ארבעה מצבי כשל, שני תיקונים, ומעגל

מה קרה כששאלנו "האם המנהל נקב בשם", והטקסט היה עברי

תוכן העניינים

1. הבאג הראשון
2. התיקון שכיביתי איתו את המערכת
3. הערה על אנגלית, כי היא לא מה שחשבתי
4. מה שכן עבד, וזה דווקא חיסור
5. המדידה
6. ואז מישהו הצביע על מה שהקורפוס שלי לא בודק
7. מצב כשל 4: תו שאינו אות בתוך מילה
8. ומה קרה בפועל, בפרודקשן
9. תקלה או נתיב תקיפה
10. וכאן צריך לומר משהו לא נוח על המאמר הזה
11. למה זה לא ניתן לסגירה, בזהירות המתאימה
12. "אבל יש לנו YAP"
13. הרעיון שנראה טוב, ומה קרה לו
14. ולמה באנגלית זה כן עובד
15. עברית איננה מקרה מיוחד
16. איפה זה יושב ביחס למה שכבר מתועד
17. המעגל
18. מגבלות
19. נספח: 89 השמות שהורצו מול שלושת הלקסיקונים

אנחנו מפעילים מנוע שעושה דברים בשם מנהל עסק. פותח משימות, מעביר אחריות בין עובדים, שולח התראות. פעולה שנוגעת באדם היא רגישה, ולכן יש שער: אם המודל בחר יעד אנושי שהמנהל לא נקב בו בפקודה שלו, הפעולה לא מתבצעת. היא עולה לאישור.

הרעיון פשוט. בחירת אדם היא סמכות המנהל, לא של המודל. השער רק בודק אם המנהל אמר את השם.

המימוש הראשון היה שורה אחת:

```
text.includes(name)
```

זה החזיק בערך יומיים.

ומה שלא ידעתי אז הוא שכשאנסה לספור כמה פעמים הוא טעה, לא יהיה מה לספור: השער אינו שומר את הטקסט שהכריע עליו, ולכן שגיאותיו אינן ניתנות לשחזור. זה הממצא שהכי הפתיע אותי, והוא מחכה בהמשך.

1. הבאג הראשון

עברית מדביקה מילות יחס וחיבור לתחילת המילה. מש"ה וכל"ב: מ, ש, ה, ו, כ, ל, ב. הן מצטברות, ו"וכשדנה" היא מילה חוקית אחת. התוצאה היא ששמות קצרים נבלעים בתוך מילים יומיומיות כל הזמן.

"שי" יושב בתוך "השייכות". "אור" בתוך "האור". "גל" בתוך "בגלל".

מה שהפך את זה מתקלה לבאג הוא הכיוון. בדרך כלל כשמסנן מתאים יותר מדי, הוא חוסם יותר מדי, וזה מעצבן וגלוי ומישהו מתלונן. כאן ההתאמה עושה את ההפך: היא אומרת לשער "כן, המנהל נקב באיש הזה", והכתיבה מדלגת על כרטיס האישור ומתבצעת על מי שהמנהל מעולם לא הזכיר.

השער שנועד לעצור נפתח.

2. התיקון שכיביתי איתו את המערכת

הצעד הבא הוא מובן מאליו לכל מי שכתב פעם ביטוי רגולרי. עוברים לגבול מילה:

```
new RegExp(`\\b${name}\\b`).test(text)
```

ושורה אחת על הקוד הזה לפני שממשיכים, כי הוא שגוי בדרך נוספת: השם מוזרק לביטוי כמו שהוא. שם שמכיל תו מטא — נקודה, סוגריים, כוכבית — מרחיב את השער או מפיל אותו בחריגה. הוא חייב לעבור escaping קודם, וזה נכון גם כששמות מגיעים מרשימת עובדים ולא מקלט משתמש:

```
const esc = s => s.replace(/[.*+?^${}()|[\]\\]/g, '\\$&')
new RegExp(`\\b${esc(name)}\\b`).test(text)
```

בעברית זה מחזיר `false`. תמיד. על כל קלט. עם דגל `u`, עם דגל `v`, ובלעדיהם.

הסיבה מתועדת, וברגע שרואים אותה היא מובנת מאליה. `\b` איננו "גבול מילה" במובן הלשוני, הוא מעבר בין תו `\w` לתו שאיננו `\w`. ו-`\w` ב-JavaScript הוא `A-Z`, `a-z`, `0-9` וקו תחתון. אות עברית איננה `\w`, ולכן העמדה שצמודה לה תלויה כולה בצד השני, ובטקסט עברי הצד השני הוא רווח או פסיק. שניהם לא `\w`. אין מעבר, אין גבול, אין התאמה.

וזה כבר לא באג באותו כיוון. שער שמחזיר `false` תמיד, בתכן קוטבי כמו שלנו שאומר "לא נקוב, לכן הסלם", פירושו שכל פעולה עולה לאישור. עברנו מכשל שקט לכשל רועש.

זה השלד של המאמר: **שני כיווני כשל הפוכים, והתיקון המקובל לאחד מייצר את השני.**

3. הערה על אנגלית, כי היא לא מה שחשבתי

הייתי בטוח שהסיפור הוא "הבאג בלתי נראה באנגלית". הוא לא. התאמת תת-מחרוזת שוברת באנגלית מצוין:

שם	טקסט	`includes`	`\b`
mark	this is a **mark**et problem	✓	✗
bill	the **bill**ing page is broken	✓	✗
ann	the **ann**ouncement goes out	✓	✗
sam	send the **sam**e file	✓	✗
ron	that assumption is w**ron**g	✓	✗
ian	the dev**ian**t behaviour persists	✓	✗

מה שבלתי נראה באנגלית זה **כישלון התיקון**. מפתח דובר אנגלית רואה את `mark`/`market`, מוסיף `\b`, הבדיקות עוברות, הוא ממשיך הלאה. בעברית אותו תיקון בדיוק לא מצמצם את השער אלא מכבה אותו, ואין לזה שום סימן בבדיקות שהוא כתב.

4. מה שכן עבד, וזה דווקא חיסור

מה שהחזיק בסוף הוא טוקניזציה עם קילוף תחיליות: מפרקים את הטקסט למילים, ומתירים עד שלוש תחיליות על הטוקן הראשון של השם.

הפרט שנראה לי כנושא את רוב המשקל הוא מה שהוצא מהרשימה, לא מה שנכנס אליה. ה' הידיעה מוחרגת. בהמשך הפרק אמדוד את זה ואמשוך את הניסוח הזה בחזרה.

הנימוק הוא דקדוק, ואינו שלי. ה' הידיעה אינה נסמכת לשם פרטי, כי שם פרטי מיודע מעצם טבעו. גזניוס מנסח את זה בסעיף 125. אף אחד לא כותב "הדנה" על אדם. לעומת זאת המון שמות עצם רגילים הם בדיוק שם ועוד ה': "האור" אינו "אור", "השני" אינו "שני", "הקרן" אינה "קרן". כלומר ה' היא התחילית היחידה שנוכחותה לפני שם היא ראיה שזה לא השם.

וכאן הייתי צריך לעצור ולשאול כמה פעמים הראיה הזאת בכלל נוכחת בכתב. ה' נבלעת אחרי ב, ל, כ. כותבים "לבית" ולא "להבית", ו"לַגפן" המיודעת נכתבת בדיוק כמו "לגפן" הלא-מיודעת. כלומר בשלושת ההקשרים המיודעים הנפוצים ביותר, הראיה שההחרגה מחפשת נמחקת מצורת השטח.

מדדתי את זה על המימוש עצמו, על חמישה שמות שהם גם שמות עצם:

```
הגפן  האדר  התדם  הקרן  השרדן              0 מתוך 5    ההחרגה פועלת
לגפן  בגפן  בגפן  לאור  באדר  באדר  ...   15 מתוך 15   ההחרגה עידודרת
```

ה' נבלעת רק אחרי ב, ל, כ; אחרי מ, ש, ו היא נשארת גלויה — `מהגפן`, `שהגפן`, `והגפן` כולם נדחים נכון. מתוך עשרים רצפי-התחיליות שבנספח, שמונה שומרים על העדות ושנים עשר עיוורים לה, כלומר שישים אחוז. וכשמסתכלים על הצורות הנפוצות ביותר שמצאתי בפרודקשן, `לגפן` 27 פעמים ו-`לרינה` 28, שתיהן יושבות בדיוק בטווח העיוור. שתי העובדות האלה ישבו אצלי זו לצד זו זמן ארוך בלי שחיברתי ביניהן. וגם `בשרדן`, אחת משלוש שגיאות השווא שנשארו לי, היא ב+ה+שרון — ו"השרון" הוא בדיוק אחד החריגים הגיאוגרפיים שאני מונה בפסקה הבאה.

