 אופקודים, ומחליפה אותם במבנה אחר, לפי טווח חיפוש שאנו הגדרנו לה.
- הפקודה להפעלת הפצצה חכמה: **int 87h** (מוכרת רק למנוע המשחק, אל תנסו בסביבה אחרת :))
- בצורה כזאת, במידה וידוע לנו המבנה הבינארי של השורד היריב, נוכל לטפל בו באופן פרטני.

הפצצה חכמה

- "הפצצה חכמה" של איזור בזיכרון: כאשר הפקודה int 87h נקראת, מנוע המשחק מחפש את ההופעה הראשונה בסגמנט של 4 הבתים DX:AX, החל מהכתובת ES:DI (חיפוש הבתים הוא לפי הסדר הבא: AL, אח"כ AH, אח"כ DL ולבסוף DH).
- במידה והבתים נמצאו, המנוע יכתוב במקומם את 4 הבתים CX:BX. כיוון החיפוש בתוך הסגמנט נקבע לפי ערך ה-direction flag, אך ערכם של ES:DI לא משתנה בכל מקרה.

- פרמטרים:
- ax,dx – הערך אשר נחפש.
- es –המקטע ממנו תחל הפעולה.
- di – היסט ממנו תחל הפעולה.
- cx,bx – הערך אשר יכתב.
- Direction Flag – כיוון החיפוש.
- Int 87h – פקודת ההפעלה. ניתן לקרוא פעם בסיבוב.

ה"קורבן" המיועד לפצצה החכמה

```
1  IDEAL
2  MODEL tiny
3  CODESEG
4  org 100h
5  start:
6      mov     cx, 100
7  l:
8      call    smile
9      add     bx, 623h
10     loop    l
11 smile:
12     mov     [bx+2041h], al
13     mov     [bx+2045h], al
14     mov     [bx+2243h], al
15     mov     [bx+2340h], al
16     mov     [bx+2441h], al
17     mov     [bx+2542h], ax
18     mov     [bx+2444h], al
19     mov     [bx+2345h], al
20     ret
21 END start
```

- הקוד הבא מצייר סמיילים ברחבי המסך
- הפקודה mov [bx+2041h], al מיתרגמת לשפת מכונה: 88874120

דוגמה ל"מפציץ חכם"

- נסו להריץ את הקוד הבא יחד עם ה"קורבן" וראו איך ה"קורבן" מפסיד מהר מאד...

```
1  IDEAL
2  MODEL tiny
3  CODESEG
4  org 100h
5  start:
6      mov si, ax
7      mov di, 0
8      push cs
9      pop es
10     mov dx, 8788h    ; search for dx:ax "88874120"
11     mov ax, 2041h
12     mov cx,0cccch    ; replace with cx:bx "CCCCCCCC"
13     mov bx,cx
14     int 87h
15     mov bx,si
16     add bx, offset exit
17 write_word:
18     mov [bx],cx
19     add bx, 2
20     jmp write_word
21 exit:
22 END start
```

קודגורו אקסטרים

זומבים והשתלטות על זומבים

זומבים

- בכל תחרות משתתפים בזירת המשחק שורדים מיוחדים הנקראים "**זומבים**", שורדים אלו נכתבים על ידי צוות קודגורו ובעלי משימה מוגדרת.
- במהלך התחרות קוד הזומבים נמסר למשתתפים
- קוד הזומבים כולל חידה תכנותית / מתימטית

דוגמה לזומבי פשוט

- זומבי זה כותב את כתובת ההתחלה שלו לתוך התא 1234h

```
1  IDEAL
2  MODEL tiny
3  CODESEG
4  org 100h
5  start:
6      mov [1234h], ax
7      jmp start
8  END start
```

