קודגורו אקסטרים

בשיתוף גבהים

יונתן פרי

ברק גונן

קודגורו אקסטרים

רקע על התחרות

אתגר מבית קודגורו

- **תחרות קבוצתית** בה כל קבוצה מפתחת (בשפת אסמבלי) תוכנה, המשמשת כשחקן בסבבי התחרות
  - מותר להרשם בקבוצות של עד 5 משתתפים
  - אופי התחרות מקנה יתרון לקבוצה על פני משתתף יחיד
- שחקן זה מכונה בשם "**שורד**" ותפקידו 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 (אין צורך לדעת פרוצדורות ופסיקות)
  - מוטיבציה
  - רצון להתנסות בחווית משחק לא שגרתית

מבנה התחרות

- החל משנת 2016 המתמודדים יחולקו לבתים, באופן הבא:
  - בבוקר התחרות תבוצע ריצה מקדימה לקביעת הדירוג
  - 8 הקבוצות בדירוג הגבוה ביותר יעלו אוטומטית לשמינית הגמר
  - יתר הקבוצות יחולקו ל-8 בתים
  - לכל קבוצה יינתן זמן לשפר את השורד שלה ולהשתמש בפצצות חכמות
  - בתום הזמן, יורץ כל בית בנפרד
  - 8 הקבוצות שיזכו בראשות הבתים יעלו גם הן לשמינית הגמר
  - החל משלב שמינית הגמר אין מקצה שיפורים נוסף, השורדים מורצים עד לקביעת הזוכים

מבנה התחרות - המשך

- בכל **תור** מנוע המשחק מריץ פקודת אסמבלי אחת* מכל שורד, לאחר מכן עובר לשורד הבא וחוזר הלאה באופן מעגלי
- שורד שמנסה להריץ פקודה לא חוקית נפסל והריצה שלו מופסקת

* ניתן להריץ יותר מפקודת אחת תוך שימוש במנגנון ה"אנרגיה" שיפורט בהמשך. למען הפשטות נניח בינתיים שרק פקודה אחת בכל תור

מבנה התחרות- המשך

- **משחקון** מורכב ממספר קבוע של תורות (עשרות אלפים). רק שורדים שהגיעו לסוף המשחקון מקבלים ניקוד. אם נותר שורד יחיד, המשחקון נפסק
  - בתחילת כל משחקון מוחזרים כל השורדים לזירה, מאותחלים כל הרגיסטרים ומוגרל מיקום חדש לשורדים
- בתחרות, ציון ה**משחק** של כל שורד נקבע לפי הניקוד שהשורד השיג על פני מאות אלפי משחקונים
  - בהרצות ביתיות ניתן לקבוע את כמות המשחקונים

לתלמידי כיתות יוד

- 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 הוראה
  - התחרות איננה בין המורים ועוזרי ההוראה בבתי הספר ☺
  - תקבלו את כל העזרה בהבנת הנושאים שמכוסים במצגת זו, אך לא מעבר לכך