זה לא הורס את השיטה, והיא עדיין הטובה מבין אלה שבדקתי. אבל "הפרט שנשא את רוב המשקל" הוא ניסוח שאני מושך בחזרה: ההחרגה מסלקת קבוצה אחת של שגיאות שווא, והיא איננה הקבוצה הגדולה. הרצפה שתיארתי גבוהה יותר משחשבתי.

שלוש הסתייגויות שאני מעדיף לומר בעצמי. הכלל תקף לשמות בני אדם; שמות גיאוגרפיים דווקא נוטים עם ה' — הירדן, הגלבוע, השרון — וגזניוס מונה בדיוק את החריגה הזאת; הדוגמאות שהוא עצמו מביא הן הגבעה, הלבנון, היאור והירדן. בדיבור לא פורמלי כן אומרים "הדנה הזאת מהחשבונות", וזה בדיוק הרג'יסטר שמנהל מקליד בו. ותקרת שלוש התחיליות מפספסת ארבע, כך ש"ולכשדנה" נדחית. בשלושתם הכשל מוסיף כרטיס אישור מיותר ולא מוציא פעולה החוצה, וזה הכיוון שאני מוכן לשלם בו.

5. המדידה

קורפוס א׳, שמונה משפטים שבהם השם נבלע בתוך מילה אחרת ואינו מתייחס לאדם:

```
שי    העברתי את המשימה בגלל השייכות לפרויקט
אור   צריך לבדוק את האור במשרד
שני   הלקוח השני בתור מחכה
קרן   הקרן החדשה אושרה אתמול
גל    בגלל העיכוב לא הספקנו
רון   יש בעיה בשרון הצפוני היום
תום   בתום היום נסגור את הדוח
אור   לאור הדברים נמשיך ברגיל
```

ושמונה אזכורים אמיתיים. שלושה נושאים תחילית על השם, חמישה חשופים:

```
שי    תעביר לשי את הקובץ
דנה   ולדנה תשלח תזכורת
גל    תן לגל את המשימה הזו
אור   תשאל את אור מה קרה
שי    שי סיים את הבדיקה
אור   אור צריכה לסיים את המשימה
דנה   דנה מטפלת בזה
גל    גל ואני נסגור את זה
```

שבע השיטות בטבלה הן `includes()`, ארבע גרסאות של גבול, קילוף התחיליות שתיארתי זה עתה, והיברידי. "היברידי" הוא ביטוי אחד שמחבר בין השניים האחרונים: גבול `\p{L}` משני הצדדים, ובתוכו עד שלוש תחיליות לפני השם.

```
new RegExp(`(?<!\\p{L})[ושכבלמ]{0,3}${esc(name)}(?!\\p{L})`, 'u')
```

שיטה	שגיאות שווא	זיהויים אמיתיים
`includes()`	8 מתוך 8	8 מתוך 8
`\b`	0	0
`\b` עם `u`	0	0
`\b` עם `v`	0	0
גבול `\p{L}`	0	5 מתוך 8
קילוף תחיליות	3	8 מתוך 8
היברידי	3	8 מתוך 8

שלוש שגיאות השווא הן `לאדר`←אור, `בתדם`←תום, `בשרדן`←רון. בניתי שיטה שלישית כדי לנסות לסגור אותן וקיבלתי בדיוק את אותן שלוש, וסוקר שבנה קורפוס משלו הגיע גם הוא לאותן שלוש.

בכל אחת מהן, התחילית והשם יחד מאייתים מילה עברית קיימת. וכל מה שלא ירה הוא המקרה ההפוך: "גל" אינו מתאים ל"בגלל", כי הקילוף נעצר ב"גלל" ולא מגיע ל"גל".

שווה לומר כאן שאני לא הראשון שנתקל בזה. איתמר סין-הרשקו תיאר את בעיית אותיות השימוש כבר ב-2010, בעבודתו על HebMorph, והראה איך התאמה נאיבית נשברת עליהן. ההבדל הוא שאצלו זו הייתה בעיית אחזור מידע, כלומר תוצאת חיפוש חסרה או מיותרת, ואצלי היא החלטה אם אדם נשאר בלולאה.

בשלב הזה חשבתי שסיימתי.

6. ואז מישהו הצביע על מה שהקורפוס שלי לא בודק

כל שמונת המשפטים בקורפוס א׳ נושאים תחילית. שמונה מתוך שמונה. בניתי קורפוס שבודק מצב כשל אחד בלבד, ולא שמתי לב עד שסוקר אמר לי.

והמצב השני הוא הרוב, כי קונבנציית השמות הישראלית בונה שמות מתוך מילים. אור, שי, גל, תום, קרן, שחר, רון, שני, יעל, תמר, ורד, שקד, אלון, ארז, טל, ברק, אביב, עדן, נועם, שירה, ים, דרור, חן. אלה שמות, ואלה מילים, ואין שום דבר שמפריד ביניהם.

קורפוס ב׳, תשעה משפטים שבהם השם מופיע כטוקן חשוף ואינו מתייחס לאדם:

```
אור   קיבלנו אור ירוק מהלקוח
שי    הוא הביא שי לכבוד החג
גל    גל של תלונות הגיע הרגע
שחר   עם שחר נצא לדרך
יעל   יעל טיפסה על הסלע
קרן   קרן השמש חדרה פנימה
טל    טל של בוקר על העשב
ברק   ברק האיר את השמיים
ים    ים של עבודה מחכה לנו
```

שיטה	שגיאות שווא מתוך 9
`includes()`	9
גבול `\p{L}`	9
קילוף תחיליות	9
היברידי	9
`\b`	0

קילוף התחיליות נכשל בכל אחת מהשורות. וכך גם ההיברידי, וכך גם `\p{L}`. השיטה היחידה שעובדת היא `\b`, ורק מפני שהיא לא מתאימה שום דבר לעולם, וזו לא הצלחה.

ההבדל בין שני המצבים הוא מהותי ולא כמותי. תחילית מוסיפה תווים, ולכן אפשר לקלף אותה. הומונים חשוף לא מוסיף כלום. אין מה לקלף, אין גבול לתקן, אין שום מניפולציה על המחרוזת שתיגע בו, כי הוא זהה לשם, תו בתו.

7. מצב כשל 4: תו שאינו אות בתוך מילה

את זה לא מצאתי בעצמי. סוקר הריץ את הקוד ושאל מה קורה כשבתוך מילה יושב תו שאינו אות — ניקוד, מקף, או תו בלתי-נראה. התשובה מפילה את השיטה שבחרתי:

```
שי    "שִׁירֵי המלאי"        קילוף תחיליות: התאמה   (ניקוד)
אור   "אור<ZWSP>ות המשרד"  קילוף תחיליות: התאמה
אור   "מערכת אור-קולית"    קילוף תחיליות: התאמה   (מקף)
תום   "תום<RLM>ה נשלחה"    קילוף תחיליות: התאמה
```

שש מתוך שש שבדקתי, גם בקילוף התחיליות, גם ב-`\p{L}` וגם בהיברידי. **והמנגנון בקילוף גרוע יותר, לא זהה:** הטוקנייזר מפצל על `[^\p{L}]+`, ולכן תו בלתי-נראה בתוך מילה **יוצר טוקן חדש ששווה בדיוק לשם**. זו לא התאמה חלקית, זו התאמה מושלמת על טוקן שלא היה שם.

וזו אותה צורה בפעם השלישית. `\b` מת כי `\w` הוא ASCII. בודק-האיות מת כי אין רישיות. ו-`\p{L}` מת כי "מילה היא רצף אותיות" — הנחה שנכונה לכתב הלטיני ולא לכתב שמסמן תנועות בסימנים נפרדים. שלוש שכבות, אותה הנחה.