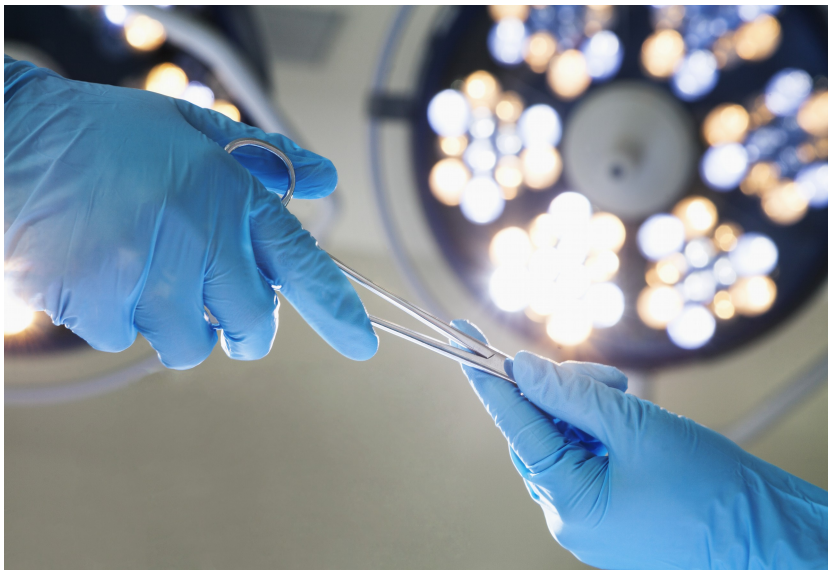
מקור: nolnet\Flicker

רעיון: השתלטות על זומבי

- פעמים רבות ראינו כי מתמודדים מנסים "להשתלט" על הזומבים על ידי כתיבת פקודות בינאריות לאזורי הזיכרון של אותם זומבים, מצב זה גורם לזומבי לבצע משימות שאותו מתמודד בוחר.

- שיטה זו מועילה לנו מכיוון שבעזרת אנו "מרוויחים" את זמן הריצה של השורד, ובנוסף לזאת את סט הפצצות שלו (בהנחה ולא השתמש בהם בעבר)
- חשוב לציין: זומבי מתפקד כשורד רגיל. מלבד העובדה שיעודו הוא (לרוב) לא לנצח במשחק, אלא לאפשר יתרון מסויים לאלו שהבינו את השיטה לשבייתו.
  - זומבי אינו מקבל ניקוד בסוף הסיבוב

דוגמה להשתלטות על הזומבי



- ננתח את הזומבי בעזרת דיבאגר:

```
cs:0100►A33412          mov     [1234],ax
cs:0103 EBFB            jmp     0100
```

- תרגום קוד הזומבי לשפת מכונה: A33412EBFB
- קוד זה תופס 5 בתים בזיכרון- חשוב להמשך

דוגמה להשתלטות על הזומבי

הרעיון הכללי:

- בסוף הזומבי נוסיף קוד שאומר לו לקפוץ אל השורד שלנו ולהריץ אותו
- לאחר הוספת הקוד נדרוס את פקודת הקפיצה שבסוף הזומבי המקורי
- מה יקרה אם נדרוס את פקודת הקפיצה לפני שכל הקוד שלנו מוכן בסוף הזומבי?

- הקוד שאנחנו רוצים שהזומבי יריץ הוא מהצורה:

```
Mov     si, xxxx      ; xxxx is the value of ax
Jmp     si
```

- הסבר: ax היא כתובת ההתחלה של השורד שלנו
- ע"י דיבאגר נמצא כי הקוד צריך להתחיל ב-BE, שני בתים של ax, אחר כך FFE6 (בדיבאגר הנחנו ax=0ABCDh)

```
cs:0100 BECDAB          mov   si,ABCD
cs:0103 FFE6            jmp   si
```

דוגמה להשתלטות על הזומבי

קוד שנשתל בסוף הזומבי ומורה לקפוץ אל נקודת ההתחלה שלנו:

```
mov bx,[1234h]
mov [byte bx+5], 0BEh
mov [word bx+6], ax
mov [word bx+8], 0E6FFh
```

- רק לאחר שהכנו את הקוד בסוף הזומבי, דריסת פקודת ה-jmp של הזומבי:

```
mov [byte bx+4], 0
```