קודגורו אקסטרים

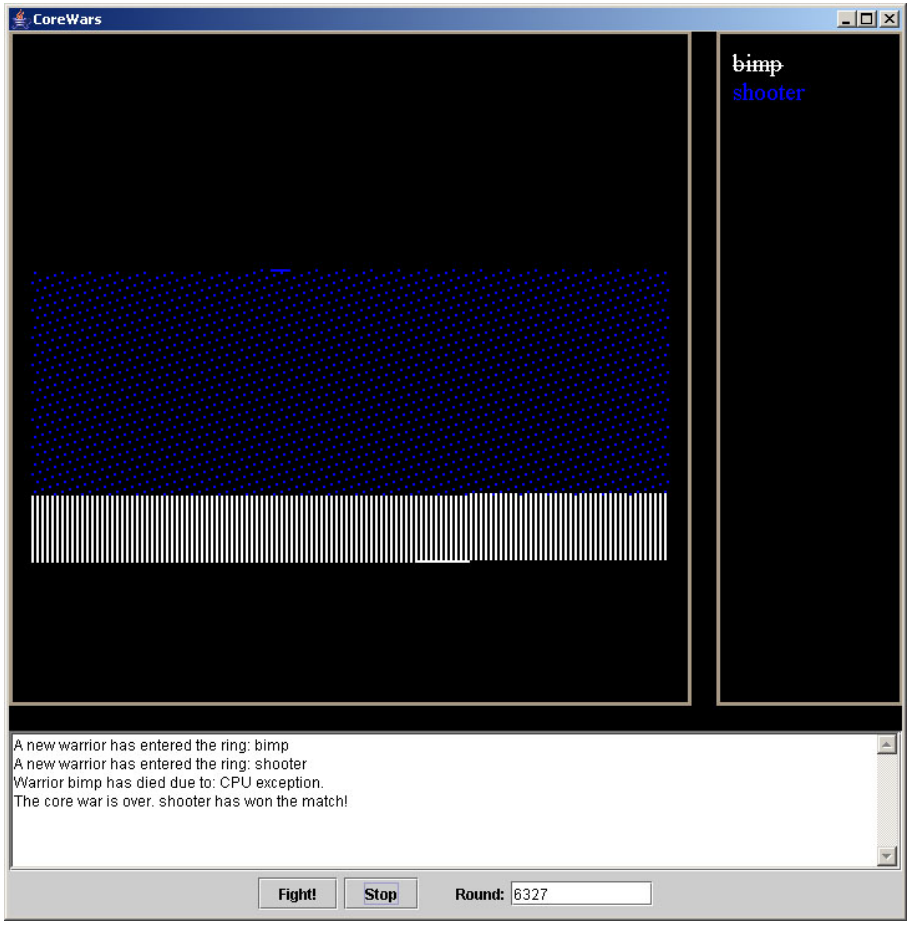
מנוע, זירה, שורד

מנוע המשחק

- **זירה וירטואלית** בה מתרחש המשחק בין מספר תוכניות
- כל תוכנית ("שורד") נכתבת בשפת אסמבלי 8086, נטענת לזירת המשחק ומורצת במקביל לשאר התוכניות
- פגיעה בשורד נעשית על ידי כתיבת פקודה לא חוקית לאזור בו נמצא אותו שורד
  - כאשר מנוע המשחק ינסה להריץ את הפקודה הלא חוקית, השורד 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 65,546 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 ייעודם לכתוב כמות גדולה של פקודות לזיכרון המשחק במספר תורות מצומצם ככל שניתן, על מנת להביס את היריבים
  - **הגנתיים**: שורדים אשר ייעודם להתחמק מפגיעה של יריבים על ידי טכניקות שונות

קודגורו אקסטרים

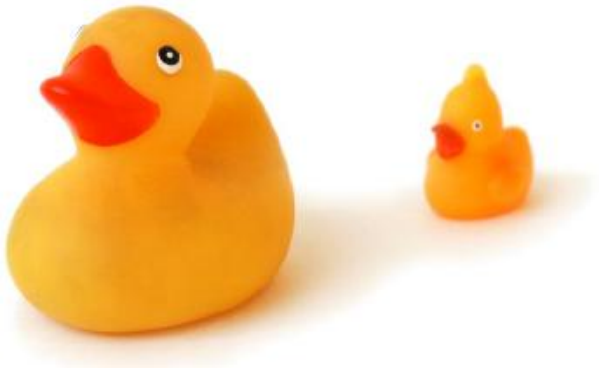
כתיבת שורד

קובץ שלד לכתיבת שורד

- נתבונן בקטע הקוד הבא. בשונה מקובץ שלד "רגיל" (base.asm) בקובץ זה:
  - מודל הזיכרון הוא tiny- מתאים לקובצי com
  - אין DATASEG
  - הגדרת org 100h, כפי שנדרש מקובץ com

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

כתיבת השורד הראשון



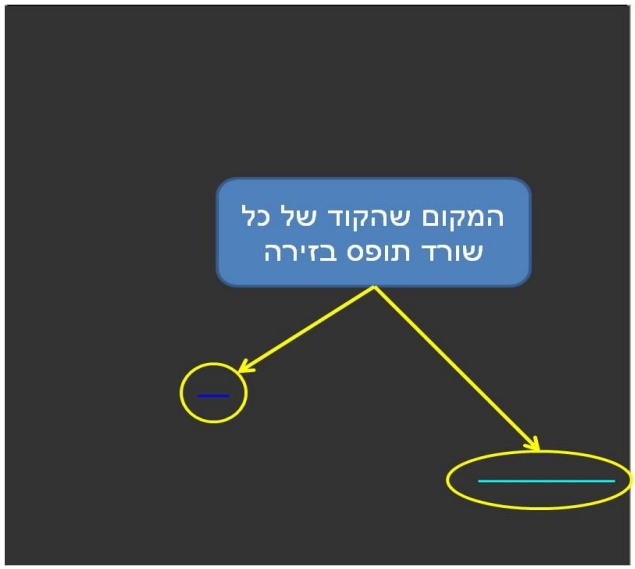
- נתבונן בקטע הקוד הבא

```
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 מיקום השורדים בזירה כ"קווים אופקיים" בזירה, אלו המקומות שהם תופסים בזיכרון
- מנוע המשחק דואג לכך ששני שורדים לא ייטענו למקום זהה בזיכרון
- מנוע המשחק מאפשר לכל שורד להיות באורך של עד 512 בתים



תפיסת מקום בזירה

- נבצע שינוי בשורד שלנו, 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
```