ההשלכה החמורה היא לא הכיוון אלא ההגנה. בפרק התקיפה אני נשען על כך שבוני-האירועים לא מכניסים טקסט של צד שלישי. זה לא מגן כאן בכלל, כי המנהל עצמו מדביק טקסט ממייל, מוואטסאפ ומטפסים, וטקסט מודבק נושא ZWSP, מקפים רכים, סימני כיווניות וניקוד כדבר שבשגרה. ולהבדיל מ"לאור הדברים", שדורש משפט מתקבל על הדעת, תו בלתי-נראה ליד שם עובד אינו דורש כלום ואינו נראה בטקסט המרונדר.

וכמה עוד כאלה יש. חיפשתי בעצמי מצב חמישי, ומצאתי חמישה:

```
צורת תצוגה (U+FB21)     אור      אין התאמה
גרש עברי (U+05F3)       ג'ון     אין התאמה
גרשיים בתוך טוקן       ש"י      אין התאמה
ר      אין התאמהסלטינית מוחלפת  א o
ניקוד ברשומת העובד     אוֹר     אין התאמה
```

כולם נכשלים לכיוון הסגור — אין התאמה, ולכן הסלמה. מצב 4 הוא היוצא מן הכלל, והוא היחיד שנכשל פתוח.

וזה נותן קריטריון עצירה במקום רדיפה: **מצב כשל נכנס לדיון האבטחתי אם ורק אם הוא נכשל פתוח.** המחלקה הזאת קטנה וניתנת לאפיון, כי היא דורשת שהמניפולציה **תיצור** התאמה — קשה בהרבה מלשבור אחת. השאר, זה שנכשל סגור, אינו פרצה אלא עומס:

אם תוקף יכול להפעיל אותו לפי רצון הוא מייצר הסלמה המונית לפי דרישה, וזה בדיוק T1621 שאני מצטט בפרק המעגל.

אני לא טוען שהרשימה סגורה. אני טוען שהזנב הזה הוא לא פער במאמר אלא הטענה שלו: לפרדיקט על מחרוזת בשפה עשירת-מורפולוגיה אין רשימת כשלים שנסגרת, יש התפלגות שממיינים לפי כיוון. וזו בדיוק הסיבה שהתשובה היא עוגן מבני ולא ביטוי רגולרי טוב יותר.

8. ומה קרה בפועל, בפרודקשן

משפט אחד לפני המספרים, על מה שהם באמת. מה שאני קורא לו כאן פרודקשן הוא מסד-הנתונים החי שלנו, אבל כמעט כולו העסק שלנו עצמו: מתוך 766 הטקסטים, 764 הם משם ורק שניים מלקוח. ורשימת העובדים שבו נזרעה בסקריפט seed ולא נוצרה מהתפלגות שמות טבעית.

עד כאן כל מספר במאמר בא מקורפוס שבניתי ביד. יש לנו לוגים, אז ניסיתי למדוד את שיעור שגיאות השווא בפרודקשן.

לא הצלחתי, והסיבה שווה יותר מהמספר שחיפשתי.

חיפשתי את טקסט-הפקודה שהשער בודק מולו. הוא לא נשמר. אין לו עמודה, אין צעד-ריצה שמחזיק אותו, ואף רשומה לא מכילה אותו. המקום היחיד שבו הוא שורד הוא הרציונל של כרטיס-אישור.

וכאן טמונה הבעיה. **כרטיס אישור נוצר רק כשהשער הסלים.** ושגיאת שווא, מעצם הגדרתה, היא המקרה שבו השער אמר "נקוב", לא הסלים, והכתיבה בוצעה. כלומר במקום היחיד שבו טקסט-הפקודה נשמר, שגיאת שווא אינה יכולה להופיע.

בשער הזה, שגיאות שווא אינן ניתנות לשחזור בלי טקסט-הקלט. לא נדירות, לא אפסיות — פשוט אין ממה לשחזר אותן. הכתיבה עצמה כן נרשמת, ולכן אפשר להצליב כתיבות-לאדם מול כרטיסי-אישור ולסקור ידנית את מה שנכתב בלי כרטיס; זה לא עובד בקנה מידה, אבל זה לא אפס. וזה כשל תצפית שלא ידעתי עליו עד שניסיתי למדוד.

מה שכן יכולתי לבדוק הוא הרשימה עצמה, וגם היא הפתיעה אותי. סרקתי 766 טקסטים בחלון שבין 3.7 ל-17.8 — 279 רציונלים של כרטיסים ו-487 סיכומי-ריצה — מול 24 השמות הפרטיים של העובדים הפעילים, כשההשוואה נעשית בתוך כל עסק לעצמו. וגיליתי ש**שלושה מהשמות ברשימה שלנו הם מילים עבריות רגילות**. אני מחליף אותם כאן בשלושה שמות בדויים שנושאים בדיוק את אותה תכונה, והספירות עצמן אמיתיות: רינה היא שירת שמחה, עופר הוא איל צעיר, וגפן היא הצמח.

ומכיוון שהרשימה הזאת נזרעה בסקריפט, "שלושה מתוך 24" הוא כאן אמירה על מי שכתב את ה-seed ולא על האופן שבו ישראלים קוראים לילדיהם. אני מביא אותו כדגימה של התופעה, לא כשיעור.

והצורות המתנגשות מופיעות, שוב ושוב:

```
לגפן   27  ·  לרינה   28  ·  מגפן   4
דרינה  4   ·  שגפן    4   ·  שלרינה  2  ·  דעופר  1
```

קראתי אותן והן נראות כמו אזכורים אמיתיים של אנשים. אבל שימו לב מה עשיתי כדי לדעת את זה: **קראתי את המשפט.** הכרעתי בהקשר, כלומר בדיוק הדבר שאני טוען לאורך המאמר ששער ברמת-הטוקן אינו מסוגל לעשות. הביקורת הידנית שלי הצליחה במקום שהשער היה עיוור, וזו הדגמה חיה של הפער בין שתי מחלקות הפתרון, לא ראיה לניקיון.

לצד ההתנגשויות דרך תחילית יושבת ההתנגשות החשופה, וזו הגדולה מבין השתיים: 451 התאמות לשם חשוף. מתוכן, 157 ההתאמות על שלושת שמות-ההומונים נבדקו ידנית; 294 הנותרות, על שמות שאינם מילים עבריות, לא נבדקו — כי שם התנגשות חשופה אינה אפשרית מלכתחילה. עברתי על 143 חלונות-ההקשר הייחודיים של ה-157, וכולם אזכורי אדם — אף לא אחד היה שם-העצם. אלא ש-441 מתוך 451 יושבות בסיכומי-ריצה, כלומר בטקסט שהמנוע עצמו כתב על העובדים שלו, ובטקסט כזה אזכור-אדם מובטח כמעט מראש. זו אינה הפרכה של ההתנגשות החשופה, זו מדידה שמסגרת-הדגימה שלה כמעט מבטיחה את תוצאתה.

אז למה זה לא התפוצץ. **לא בגלל רשימת השמות, שהיא כן מתנגשת.** אלא בגלל אוצר המילים של התחום. קורפוס של ניהול משימות לא מייצר משפטים על שירת שמחה, על איילים או על גפנים. `לרינה` בהקשר שלנו הוא כמעט תמיד אדם, כי שום דבר אחר לא מדובר.

וזו מסקנה מטרידה יותר מ"השמות שלנו במקרה בסדר". **החשיפה היא מכפלה של רשימת השמות באוצר המילים של התחום, והיא בלתי נראית עד שאחד מהשניים משתנה.** אותו שער, אותה רשימת עובדים, בחברת הפקות או במשק חקלאי או בעסק אינסטלציה, ומתחילות התנגשויות. בלי ששורת קוד אחת השתנתה, ובלי שאיש ידע שמשהו זז.

זה המעגל שלנו בגרסה שלישית: הרצפה קיימת תמיד, ומה שמסתיר אותה הוא צירוף מקרים לקסיקלי שאף אחד לא בחר, אף אחד לא מנטר, ואי אפשר לראות בלוגים.

ומכאן התיקון המעשי, והוא עמודה אחת. לשמור את הקלט שהשער הכריע עליו — או, אם הטקסט המלא כבד מדי, את המחרוזת שהותאמה ואת מיקומה בתוכו — בכל הכרעה, ולא רק כשהשער הסלים. זה כל ההבדל בין שגיאות שווא שאי אפשר לספור לבין שגיאות שווא שייספרו מחר.