דוגמה להשתלטות על הזומבי

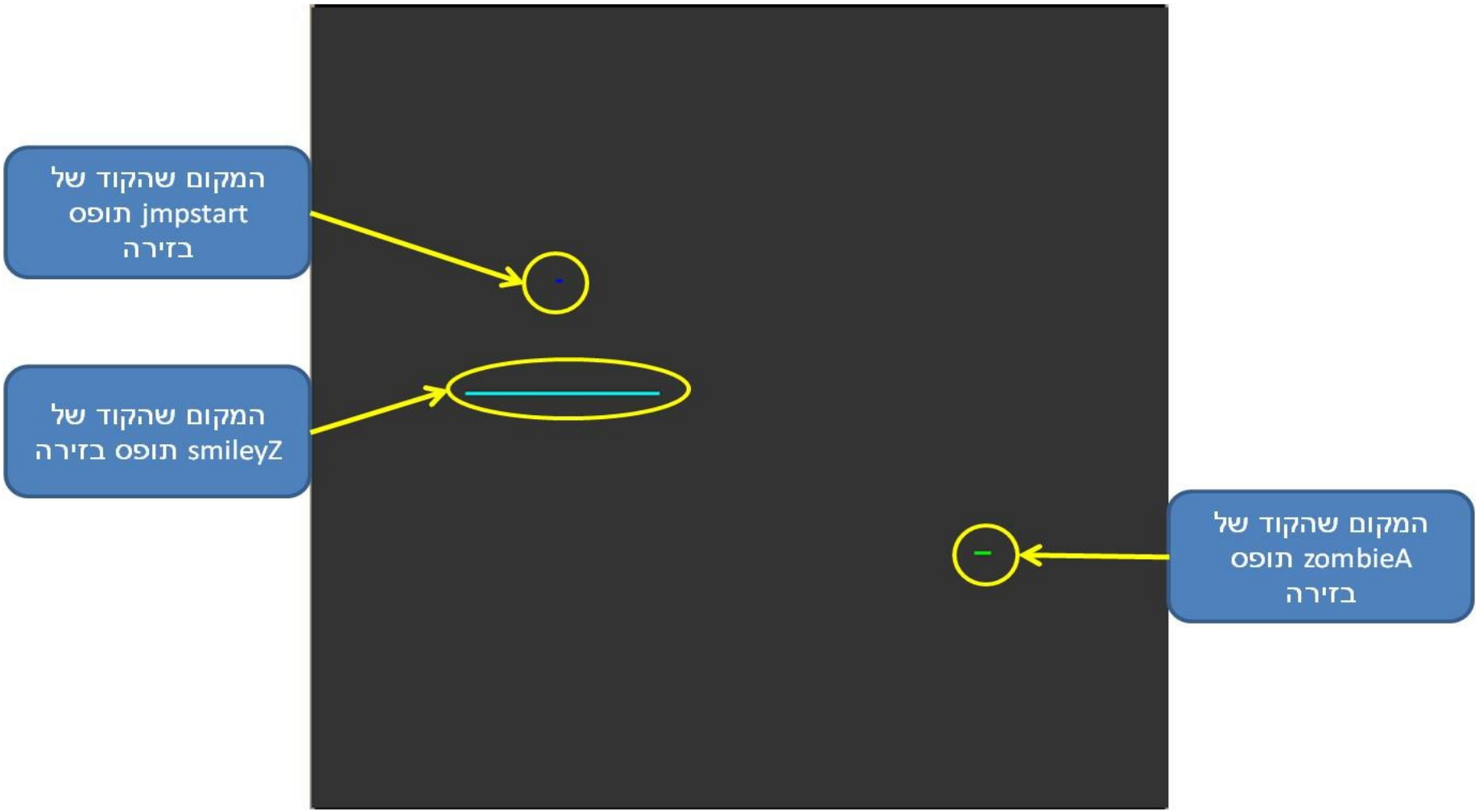
חיבור כלל הקוד לקוד השתלטות:

- יש לשים לב לשורה הראשונה- נועדה לאפשר לזומבי תור לכתוב את כתובת ההתחלה שלו ל-1234h, לפני שאנחנו משתמשים בערך שבכתובת זו

```
1  IDEAL
2  MODEL tiny
3  CODESEG
4  org 100h
5  start:
6  ; taking over the zombie
7      add bx, 1          ; let the zombie do the first move
8      mov bx,[1234h]
9      mov [byte bx+5], 0BEh
10     mov [word bx+6], ax
11     mov [word bx+8], 0E6FFh
12     mov [byte bx+4], 0
13 Sored_code:
```

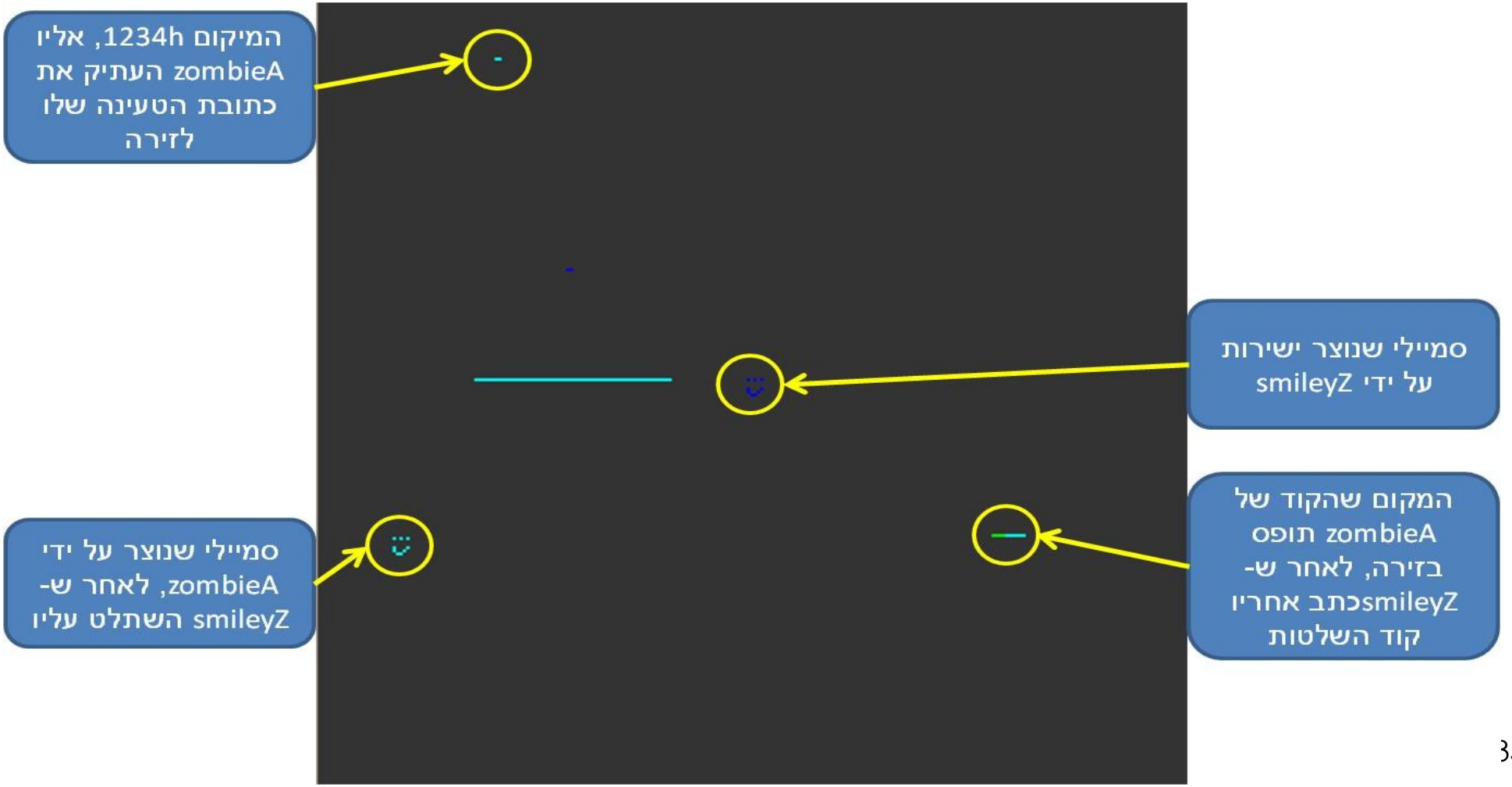
השתלטות על זומבי- המחשה בזירה

- מצב התחלתי: זומבי, שורד א' שמצייר סמיילי, שורד ב' מסוג jmp start



השתלטות על זומבי- המחשה בזירה

- המצב לאחר ההשתלטות:



קודגורו אקסטרים

טקטיקה מתקדמת: שכפול שורד

שכפול שורד

- **"שכפול שורד"** היא אסטרטגיה נפוצה בתחרויות קודגורו.
- מדובר בשיטה בה שורד "קורא" את עצמו מאזור זיכרון אחד לאזור זיכרון אחר.
- לשיטה זאת מספר יתרונות:
  - מיקום משתנה לאורך המשחק.
  - השורד עלול "לדרוס" את יריביו במהלך ההעתקה.
  - יצירת עותקים "מתים" רבים על גבי לוח המשחק.
- **חשוב להדגיש**: בכל רגע נתון, ישנו "מופע" אחד חי של השורד.

שכפול שורד

- שיטה זאת איננה חפה בעיות:
- פיתוח השורד נעשה קשה יותר ככל שהמורכבות עולה.
- יחס הזיכרון שנכתב לעומת יחס הפעולות איננו אופטימלי.
- נדרשת הבנה עמוקה יותר של של התוצר הבינארי.