- start ו-exit 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 שהוא ינצח את השורד הקודם?
  - איך אפשר לשפר אותו?

```
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
```

קודגורו אקסטרים

יכולות מיוחדות של שורדים:

* הפצצה כבדה
* הפצצה חכמה
* האצה

"הפצצה כבדה"

- 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 לזירת המשחק! ניתן להשתמש פעמיים בלבד בכל סיבוב בהפצצה כבדה (עבור כל שורד)
- בדוגמה הבאה, נפציץ את האזור הקרוב ביותר אלינו (נכתוב 512 בתים במהירות) ולאחר מכן נעבור להפצצה רגילה
  - לכמה מהלכי כתיבה רגילה שקולה "הפצצה כבדה"?

הפצצה כבדה- המשך

- "הפצצה כבדה" של אזור בזיכרון: כאשר הפקודה נקראת, מנוע המשחק יכתוב 64 פעמים את dx:ax לכתובת המתחילה ב-es:di (סה"כ ייכתבו 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 ה-direction flag, וערכם של es:di לאחר הקריאה מתעדכן בהתאם

- פרמטרים:
  - ax, dx- הערך שייכתב
  - es- הסגמנט אליו נכתוב
  - di- האופסט אליו נכתוב
  - Direction flag- כיוון ההפצצה
    - CLD
    - STD
  - int 86h- פקודת ההפעלה, עד פעמיים 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
```

"פצצה חכמה"

- לעיתים נראה שישנם שורדים יריבים שמהווים איום ממשי על ניצחוננו
  - נוכל לטפל בהם בצורה "כירורגית"
- מנוע התחרות מקצה לכל שורד "**פצצה חכמה**" אחת בכל סיבוב
- פצצה חכמה: פקודה המחפשת אחר מבנה נתון של 4 אופקודים, ומחליפה אותם בקוד אחר שהגדרנו לה
- הפקודה להפעלת פצצה חכמה: int 87h (מוכרת רק למנוע המשחק, אל תנסו בסביבה אחרת ☺)
- בצורה זו, במידה וידוע לנו המבנה הבינארי של השורד היריב, נוכל לטפל בו באופן פרטני

הפצצה חכמה- המשך

- "הפצצה חכמה" של אזור בזיכרון: כאשר הפקודה נקראת, מנוע המשחק מחפש את ההופעה הראשונה 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 (קדימה או אחורה) נקבע לפי ה- direction flag, אך ערכם של es:di לא משתנה

- פרמטרים:
  - ax, dx- הערך שנחפש
  - es- הסגמנט בו נחפש
  - di- האופסט ממנו יחל החיפוש
  - cx, bx- הערך שייכתב
  - Direction flag- כיוון החיפוש
    - CLD
    - STD
  - int 87h- פקודת ההפעלה, פעם אחת בסיבוב

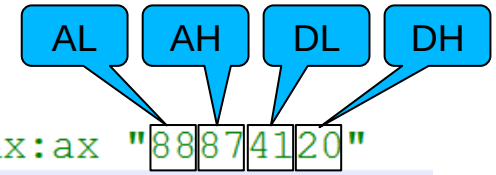
ה"קורבן" המיועד לפצצה החכמה