9. תקלה או נתיב תקיפה

זו השאלה הראשונה שקורא אבטחה ישאל, והיא קובעת אם קראתם באג נכונות עם טעם אבטחתי או עקיפת הרשאה. התשובה תלויה בדבר אחד: **מי כותב את הטקסט שהשער בודק מולו.**

אצלנו הטקסט הזה הוא פקודת המנהל, ולצידה הקשר-האירוע כשהריצה אוטונומית. שני דברים מגבילים את החשיפה. ראשית, הקשר-האירוע עובר סניטייזר, כלומר הוא מסומן כקלט לא-מהימן עוד לפני הדיון הזה. ושנית, בוני-האירועים אצלנו מנוסחים כך שלא יכניסו לתוכו טקסט שלקוח כתב: אירוע כשל-חיוב נראה כמו "חיוב על סך 49.00₪ נכשל. סיבה: כרטיס חסום", בלי שם הלקוח, ויש בדיקות שנופלות אם מישהו יוסיף אותו.

אז אצלנו, היום, שלוש שגיאות השווא של מצב 1 הן תקלה. לא הדגמתי ניצול, ואינני טוען לו.

למצב 4 זה לא חל. שם ההגנה אינה תלויה בבוני-האירועים בכלל, כי המנהל מדביק טקסט מבחוץ, וטקסט מודבק נושא תווים בלתי-נראים דרך קבע. את זה לא מדדתי, ואיני יודע כמה מהפקודות אצלנו הודבקו.

אבל שימו לב מה מגן עלינו, ומה נאבד בדרך. הסניטייזר מסמן את הקשר-האירוע כקלט לא-מהימן, וזה נכון. אלא שהשער מקבל את שני המקורות **משוטחים למחרוזת אחת**, פקודת המנהל והקשר-האירוע מחוברים בשורה חדשה, ובנקודה הזאת הסימון כבר לא קיים. השער אינו יודע איזה חלק מהטקסט מהימן ואיזה לא; הוא מחפש שם במחרוזת.

כלומר מה שמגן הוא לא השער אלא העובדה שבוני-האירועים בחרו לא להכניס טקסט חופשי של צד שלישי. זו החלטת תוכן, והיא יכולה להשתנות בשורה אחת, בלי שאיש ייגע בשער.

ומכאן הכלל הכללי, שנכון לכל מערכת ולא רק לשלנו: **ברגע ששער-ההרשאה בודק טקסט שהמערכת לא חיברה** — תיאור משימה שעובד כתב, גוף מייל נכנס, הודעה מלקוח, שדה חופשי בטופס — ההתנגשות מפסיקה להיות מקרית. מי שיודע שיש בארגון עובדת בשם אור יכול לכתוב "לאור הדברים" בשדה חופשי, ולגרום לכתיבה שהייתה אמורה לעלות לאישור להתבצע בלעדיו.

זה לא דורש ידע מיוחד. זה דורש לדעת שם אחד של עובד, וזה מידע שכל לקוח נחשף אליו.

וצריך לקרוא לתנאי המקדים בשמו. טקסט של צד שלישי שמגיע להקשר של סוכן הוא הזרקת פרומפט, וזו פגיעות גדולה יותר מכל מה שתיארתי כאן. השער אינו ההגנה מפניה. הוא מיטיגציה במורד הזרם, שמצמצמת את מה שהזרקה מוצלחת יכולה לגרום. מי שהגיע למצב שטקסט חיצוני נכנס לפקודה שהסוכן פועל לפיה, הבעיה שלו גדולה משאלת התאמת מחרוזות, והמאמר הזה עוסק בשכבה השנייה בלבד.

ורדיוס הנזק שווה שורה, כי בלעדיו אי אפשר לתעדף. כתיבה אחת שחמקה מהשער אצלנו היא משימה שנפתחה על אדם, אחריות שהועברה, או התראה שנשלחה. היא נרשמת ביומן, היא הפיכה, היא אינה נוגעת בכסף ואינה מוציאה נתונים החוצה. זה אינו אסון; זו החלטה על אדם שנעשתה בלי האדם שאמור היה לאשר אותה. בפריסה אחרת של אותו שער, רדיוס הנזק ייקבע לפי מה שהכלים שם מסוגלים לעשות, ולא לפי מה שהשער עצמו עושה.

10. וכאן צריך לומר משהו לא נוח על המאמר הזה

המצב הזה אינו עברי.

הרצתי אותו על אנגלית ו-`\b` נורה על כל שתים עשרה השורות שבדקתי: Rose ב-"revenue rose", Mark ב-"a mark on the wall", Bill ב-"send me the bill", Grace, Dawn, Will, Art, Hope, Robin, Frank, Summer, Sky. כולם שמות אנגליים נפוצים שהם גם מילים.

ושימו לב מה זה עושה לטבלה שלמעלה. שם `mark` בתוך `market` נותן `\b` שלילי, וזה נכון. אבל `mark` ב-"he left a mark on the wall" נותן `\b` חיובי. אותו שם, אותה שיטה, תוצאה הפוכה, כי הראשון הוא הבאג הראשון והשני הוא הבאג השלישי.

אז `\b` מתקן מצב כשל אחד, בכל שפה. הוא לא מתקן שום דבר אחר, בשום שפה.

זה מסדר את התמונה מחדש, ושווה לדייק בה כי שלושת המצבים תלויים בדברים שונים. מצב 1 תלוי **מורפולוגיה**: עברית מדביקה תחיליות, וכך גם תעתיק לטיני שלה, וכך גם טורקית עם הסיומות שלה. מצב 2 תלוי **כתב ומנוע**: הוא קיים רק כשהאותיות אינן `\w` במנוע שבו אתה עובד. מצב 3 אינו תלוי בכלום, הוא פשוט קיים. ומצב 4 תלוי **קידוד**: הוא קיים בכל כתב שמסמן תנועות או כיווניות בתווים נפרדים.

ארבעה מצבים, אבל שני כיוונים: 1, 3 ו-4 מרחיבים את ההתאמה, 2 מצמצם אותה. זה השלד שהבטחתי בהתחלה, עכשיו עם המצב השלישי בתוכו.

ומה שמייחד את העברית הוא לא שהיא ממציאה את השלישי, אלא ששלושת המצבים יורים בה בבת אחת. באנגלית מצב 1 קיים ומתוקן בקלות, ומצב 2 לא קיים כלל, ולכן הבעיה שם קטנה בהרבה. אבל עברית איננה יחידה בזה — ערבית וטורקית סובלות משלושתם גם הן, וזה נושא אחד הפרקים הבאים.

כמה שכיח מצב 3 בעברית? את זה כן מדדתי, ובהמשך המאמר יש המספר: מתוך 89 שמות פרטיים ישראליים שכתבתי מראש, **40 הם גם שמות עצם, כמעט מחצית**. אבל הטענה המעניינת יותר היא ההשוואתית, שבעברית זה שכיח *יותר* מאשר באנגלית.

אני לא יכול להוכיח את זה. אין לי מדידה אנגלית מקבילה, על מדגם דומה ובאותה שיטה, ובלי אחת כזאת אין למה להשוות. הפער, אם קיים, קטן ממה שהאינטואיציה שלי אמרה, ולכן ההשוואה נשארת תצפית ולא טענה.

11. למה זה לא ניתן לסגירה, בזהירות המתאימה

"לאור" הצירוף ו"לאור" האדם הם אותה סדרת תווים. "אור" השם ו"אור" הזוהר הם אותה סדרה בלי אפילו תחילית מפרידה. לכן כל מתאם שמכריע לפי הטוקן לבדו מוכרח לתת להם אותה תשובה, וזו לא תצפית אלא תוצאה של כך שפונקציה דטרמיניסטית מחזירה אותו ערך לאותו קלט.