שכפול שורד

```
1  IDEAL
2  MODEL tiny
3  CODESEG
4  org 100h
5  start:
6      push    cs
7      pop     es
8      xchg    di, ax
9      add     di, 12
10     mov     si, di
11     add     si, 10
12     std
13 myloop:
14     ; This part is copied, then jumped to
15     dec     di
16     dec     di
17     movsw
18     movsw
19     movsw
20     movsw
21     movsw
22     movsw
23     inc     di
24     inc     di
25     jmp     di
26 END start
```

- ראשית נכיר את הפעולה movsw: זאת מעבירה מילה מכתובת ds:si לכתובת es:di, ערכי האוגרים מתעדכנים בהתאם לכיוון הdirection flag.
- הקוד הבינארי שיווצר יכיל 12 בתים לפני התווית myloop ו-12 בתים אחריה.
- 6 פקודות ה-movsw יעתיקו את 12 הבתים שאחרי ה-myloop
- מכיוון שהאוגר di מתעדכן בהתאם, כל שנותר הוא לבצע jmp למיקום החדש.

קודגורו אקסטרים

טקטיקה מתקדמת: כתיבה בקצב כפול

Far Call

- הרעיון: בשפת אסמבלי, כאשר אנחנו קוראים לפונקציה בעזרת פקודת call, אנו דוחפים למחסנית **2 בתים** המציינים את מיקומנו טרם הקריאה.
- במידה וביצענו קריאת far לפונקציה – כלומר פונקציה הנמצאת במקטע זיכרון אחר, נדחפים **4 בתים** למחסנית.
- ... במידה ונחליט שזירת המתמודדים היא המחסנית שלנו, נוכל לדרוס ארבעה בתים בכל תור במקום שניים :)
  - מהם חסרונות השיטה הזאת?