```
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, 2041h    ; search for dx:ax "88874120"
11     mov ax, 8788h
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
```

האצת שורד

- זוכרים שאמרנו שבכל תור מנוע המשחק מריץ פקודה אחת של כל שורד? יש דרך לקבל זמן מעבד רב יותר ☺
- לכל שורד מוגדר משתנה בשם Energy, שגודלו 16 ביט. הערך ההתחלתי שלו הוא 0
- אם הערך גדול מ-0, כל 5 תורות מנוע המשחק מחסיר מערכו 1 (אם הערך 0, נשאר 0)
- שורדים יכולים להגדיל את ערך האוגר ב-1 ע"י קריאה לאופקוד wait פעמיים בדיוק
  - מנוע המשחק מזהה שתי קריאות רצופות ל-wait ומריץ אותן בתור יחיד

```
cs:0005 9B                  wait
cs:0006 9B                  wait
```

האצת שורד- המשך

- בתחילת כל תור, אם האנרגיה שונה מאפס, "מהירות" השורד מחושבת ע"י הנוסחה

$$speed = \log2(energy)+1$$

  - הערה: אם energy=0 אז speed=0
- הסיכוי ששורד יקבל תור נוסף הוא speed/16
- לדוגמה:
  - השקענו תור בהעלאת האנרגיה ב-1. כעת speed=1. למשך 5 התורות הבאים יש סיכוי של 1/16 לקבלת תור נוסף.
  - השקענו 2 תורות והעלינו את האנרגיה ל-2. כעת speed=2. למשך 5 התורות הבאים יש סיכוי של 1/8 לקבלת תור נוסף, למשך 5 התורות שאחריהם יש סיכוי של 1/16 לקבלת תור נוסף.

קודגורו אקסטרים

זומבים והשתלטות על זומבים

זומבים

- בכל תחרות משתתפים בזירת המשחק שורדים מיוחדים הנקראים "**זומבים**"
- שורדים אלו נכתבים ע"י צוות התחרות ובעלי משימה מוגדרת
- לקראת התחרות קוד הזומבים נמסר למשתתפים
- קוד הזומבים כולל חידה תכנותית / מתימטית

דוגמה לזומבי פשוט

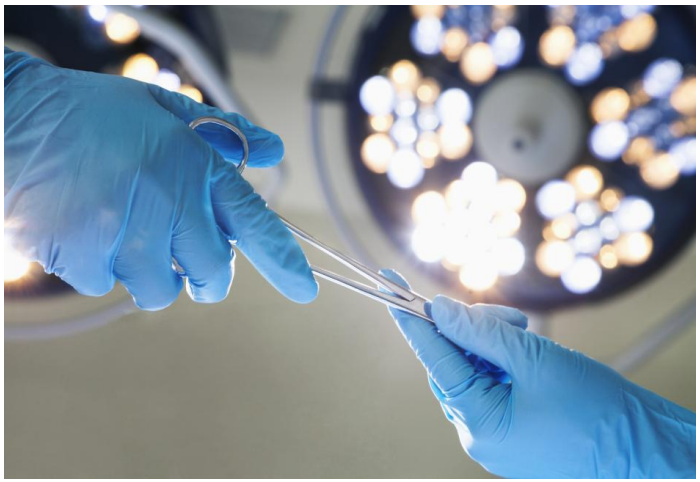
- זומבי זה כותב את כתובת ההתחלה שלו לתוך התא 1234h

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

רעיון" השתלטות על זומבי

- לעיתים מתמודדים מנסים "להשתלט" על הזומבים ע"י כתיבת פקודות בינאריות לאזורי הזיכרון של אותם זומבים. מצב זה גורם לזומבי לבצע משימות שהמתמודד בוחר
- שיטה זו מועילה למתמודד מכיוון שבעזרתה אפשר "להרוויח" את זמן הריצה של השורד ואף את סט הפצצות שלו (אם טרם השתמש בהן)
- חשוב לציין: זומבי מתפקד כשורד רגיל, מלבד העובדה שהייעוד שלו אינו לנצח במשחק אלא לאפשר יתרון לאלו שהצליחו לשבות אותו
- זומבי אינו חולק בניקוד בסוף הסיבוב

דוגמה להשתלטות על זומבי



- 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- מידע חשוב להמשך

הרעיון הכללי להשתלטות על הזומבי

- בסוף הזומבי נוסיף קוד שאומר לו לקפוץ אל השורד שלנו
- לאחר הוספת הקוד נדרוס את הלולאה שבסוף הזומבי המקורי, כך שהוא יריץ את פקודת הקפיצה ששתלנו
  - מה יקרה אם נדרוס את פקודת הקפיצה לפני שכל הקוד שלנו מוכן בסוף הזומבי?
- הקוד שאנחנו רוצים להריץ הוא מהצורה:

- mov si, xxxx ; xxxx is the value of ax
- jmp si

  - הסבר: ax הוא כתובת ההתחלה של השורד שלנו

- ע"י דיבאגר נמצא כי הקוד צריך להתחיל ב-BE, שני בתים של ax, ואחר כך FFE6 (בדיבאגר הנחנו ax=0ABCDh)

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

דוגמה לקוד שמשתלט 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
```