וזה לא אומר שאי אפשר להבחין. ההקשר מבחין היטב:

```
קיבלנו אדר ירוק מהלקוח        אחריו שם תואר
תשאל את אדר מה קרה            אחריו פועל, לפניו "את"
```

מי שקורא את המשפט השלם, או מחזיק לקסיקון, או מריץ תיוג חלקי-דיבר, יבחין. זה בדיוק מה ש-YAP קיים בשבילו: מנתח המורפולוגיה והתחביר של מעבדת ONLP, שצמח מעבודתם של אמיר מור וראוט צרפתי. הוא עושה ניתוח מורפולוגי מבוסס-לקסיקון ואחריו דיסאמביגואציה משולבת של מורפולוגיה ותחביר, כלומר בדיוק ההכרעה שמתאם ברמת-הטוקן אינו יכול לעשות.

הטענה המדויקת היא שלמתאם ברמת-הטוקן יש רצפת שגיאה שלא ניתן להוריד, ושיציאה ממנה מחייבת מעבר למחלקת פתרון אחרת, שמשתמשת בהקשר או במילון. וזה בדיוק העניין, כי שער הרשאה נבנה כמעט תמיד כמתאם ברמת-הטוקן.

12. "אבל יש לנו YAP"

ההתנגדות הוגנת, ודיסאמביגואציה מורפולוגית באמת פותרת את "אור" מול "אור". וצריך לומר שהעבודה הזו טובה: צרפתי ועמיתיה כתבו במפורש על מה שגוי ב-NLP העברי וכיצד לתקן אותו, וכלי כמו YAP ותוסף העברית של Elasticsearch קיימים בדיוק כי עברית לא נכנעת לטוקניזציה נאיבית. אני לא מגלה את זה כאן, אני נתקל בזה מכיוון אחר.

ההתנגדות המתבקשת שלי הייתה שהיא "הסתברותית" — שמקבלים ממנה ציון ביטחון וסף, ושהסף הזה הוא פרמטר אבטחה שאיש לא מכייל. הלכתי לבדוק לפני שכתבתי את זה, וזה לא נכון. פורמט הפלט המתועד של YAP הוא עשר עמודות CoNLL — ID, FORM, LEMMA, CPOSTAG, POSTAG, FEATS, HEAD, DEPREL, PHEAD, PDEPREL — ואין באף אחת מהן ציון, ואין בתיעוד סף. אין שם מה לכייל, כי אין סף ואין ציון חשופים. כמעט תקפתי מנגנון שאינו קיים.

ההבדל האמיתי אינו דטרמיניזם אלא סוג הידיעה שיש לך על הכשל. בכלל אפשר למנות את קבוצת הכשל מראש, וזה מה שעשיתי: שלוש שגיאות השווא בטבלה למעלה הן הרשימה כולה עבור מצב 1, והן ניתנות לקריאה ולביקורת. אבל זה נכון רק שם: מחלקת מצב 4, של תו שאינו אות בתוך מילה, איננה בת-מנייה — כל תו בלתי-נראה, בכל מיקום, בכל שם. כלומר גם ההבטחה של הכלל, שאפשר למנות את הכשלים שלו, מחזיקה רק חלקית. במודל נלמד יש שיעור שגיאה אמפירי שצריך למדוד, הוא אינו אפס, ואינך יודע מראש אילו משפטים נופלים בו.

וזה עדיין הכיוון הנכון יותר, בתנאי שמודדים. הבעיה לא נעלמת; היא עוברת ממקום שאפשר למנות למקום שצריך למדוד.

13. הרעיון שנראה טוב, ומה קרה לו

הרצפה ברמת-הטוקן ניתנת למנייה מראש, וזה נשמע כמו מוצא אלגנטי.

לכל שם ברשימת העובדים, ולכל צירוף של אפס עד שלוש תחיליות, שואלים שאלה אחת: האם התוצאה היא מילה קיימת. זה חישוב סופי, זעיר, ורץ פעם אחת אופליין. ואז מנתבים: שם בטוח עובר במתאם הזול, שם מסומן מוסלם תמיד לאישור, בלי לנסות להכריע.

והשאלה "האם זו מילה קיימת" נראית כמו שאלת חברות במילון, כלומר בדיוק מה שבודק-איות עושה. אז הרצתי את זה על שמונים ותשעה שמות פרטיים ישראליים שכתבתי לפני שראיתי תוצאה כלשהי, ומול שני לקסיקונים בלתי תלויים: הראשון הוא hspell עצמו, של נדב הראל ודן קניגסברג, והשני הוא הלקסיקון העברי של Word (MSSP7HE.LEX). הרשימה מתפרסמת עם המאמר.

אצל Word סומנו 89 מתוך 89. אצל hspell 88 מתוך 89. לא רוב הרשימה, אלא כולה חוץ משם אחד: "מיטל", שאיננו במילון של hspell כלל.

ארבע שורות מהרשימה, עם כל צורה ש-Word קיבל מתוך עשרים רצפי-התחיליות החוקיים המנויים בנספח:

```
אור      מסומן  20/20   אור, באור, באור, לאור, מאור, שאור, שבאור, ...
דנה      מסומן  20/20   דנה, בדנה, בדנה, לדנה, מדנה, שדנה, שבדנה, ...
אברהם    מסומן  20/20   אברהם, באברהם, באברהם, לאברהם, מאברהם, שאברהם, ...
דניאל    מסומן  20/20   דניאל, בדניאל, בדניאל, לדניאל, מדניאל, שדניאל, ...
```

הסתכלו על שתי השורות האחרונות. "אברהם" ו"דניאל" אינם שמות עצם, הם לא אמורים להתנגש בשום דבר, והם סומנו בדיוק כמו "אור".

הסיבה איננה שהשמות רעים אלא שבחרתי את הכלי הלא נכון. **מילון של בודק-איות מכיל שמות פרטיים בכוונה**, אחרת הוא היה מסמן בקו אדום כל שם בכל מסמך. בדקתי בשני הלקסיקונים: שישה עשר שמות מקראיים שאינם מילים נמצאו בשניהם, שישה עשר

מתוך שישה עשר. ושניהם אינם מקבלים הכול, כי מתוך שמונה מחרוזות שהמצאתי בצורת שם, Word דחה שבע ו-hspell דחה את כל השמונה. כלומר הכלי מבחין היטב בין מילה למחרוזת שרירותית; הוא פשוט לא מבחין בין מילה לשם.

ובמקביל הוא מקלף תחיליות בכוונה, כי "ולאברהם" היא עברית תקינה. לכן הוא מקבל את "לאברהם" ודוחה את "לרגליב". שתי ההתנהגויות נכונות עבורו, ושתיהן יחד הורסות את המנייה: כל שם אמיתי נמצא במילון, וכל צורה מוטה שלו נמצאת גם היא.

כלומר בודק-איות אינו האורקל שהמנייה צריכה. הוא עונה "האם זה ניתן לאיות". המנייה שואלת "האם זה גם שם עצם". באנגלית אלה כמעט אותה שאלה, כי הרישיות מפרידה; בעברית אלה שאלות שונות, והמילון אינו מבחין ביניהן.

וכאן כמעט טעיתי טעות שנייה, בכיוון ההפוך. ההנחה המתבקשת היא שהאורקל הנכון אינו קיים כמדף, וזו הכללה שקל לעשות ולא לבדוק. בדקתי, והוא כן קיים: יש רשימות מילים עבריות שבהן שמות-עצם ושמות-פרטיים הם קבצים נפרדים, ואז השאלה מפסיקה להיות שאלת איות והופכת לחיתוך קבוצות.

הרצתי את 89 השמות מול רשימה כזאת, 158,346 שמות עצם מול 116,669 שמות פרטיים, אחרי שש בקרות שכולן עברו:

```
שדלחך, מחשב     שם-עצם, לא שם פרטי
אברהם, דניאל    שם פרטי, לא שם-עצם
רגליב, נפתדק    לא זה ולא זה
```

40 מתוך 89 השמות הם גם שמות עצם. 44.9%.

וזה משנה את המסקנה של הפרק, אבל לא את כיוונה. עם האורקל הנכון המנייה מסמנת 45% מהרשימה במקום 99%, וזה שיפור אמיתי. אלא שכלל שאומר "מסומן, לכן הסלם" עדיין מסלים כמעט מחצית מהפעולות ללא תנאי, וזה עדיין כיוון הכשל השני. הפתרון שהצעתי נגד הראשון מתכנס אל השני, אותה תבנית, קומה אחת מעלה — רק שעכשיו יש לזה מספר במקום הערכה.