דוגמה: שימוש ב Far Call

```
1  IDEAL
2  MODEL tiny
3  CODESEG
4  org 100h
5  start:
6      add ax,call_far-start
7      push es
8      pop ds
9      mov bx,50h
10     mov [bx+2],cs
11     mov [bx],ax
12     push cs
13     pop ss
14     mov sp,ax
15 call_far:
16     call [dword bx]
17 END start
```

- מקור הקוד ומידע נוסף:

  http://goo.gl/0zvLgU

  (מומלץ לגלוש מדפדפן כרום)

קודגורו אקסטרים

טקטיקה מתקדמת: שיתוף פעולה בין שורדים

שיתוף פעולה בין שורדים

- מנוע המשחק מאפשר לנו ליצור קבוצות הבנויות מזוג שורדים (בעלי שם זהה בתוספת הסיפרה 1 ו- 2 בסוף שמם).
  - לדוגמא: קבוצה בשם rocky תקרא לשורדים בשמות rocky1 ו-rocky2
- היתרון ברור לחלוטין: יותר אוגרים, יותר זמן משחק, יותר סיכוי לנצח!
- הניקוד מתחלק בין שני השורדים (אם רק אחד שרד- מחצית הניקוד)
- התקשורת בין השורדים נעשת בסגמנט אישי (es) בגודל 1024 בתים, סגמנט זה פרטי ואין אפשרות לשורדים מקבוצות יריבות לבצע בו כל פעולה.

מקור: Nick Royer\Flicker

דוגמה: cannon1.asm

```
1  IDEAL
2  MODEL tiny
3  CODESEG
4  org 100h
5  start:
6      stosw
7      mov     cx, 0cccch
8      mov     bx,ax
9      add     bx, stop-start
10 write_word:
11     mov     [bx],cx
12     add     bx, 2
13     jmp     write_word
14 stop:
15 END start
```

- נבחין כי ההבדל היחיד למול השורד cannon (המקורי) הוא הפקודה stosw שמאכסנת את הערך ax בכתובת es:di (כאשר di מאותחל ל 0)

דוגמה: cannon2.asm

```
1  IDEAL
2  MODEL tiny
3  CODESEG
4  org 100h
5  start:
6      push    es
7      pop     ss
8      mov     bp,0
9      mov     di,[bp]
10     mov     cx, 0cccch
11     mov     bx,di
12     dec     bx
13 write_word:
14     mov     [bx],cx
15     sub     bx, 2
16     jmp     write_word
17 END start
```

- שורד זה מורכב מעט יותר.
בראשיתו, אנחנו מגדירים את
המחסנית להצביע למקטע es
ומשתמשים בה לביצוע פעולות
תקשורת בין השורדים.

אתגר

- נסו לשלב בין יכולות השורדים, כלומר, ליצור זוג שורדים המעתקים עצמם למיקומים שונים ונמנעים מדריסה אחד של השני.

קודגורו אקסטרים

מקורות נוספים

כלים נוספים – חוברת הדרכה וספר לימוד

- המדריך, בגרסת "גבהים", נמצא באתר קודגורו אקסטרים
- המדריך כולל הוראות התקנה של כל התוכנות הנדרשות להרצת מנוע המשחק
- http://www.codeguru.co.il/xtreme/tutorial/cgxg.pdf
- קישור לספר לימוד אסמבלי של גבהים:
- http://www.cyber.org.il/assembly/gvahim_assembly_book.pdf

כלים נוספים - הפורום

- ניתן להיכנס לפורום האתר – דוגמאות רבות לשורדים נמצאים שם.
  - חלקם אפילו מתועד!
- חלק מהדוגמאות שהוצגו במצגת זו נלקחו מדפי הפורום, אלו עמוסים בשנים של ניסיון – נצלו אותם כדי ללמוד, וגם כדי לשאול שאלות
- קישור: http://goo.gl/49h62H (מומלץ לגלוש מדפדפן כרום)

כלים נוספים – ספר הפרצופים

- מקום המפגש של קודגורו-ים בעבר בהווה.
- בדף זה נפורסם עדכונים שוטפים במהלך התחרות.
- ניתן להתעדכן באתגרים חדשים מבית קודגורו, להשתמש בחוכמת ההמונים.
- קישור http://facebook.com/CodeGuru.IL

בהצלחה!

גבהים
חינוך • מצוינות • סייבר

התחרות החדשה
מבית קוד גורו
אקסטרים
Code Guru
EXTREME