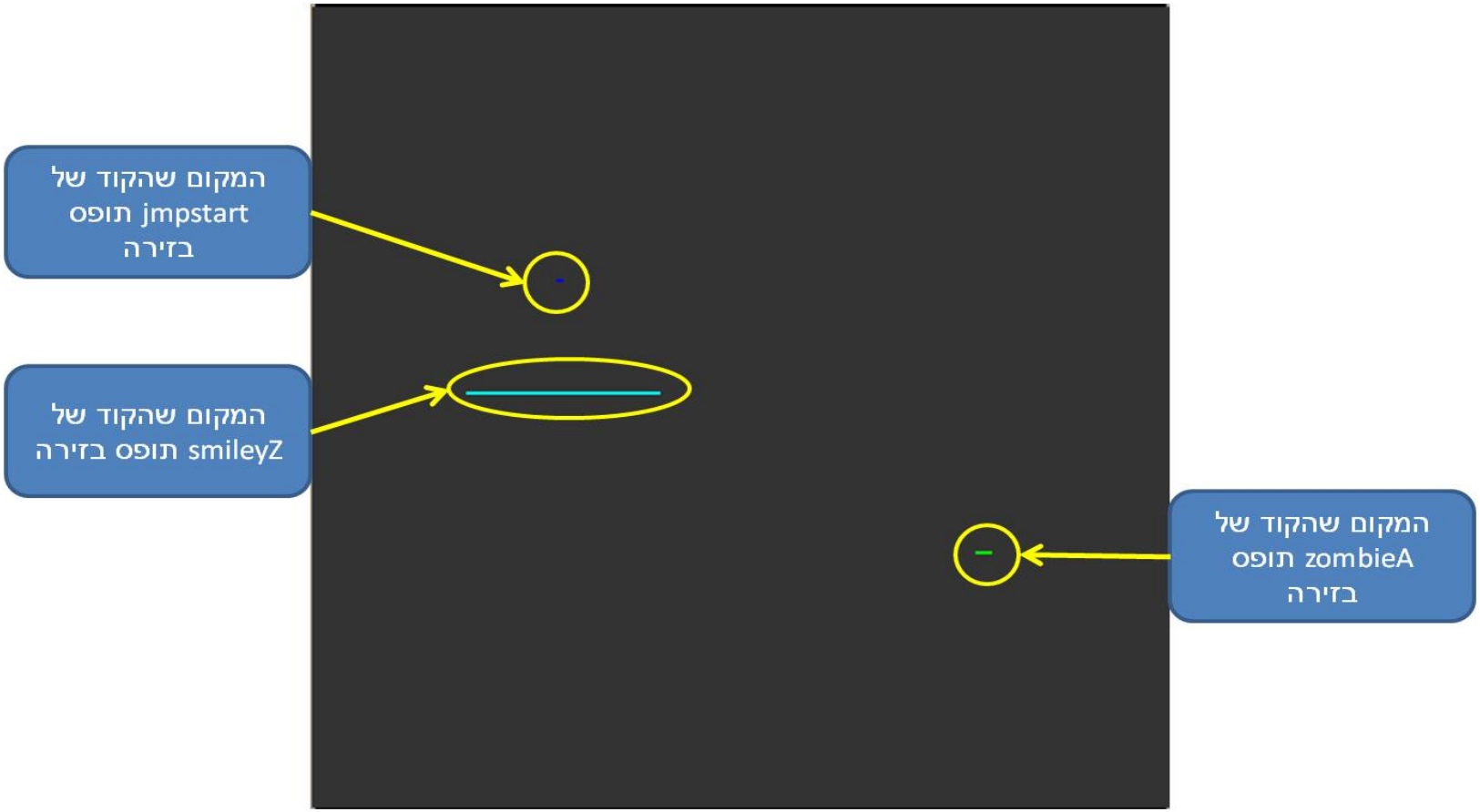
דוגמה לקוד שמשתלט על הזומבי

- נחבר את כלל הקוד להשתלטות
- יש לשים לב לשורה הראשונה: נועדה לאפשר לזומבי תור לכתוב את כתובת ההתחלה שלו ל-1234h, לפני שאנחנו משתמשים בערך שבכתובת זו כדי לאתר את המיקום שלו

```
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:
```