ודבר אחד יצא מ-hspell שלא ציפיתי לו. במילון שלו התחיליות נגזרות מכללי הטיה ואינן ערכים בפני עצמן, ולכן אפשר לשאול שאלה צרה יותר: אילו צירופי תחילית-ועוד-שם הם בעצמם ערך-מילון נפרד, כלומר מילה עברית אחרת לגמרי. התשובה, לארבעה מהשמות בטבלה:

```
אור   -> מאור, שאור          תום  -> בתום, שתום, שמתום
גל    -> מגל                 דנה  -> לדנה
```

"מאור" הוא גוף מאיר, "שאור" הוא חומר תפיחה, "כתום" הוא צבע, "מגל" הוא כלי קציר. זה מצב כשל 1 בדיוק, מתועד הפעם מלקסיקון חיצוני שלא אני בניתי, ולא מקורפוס שכתבתי בעצמי כדי להדגים אותו.

14. ולמה באנגלית זה כן עובד

הרצתי את אותם עשרים שמות אנגליים באות קטנה, וקיבלתי אבחנה אמיתית:

```
התקבלו באות קטנה (השם הוא גם מילה)   13/20   rose, mark, bill, grace, dawn,
                                                 will, art, hope, robin, frank,
                                                          summer, sky, melissa
נדחו באות קטנה (שם בלבד)             7/20   jennifer, daniel, abraham,
                                                deborah, kevin, brian, samantha
```

`melissa` היא הפריט היחיד ברשימה שאינו אנגלית יומיומית; זה שמו של צמח, המליסה. שתים עשרה האחרות שגורות לגמרי, ובלעדיה היחס עדיין 12 מתוך 19 מול 7.

`Jennifer` באות גדולה מתקבל, `jennifer` באות קטנה נדחה, ו-`rose` באות קטנה מתקבל. הכתיב האנגלי מסמן שמות פרטיים ברישיות, ולכן הלקסיקון מפריד בין שתי הקטגוריות ואפשר לשאול אותו את השאלה. בעברית אין אותיות גדולות, שתי הקטגוריות יושבות באותה רשומה, והשאלה בלתי ניתנת לשאילה.

וזו בדיוק אותה צורה שראינו במצב 2. שם התיקון המקובל היה `\b`, והוא מת בעברית מפני ש-`\w` הוא ASCII. כאן האורקל המקובל הוא בודק-איות, והוא מת בעברית מפני שאין רישיות. פעמיים, בשתי שכבות שונות של הפתרון, מנגנון שנשען בשקט על תכונה של הכתב הלטיני.

ואותם 44.9% הם גם התשובה לשאלה שהשארתי פתוחה קודם, כשכתבתי שאני מעריך שמצב 3 שכיח יותר בעברית ואיני יכול להוכיח את זה. עכשיו אני יכול, לפחות על המדגם הזה: כמעט מחצית משמות המדגם הם גם שמות עצם. המדגם נכתב לפני שראיתי לקסיקון כלשהו, ולכן הוא לא נבחר כדי לייצר את המספר הזה.

וחשוב לי שיהיה ברור שזו איננה תכונה של מיקרוסופט. hspell, של נדב הראל ודן קניגסברג, הוא פרויקט חופשי שנבנה לאורך שנים ואין לו שום קשר לכלי שבדקתי מולו ראשון. יש בו 469,730 ערכים, וכל שישה עשר השמות המקראיים נמצאים בו. שני לקסיקונים שנבנו בנפרד, בידי אנשים שונים, מגיעים לאותה מסקנה.

מה שנשאר לסייג הוא הנתיב אליו. hspell אינו מחלק רשימת-מילים בנויה אלא מייצר אותה בזמן הידור, ולכן מדדתי מול מילון ה-Myspell שנגזר ממנו, זה שה-README שלו קובע במפורש שנוצר מנתוני Hspell גרסה 1.4. לא הידרתי את hspell עצמו.

15. עברית איננה מקרה מיוחד

ערבית מתנהגת כמעט זהה. אין בה תו ASCII, ולכן `\b` מחזיר `false` תמיד, והתחיליות אותו סיפור: "النور" מול השם "نور", ו"لسعيد" שבו ל' נדבקת לשם עצמו.

טורקית מעניינת יותר, ודווקא היא מדגימה את מצב 2 בצורתו המסוכנת. היא נכתבת באותיות לטיניות ולכן `\b` עובד, אבל ı, ğ, ş, ç, ö, ü אינן `\w`. וטורקית תקנית מפרידה סיומת משם פרטי בגרש, והמשתמשים מדלגים על זה:

```
\b "barış"  ב-"barış'a gönder"       -> false   בן שם
\b "barış"  ב-"barışa gönder"        -> true    בן שם, בלי גרש
\b "barış"  ב-"barışma sürecinde"    -> true    אינו שם
\b "güneş"  ב-"güneş'in raporu"      -> false   בן שם
```

הסתכלו על השורות. `\b` מחמיץ את האזכור האמיתי, תופס את מה שאינו שם, ומשנה תשובה לפי סימן פיסוק שהמשתמש אולי הקליד. הוא לא רועש, הוא הפוך.

וזה ההבדל המעשי מעברית. אצלנו מצב 2 אחיד, `false` על הכול, ולכן נתפס בבדיקה הראשונה. בטורקית הוא עובר בדיקה עם Can ועם Kaya, נשבר אצל Barış, ומתהפך לפי הגרש. כשל אחיד נתפס. כשל הפכפך משתחרר לפרודקשן.

16. איפה זה יושב ביחס למה שכבר מתועד

שום עובדה בפרקים הקודמים אינה חדשה, ואני מעדיף לומר את זה בעצמי.

ש-`\b` הפשוט אינו מספיק ליוניקוד כתוב בתקן עצמו, ובאופן חד יותר משחשבתי בהתחלה. UTS #18 מתאר את המנגנון הנאיבי, גבול בין תו `\w` לתו שאינו, ופוסק עליו `This is not adequate for Unicode regular expressions`. ומיד אחר כך בא **RL1.4**, שהוא חלק מ**רמת התאימות הראשונה**, ודורש ש-`<word_character>` יכלול את כל ערכי ה-Alphabetic ממסד-הנתונים של יוניקוד, את הספרות, וגם את U+200C ו-U+200D.

כלומר `\b` של JavaScript אינו "רמה 1 בלבד". **הוא אינו עומד ברמה 1.**

והרעיון שרשימת היתר רחבה מדי נכשלת פתוח יושב בקטלוג. הבאג שלי הוא, פשוטו כמשמעו, השוואה חלקית של מחרוזת: **CWE-187, Partial String Comparison**, שהתיאור שלו נוקב במפורש בהשוואה ל-`such as a substring`. זה `text.includes(name)` בדיוק, וזו רמת ההפשטה הנכונה למפות אליה — CWE-187 הוא Variant שמסומן ALLOWED למיפוי.

שווה לומר גם מה *לא* למפות. המועמד המתבקש הוא **CWE-697**, השוואה שגויה, וזה מה שהושטתי אליו יד תחילה. 697 הוא Pillar, ו-MITRE מסמן אותו **DISCOURAGED** במפורש, בנימוק שהוא ברמת הפשטה גבוהה מדי מכדי לתאר פגיעות ממשית. הוא נכון כמשפחה ושגוי כמיפוי.

בסביבה יושבים גם **CWE-183**, רשימת קלטים מותרים מתירנית מדי, **CWE-625**, ביטוי רגולרי מתירני, ו-**CWE-777**, ביטוי רגולרי בלי עוגנים, שקובע שביטוי לא מעוגן בהקשר של רשימת היתר יגרום לכשל של מנגנון ההגנה. 777 ו-183 אינם קשורים היררכית; הקשר עובר דרך 625 והוא קשר עמיתים.

ולתבנית הכללית, של באג התאמה שמתהפך להחלטת הרשאה, יש CVE בציון 9.8 לפי **CVE-2022-22978** :NVD ב-Spring Security. שם המנגנון הוא `.` מול תו שורה חדשה בשרתי סרוולטים, לא עיגון, ולכן אני מביא אותו כתבנית ולא כמקרה זהה.

וגם עצת-העמודה שבסוף הפרק על הפרודקשן אינה שלי. תיעוד כל הכרעות ההרשאה, ולא רק הדחיות, הוא נוהג מוכר: ל-OPA ול-Aserto יש פיצ'ר בשם decision logs שרושם את הקלט של כל הכרעה, ו-Cerbos רושם את אותו הדבר ביומן-הביקורת שלו.

וגם המעגל עצמו כבר מודל, חודשיים לפני שכתבתי את זה. טוראן מראה שהמבקר האנושי הוא אנדוגני למערכת — בלשונו, `the guard's own escalation policy degrades the very oracle it escalates to` — ומודד קצב הסלמה אופטימלי של 64%, כלומר שהסלמת-הכול גרועה מהאופטימום ולא רק יקרה ממנו. יש שם מספרים שאין לי, והם תומכים בדיוק במה שתיארתי בפרק המנייה.

מה שלא מצאתי כתוב, גם לא אצלו, הוא הכיוון השני. המודל שלו עוסק בפעולות שהוסלמו ובקיבולת של מי שמקבל אותן; הוא אינו נוגע בשאלה מה נרשם כשהשער **לא** מסלים. ושם, בשער קוטבי שמתעד רק בהסלמה, שיעור הפספוסים אינו רק לא-נמדד — הוא בלתי ניתן לרישום. בשער כזה הפספוס אינו פער ביקורת, הוא אדם שהוצא מהלולאה בלי שאיש ידע, ובלי שנשאר ממה לדעת.

17. המעגל

עכשיו אפשר לתאר את מה שקרה כאן במדויק, וזה לא מה שחשבתי בהתחלה.

התחלנו במצב כשל 1, שהוא שקט ומסוכן. תיקנו אותו, ותיקנו חזק מדי, והגענו למצב כשל 2, שהוא רועש ומעצבן. הצענו מנייה כדי לצמצם את הרעש, וגילינו שהיא מסמנת את רוב הרשימה ולכן מייצרת עוד רעש.

ותור שהכול מוסלם אליו אינו יציב. הוא מייצר חתימה אוטומטית. אדם שמקבל עשרים כרטיסים ביום עדיין קורא אותם, וזו תיבת דואר ולא משטר עייפות; אבל שער שמסלים את הכול אינו נשאר בעשרים. עייפות התראות מתועדת היטב בספרות מרכזי בקרת האבטחה, והעומסים שם הם מאות ואלפי התראות ביום; מנגנון ההתנוונות מתואר שם במדויק, ומאשרים ברפלקס עד שהשער נעשה דקורטיבי.

וכאן צריך לדייק, כי זו תזת הסיום ואיני רוצה למתוח אותה. מצב 2 אינו מתנוון בחזרה למצב 1 במדויק. באישור רפלקסיבי עדיין נוצר כרטיס: יש שם מאשר, יש חותמת זמן, ויש אי-הכחשה, ואפשר לחזור אחורה ולראות מי אישר מה. במצב 1 אין כלום, כי הכתיבה לא עברה בשער בכלל. ההתנוונות מאפסת את *איכות* ההכרעה האנושית, לא את העקבות שלה. אבל היא כן מאפסת את מה שהשער נועד להשיג, כי הוא נבנה כדי שאדם יחליט ולא כדי שיחתום. וזה קורה דרך האדם ולא דרך הקוד, בלי ששורת קוד אחת השתנתה.

ויש לזה תקדים מדויק בעולם האבטחה, לא רק אנלוגיה. ל-MITRE ATT&CK יש טכניקה ייעודית לזה, **T1621, Multi-Factor Authentication Request Generation**, והתיאור שלה נוקב במפורש ב-"MFA fatigue": התוקף מציף את תור האישורים של הקורבן עד שזה מאשר לבסוף כדי שזה ייפסק.

זו אותה לולאה בדיוק, רק שמופעלת במכוון. כלומר הצפת תור האישורים איננה רק תופעת לוואי של הנדסה זהירה מדי, היא וקטור תקיפה מוכר בפני עצמו — ושער שמסלים הכול מספק את התנאי שלו מראש, בחינם, בלי שאיש יצטרך לתקוף.

זה מעגל, ולא סימטריה. הכיוון מ-1 ל-2 עובר דרך מהנדס שמגיב חזק מדי. הכיוון מ-2 ל-1 עובר דרך אדם שנשחק. שני מנגנונים שונים לגמרי, ושניהם צפויים.

ולכן אי אפשר פשוט לבחור בין שני כיווני הכשל. היציאה היחידה היא להפסיק לשאול את השאלה על מחרוזת. לא "האם השם מופיע בטקסט", אלא לתת למנהל לבחור ישות מרשימה, ולעגן את הפעולה במזהה.

אלא שזה פותר מסלול אחד בלבד. במסלול האוטונומי אין מנהל שמקליד, ואין רשימה שנפתחת לפניו — יש אירוע שמפעיל ריצה. וזה בדיוק המסלול שסימנתי כמסוכן יותר, שבו הקשר-האירוע מחובר לפקודה למחרוזת אחת והסימון של הסניטייזר נמחק בדרך. שם "תן למנהל לבחור ישות" אינו אומר כלום.

מה שנדרש שם הוא שהמזהה ייווצר **במעלה הזרם**, ברגע שבוני-האירועים מנסחים את האירוע, וייסחב דרך הריצה כולה כשדה נפרד ולא כטקסט. או, אם אין מזהה כזה, שכל פעולה שנוגעת באדם תוסלם ללא תנאי במסלול הזה. שתי האפשרויות יקרות, ושתיהן עדיפות על שער שמחפש שם במחרוזת שאיש לא חתום עליה. זה מה שאנחנו בונים עכשיו.

18. מגבלות

הטענה הכבדה ביותר כאן היא סריקת-הפרודקשן, ולכן היא זו שדורשת את הגידור החזק ביותר. מתוך 766 הטקסטים, 764 הגיעו מעסק אחד, שלנו, שרשימת העובדים שלו היא צוות-הדגמה שנזרע לבסיס-הנתונים ולא צוות שנבנה מעצמו; שני הטקסטים הנותרים הגיעו מעסק-לקוח אחד. שני המקורות היחידים היו רציונלים של כרטיסי-אישור וסיכומי-ריצה, החלון היה שישה שבועות, ומתוך 451 ההתאמות לשם חשוף נמצאו 441 בטקסט שהמנוע עצמו כתב. את 143 חלונות-ההקשר קראתי בחלון של שלושה טוקנים לכל צד, לא בטקסט המלא.

כל אחד מהגורמים האלה מטה את התוצאה לאותו כיוון, ולכן האפס שדיווחתי — אפס מקרים שבהם ההתאמה התייחסה לשם-העצם ולא לאדם — אינו טענה חזקה. הראיתי למעלה ששגיאת שווא אינה יכולה להופיע בכרטיס-אישור כלל, כי כרטיס נוצר רק בהסלמה. האפס הזה מתאר מכפלה של רשימת-שמות מסוימת באוצר-מילים של תחום מסוים בחלון-זמן מסוים, ולא תכונה של השער.

שני הקורפוסים קטנים ובחירתי, עשרים וחמש שורות בסך הכול, ושניהם מפורסמים למעלה כדי שאפשר יהיה לחלוק עליהם. הם מדגימים מנגנון ואינם מדד.

תוצאות ה-`\b` אינן רגישות לקורפוס, כי הן נובעות מהגדרת `\w`. אפשר לאמת בשורה: `/\w/u.test("א")` מחזיר `false`.

יש פינה אחת שבה `\b` כן נורה על עברית: כשתו ASCII צמוד בלי רווח. `/\bדנה\b/u` מחזיר `true` על `"דנה1"`, על `"abcדנהabc"` ועל `"_דנה_"`. בטקסט עברי טבעי זה לא קורה, אבל זה קיים.

מניית השמות רצה מול שני לקסיקונים ולא מול קורפוס תדירויות, ושתי המדידות אינן זהות באופיין: אצל Word שאלתי בודק-איות מלא, שמחיל בעצמו את כללי ההטיה, ואצל hspell חיפשתי חברות ברשימת הערכים. מכיוון שהתחיליות שם נגזרות בכללי .aff ואינן ערכים, המספר 88 מתוך 89 הוא רצפה והרחבה מלאה יכולה רק להוסיף. באותו כיוון בדיוק, בדקתי עשרים רצפי-תחיליות חוקיים ולא את 259 הרצפים שהמקלף במימוש מקבל. מנגד, רשימת 89 השמות היא בחירתי; כתבתי אותה לפני שראיתי תוצאה כלשהי, והיא מתפרסמת בנספח כדי שאפשר יהיה לחלוק עליה.