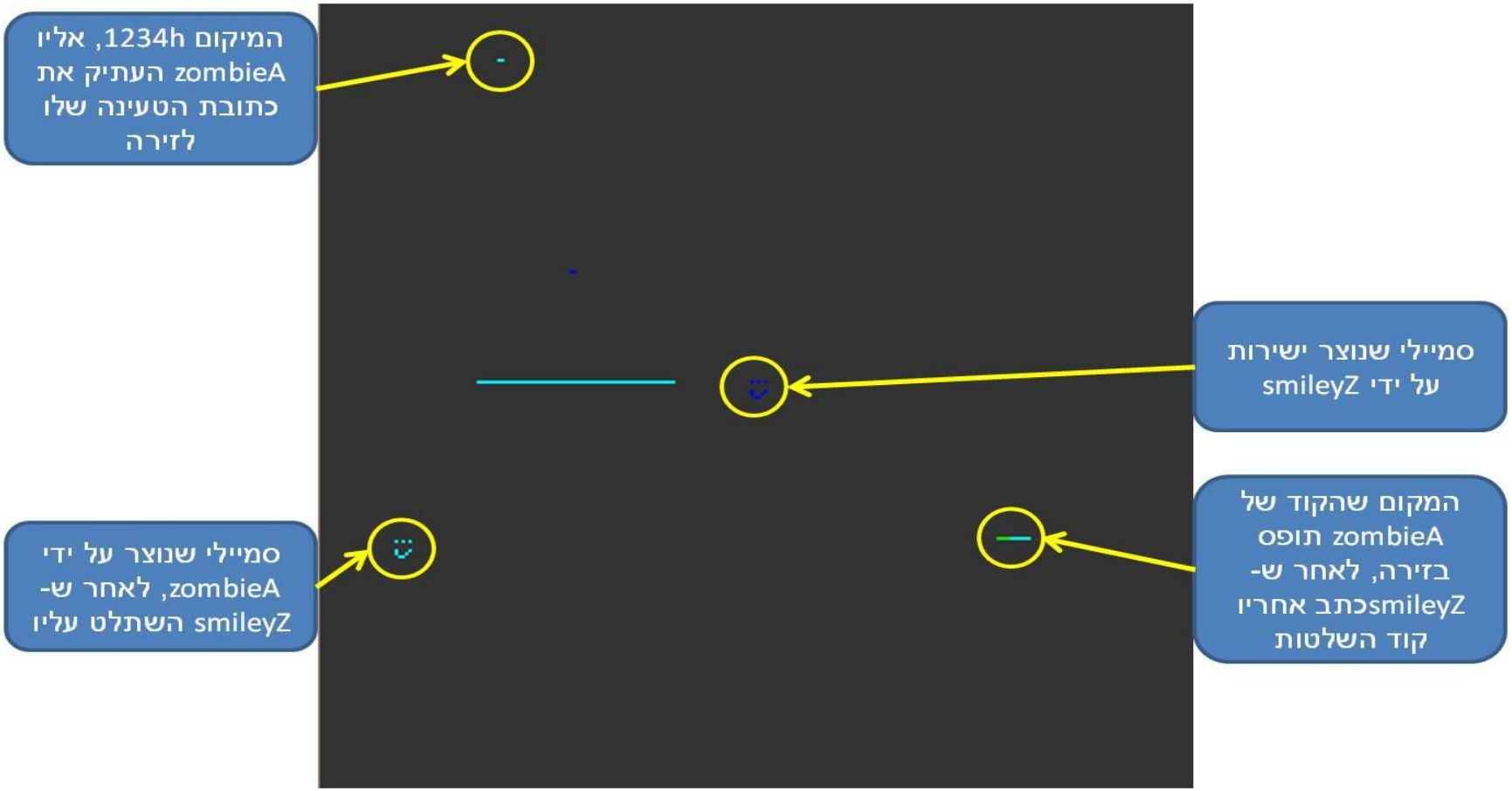
השתלטות על זומבי- המחשה בזירה

- מצב התחלתי: זומבי, שורד א' שמצייר סמיילי ומשתלט על הזומבי, 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 אקסטרים
- בשיטה זו, שורד "משכפל" את עצמו מאזור זיכרון אחד לאזור זיכרון אחר
- לשיטה זו מספר יתרונות:
  - מיקום משתנה לאורך המשחק
  - השורד עלול "לדרוס" את יריביו במהלך ההעתקה
  - יצירת עותקים "מתים" רבים בזירה- מטרות דמה לפצצות חכמות
- **חשוב להדגיש**: בכל רגע נתון ישנו רק "מופע" חי אחד של השורד