גם המספר 40 מתוך 89 הוא רצפה. רשימת שמות העצם שהשתמשתי בה איננה שלמה: `מיבל` במובן כלי קיבול, `שני` במובן צבע, ו-`רדת` במובן ידידות אינם בתוכה, ושלושתם שמות עצם עבריים. כלומר החפיפה האמיתית בין שמות למילים גדולה מ-44.9%, ואיני יודע בכמה.

לעומת זאת הסיווג של רשימת העובדים בפרודקשן, "שלושה מתוך 24 הם מילים עבריות", נעשה בעין ולא מול לקסיקון. הוא לא הורץ מחדש מול המילון, ונשאר הערכה.

ושני הקורפוסים נכתבו בלי ניקוד ובלי תווים בלתי-נראים. עמודת הזיהויים האמיתיים, 8 מתוך 8, תלויה בכך: סימן ניקוד אחד בתוך אזכור אמיתי מפיל את **כל** השיטות, כולל `includes()`, והתוצאה היא הסלמה. כלומר גם הצד שנראה נקי בטבלה נמדד על טקסט נקי.

ומדידת ההחרגה של ה' רצה על חמישה שמות בלבד, שנבחרו משום שהם גם שמות עצם. 0 מתוך 5 מול 15 מתוך 15 הוא הפרש חד מכדי להיות מקרי, אבל הוא מדגים מנגנון ואינו אומד שכיחות. כמה מהאזכורים בפועל הם מיודעים — את זה לא מדדתי.

מדדתי מנוע אחד, של JavaScript. הכשל אינו של ביטויים רגולריים אלא של מנועים שבהם `\w` נשאר ASCII, ובהם JavaScript, PCRE בלי `PCRE_UCP`, Java בלי `UNICODE_CHARACTER_CLASS`, ו-Go. ב-Python 3 וב-NET. זה עובד; לג'ון ד' קוק יש פוסט שמדגים את זה ב-Python. שווה לדייק בו: המשפט שלו שהמטא-תווים `work as we would hope` נאמר על יוונית ובא עם סייג, `at least if they're Greek letters`, ולא על עברית. באותו פוסט הוא כן מראה מלכוד עברי משלו — ניקוד שנכנס בין האותיות ושובר התאמה פשוטה. הטבלה שלמעלה מובאת מהתיעוד ולא ממדידה שלי.

ולא בדקתי את `Intl.Segmenter`, שהוא ההמלצה של MDN לגבולות מודעי-שפה. זו הבדיקה הראשונה שהייתי מריץ הלאה.

19. נספח: 89 השמות שהורצו מול שלושת הלקסיקונים

הרשימה נכתבה לפני הרצה כלשהי, בשני גושים, ובלי לבדוק אף שם. שלוש ההרצות:

```
בודק-האיות של 89          Word מתוך 89 סומנו
מתוך 89 סומנו 88          hspell (מילון he_IL)
רשימת שמות-העצם           40 מתוך 89 הם גם שמות עצם
```

אברהם, יצחק, יעקב, משה, אהרן, דוד, שלמה, שמואל, יוסף, בנימין, יהודה, שמעון, ראובן, אליהו, יהושע, מרדכי, גדעון, יונתן, נתן, איתי, עומר, גיא, רועי, אורי, נדב, אסף, אלון, תומר, ניר, עידו, רון, שחר, יובל, אמיר, ליאור, ארז, עמית, דור, נועם, אביב, איתן, דניאל, גבריאל, רפאל, מיכאל, אריאל, אדם, יאיר, בועז, עידן, שרה, רבקה, רחל, לאה, מרים, אסתר, רות, נעמי, חנה, דבורה, יעל, תמר, אביגיל, נועה, שירה, מיכל, טל, אור, הדס, עדן, רוני, מאיה, אלה, שקד, ורד, קרן, סיון, הילה, דנה, אורלי, סיגל, מיטל, ליאת, שני, אפרת, גלית, סמדר, רותם, אביטל.

עשרים רצפי-התחיליות שנבדקו לכל שם הם כל הצירופים החוקיים של ו, ש, ואחת מ-כ, ב, ל, מ, כולל הרצף הריק. במפורש, כדי שהמכנה יהיה בר-שחזור:

```
שמ  של  שב  שב  שכ  ש  מ  ל  ב  כ  (ריק)
דשמ  דשל  דשב  דשב  דש  דמ  דל  דב  דכ  ו
```

מקורות

Gesenius' Hebrew Grammar §125 — https://en.wikisource.org/wiki/Gesenius%27_Hebrew_Grammar/125._Determination_of_Nouns_in_general._Determination_of_Proper_Names

MDN, Word boundary assertion — https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Regular_expressions/Word_boundary_assertion

UTS #18, Unicode Regular Expressions — https://www.unicode.org/reports/tr18/

John D. Cook, Searching Greek and Hebrew with regular expressions — https://www.johndcook.com/blog/2020/09/20/unicode-regex/

Itamar Syn-Hershko, Challenges with indexing Hebrew texts (HebMorph, 2010) — https://code972.com/blog/2010/05/challenges-with-indexing-hebrew-texts-hebmorph-part-1-18

More, Seker, Basmova, Tsarfaty, TACL 7 (2019), 33-48 — המאמר שמתאר את YAP — https://aclanthology.org/Q19-1003/

YAP, מעבדת ONLP — https://github.com/OnlpLab/yap

Tsarfaty, Seker, Sadde, Klein, What's Wrong with Hebrew NLP? And How to Make it Right — https://arxiv.org/abs/1908.05453

hspell, מאת נדב הראל ודן קניגסברג — https://hspell.sourceforge.net/

מילון he_IL בפורמט Myspell, שנוצר מנתוני Hspell 1.4 — https://github.com/LibreOffice/dictionaries/tree/master/he_IL

Microsoft Office, כלי ההגהה לעברית (MSSP7HE.LEX) — https://support.microsoft.com/en-us/office/language-accessory-pack-for-microsoft-365-82ee1236-0f9a-45ee-9c72-05b026ee809f

Elasticsearch, תוסף הלמטיזציה לעברית — https://www.elastic.co/search-labs/blog/elasticsearch-lemmatization-hebrew-analyzer

MDN, Intl.Segmenter — https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Intl/Segmenter

Tariq, Baruwal Chhetri, Nepal, Paris, Alert Fatigue in Security Operations Centres, ACM Computing Surveys 57(9), 2025 — https://dl.acm.org/doi/10.1145/3723158

Turan, Oversight Has a Capacity, arXiv 2606.08919 (2026) — המאמר שממדל את המעגל ומכמת אותו — https://arxiv.org/abs/2606.08919

MITRE ATT&CK T1621 — הצפת תור אישורים כווקטור תקיפה — https://attack.mitre.org/techniques/T1621/

eyaler, hebrew_wordlists — שמות עצם ושמות פרטיים כקבצים נפרדים — https://github.com/eyaler/hebrew_wordlists

OPA, Decision Logs — https://www.openpolicyagent.org/docs/management-decision-logs

Aserto, Decision Logs — https://docs.aserto.com/docs/decision-logs-guide/api/overview

Cerbos, Audit block — https://docs.cerbos.dev/cerbos/latest/configuration/audit.html

CWE-187: Partial String Comparison — https://cwe.mitre.org/data/definitions/187.html

CWE-697: Incorrect Comparison (Pillar, מסומן DISCOURAGED למיפוי) — https://cwe.mitre.org/data/definitions/697.html

CWE-183: Permissive List of Allowed Inputs — https://cwe.mitre.org/data/definitions/183.html

CWE-625: Permissive Regular Expression — https://cwe.mitre.org/data/definitions/625.html

CWE-777: Regular Expression without Anchors — https://cwe.mitre.org/data/definitions/777.html

CVE-2022-22978 (Spring Security, CVSS 9.8) — https://nvd.nist.gov/vuln/detail/CVE-2022-22978