שכפול שורד- חסרונות

- פיתוח השורד מסובך ונעשה עוד יותר קשה ככל שמורכבות השורד עולה
- יחס הזיכרון שנכתב לעומת יחס הפעולות אינו אופטימלי
  - במילים אחרות: השורד תופס מקום רב בזיכרון ולכן פגיע יותר
- נדרשת הבנה עמוקה ביותר 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: מעתיקה מילה מכתובת ds:si לכתובת es:di, ערכי הרגיסטרים מתעדכנים בהתאם ל-direction flag
- הקוד הבינארי שנוצר מכיל 12 בתים לפני התווית myloop ו-12 בתים אחריה
- 6 פקודות ה-movsw יעתיקו את 12 הבתים שאחרי myloop
- מכיוון שהאוגר di מתעדכן בהתאם, כל שנותר הוא לבצע jmp למיקום החדש

שכפול שורד- הסבר

```
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
```

כיוון ההעתקה

לפני פקודת ההעתקה הראשונה, di מצביע על שורה זו

לפני הקפיצה, "נתקן" את di כך שיצביע על פקודת ה-dec הראשונה

לפני פקודת ההעתקה הראשונה, si מצביע על שורה זו

קודגורו אקסטרים

טקטיקה מתקדמת: כתיבה בקצב כפול

Far Call

- הרעיון: בשפת אסמבלי, כאשר אנחנו קוראים לפרוצדורה בעזרת פקודת call, אנו דוחפים למחסנית 2 **בתים** שמציינים את מיקומנו טרם הקריאה (האופסט מתחילת הסגמנט)
- אם ביצענו קריאת far לפרוצדורה, נדחפים למחסנית הן האופסט והן הסגמנט, סך הכל **4 בתים**
- ... במידה ונחליט שזירת המתמודדים היא המחסנית שלנו, נוכל לדרוס ארבעה בתים בכל תור במקום שניים ☺
  - מהם חסרונות השיטה הזו?

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 הספרה 1 או 2 בסוף שמם)
  - לדוגמה: קבוצה בשם rocky תקרא לשורדים בשמות rocky1 ו-rocky2
- היתרון ברור: יותר זמן משחק, יותר סיכוי לנצח!
- הניקוד מתחלק בין שני השורדים (אם רק אחד שרד- מחצית הניקוד)
- התקשורת בין השורדים נעשית בסגמנט אישי (es) בגודל 1024 בתים. סגמנט זה פרטי ואין לשורדים מקבוצות יריבות אפשרות לפנות אליו.

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 לסגמנט es ומשתמשים בה לביצוע פעולות תקשורת בין השורדים

השורד cannon1 הכניס לכתובת 0 את מיקומו ההתחלתי

בעזרת bp פונים לסגמנט המחסנית, אותו קבענו כבר להצביע ל-es, לכן di מקבל את כתובת ההתחלה של cannon1

אתגר

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

קודגורו אקסטרים

מקורות נוספים

חוברת הדרכה וספר לימוד

- המדריך למשחק, 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

הפורום

- בפורום האתר יש דוגמאות לשורדים, חלקם כולל הדרכה כיצד נבנו
- חלק מהדוגמאות במצגת זו נלקחו מדפי הפורום, נצלו את נסיון השנים הקודמות כדי ללמוד וגם כדי לשאול שאלות
- לינק: http://goo.gl/49h62H (מומלץ לגלוש מדפדפן כרום)

Facebook

- מקום מפגש של קודגור-ים בעבר ובהווה
- בדף נפרסם עדכונים שוטפים לקראת התחרות
- לינק http://facebook.com/CodeGuru.IL

בהצלחה!

גבהים
חינוך • מצוינות • סייבר

התחרות החדשה
מבית קוד גורו
אקסטרים
Code Guru
EXTREME