מיני פרויקט - נושאים במערכות הגנה לרשת

הרחבת דפדפן לזיהוי פישינג מבוסס מודל שפה מתקדם (LLM)

אוניברסיטת בן גוריון בנגב, סמסטר אביב 2025

מגישים: דניאל פרוידנטל, נגה פורת

מרצה: דורון אופק

דף תיעוד

מטרת הכלי ותפקידו

הכלי הוא תוסף לדפדפן Google Chrome שמטרתו לזהות אתרי פישינג (Phishing) בזמן אמת ולספק למשתמש שכבת הגנה אקטיבית במהלך הגלישה. התוסף מבוסס על מודל שפה מתקדם (LLM) באמצעות ממשק ה־API של Google Gemini, אשר מנתח את תוכן הדף ופרטי המטא־דאטה (כגון URL, כותרת ותוכן טקסטואלי) לצורך קביעת רמת הסיכון של האתר.

המערכת תומכת בשלושה מודלי Gemini AI עם מנגנון fallback:

1. **Gemini 2.5 Pro:** איכות גבוהה לניתוח מורכב
2. **Gemini 2.5 Flash:** איזון בין ביצועים לעלות ומהירות
3. **Gemini 2.5 Flash Lite:** מהיר וחסכוני (ברירת מחדל)

במקרה שמודל אחד נכשל לקבל תשובה (בגלל שגיאת API או timeout למשל), המערכת עוברת אוטומטית למודל הבא בשרשרת.

האם הכלי עונה למטרה?

כן. לפי תיאור הפרויקט והרכיבים שפותחו, הכלי מספק מענה ישיר למטרה שהוגדרה. השילוב בין בינה מלאכותית לניתוח תוכן בזמן אמת, יחד עם מנגנון סריקת אתרים וממשק התרעה אינטואיטיבי, מאפשרים זיהוי מהיר, אפקטיבי וידידותי למשתמש של אתרי פישינג.

התראות ויזואליות ברורות מעניקות למשתמש חוויית שימוש נוחה ומספקות שכבת הגנה משמעותית מפני איומי פישינג.

תכונות מתקדמות נוספות:

- מערכת דירוג לגיטימיות המתקבלת על ידי ניתוח האתר בידי מודל שפה טבעי (Legitimacy Score 0-100).
- הגדרת thresholds מותאמים אישית (Safe/Caution Thresholds) המגדירים את רגישות התוסף לדירוגי לגיטימיות נמוכים-בינוניים.
- בחירת מודל AI מתוך מספר אפשרויות, בהתאם להעדפת המשתמש בין מהירות ועלות לדיוק גבוה.
- אפשרות לצפות בהיסטוריית הסריקות הקודמות.
- ממשק הגדרות מקיף להתאמת התוסף להעדפות המשתמש.

בעיות ואתגרים

במהלך הפיתוח והבדיקות, נתקלנו במספר בעיות ואתגרים אשר נאלצנו להתמודד איתם:

1. תלות במפתח API חיצוני

הכלי דורש מהמשתמש להנפיק ולהגדיר מפתח API לשירות Google Gemini.

- **האתגר:** המשתמש נדרש להנפיק ולהגדיר מפתח API בעצמו. תהליך זה עלול להיות מורכב למשתמשים שאינם טכניים, ושינויי מדיניות מצד Google (כגון תמחור או הגבלות שימוש) או בעיות טכניות אחרות עלולים להשפיע על זמינות הכלי.
- **הפתרון שבחרנו:** פיתוח ממשק הגדרות אינטואיטיבי בתוסף עם אימות מפתח API בזמן אמת, הוספת מדריך מובנה וברור ליצירת המפתח, ובדיקת נגישות לכל המודלים הזמינים טרם ההח
- **פתרון עתידי:** במידה והמוצר יושק רשמית, המנגנון ינוהל **בצד השרת** כדי למנוע חשיפה למשתמש ולהפחית מורכבות.

2. עלויות שימוש ומגבלות

מודלי שפה טבעיים (LLM) הם שירותים בתשלום המבוססים על מספר ה־tokens המעובדים בקריאה.

- **האתגר:** שימוש אינטנסיבי עלול להוביל לעלויות גבוהות ולצריכת משאבים מוגברת.
- **הפתרון שבחרנו:** שימוש במודל החינמי של Gemini בשלב הפיתוח על חשבון ירידה מסוימת בדיוק המודל.
- **פתרון עתידי:** שילוב שכבת Caching ותחזוק של רשימת דומיינים בטוחים בענן, המתעדכנת גלובלית ממקורות אבטחה חיצוניים ומסריקות כלל המשתמשים, לצמצום עלויות ולשיפור הדיוק.

3. דיוק הניתוח (False Positives/False Negatives)

המודל לא נבנה במיוחד לאיתור פישינג ולכן טעויות אפשריות קיימות.

- **האתגר:**
  - *False Positive:* אתרים לגיטימיים מסווגים בטעות כפישינג ופוגעים בחוויית המשתמש.
  - *False Negative:* אתרי פישינג אמיתיים שאינם מזוהים מותירים את המשתמש חשוף.
- **הפתרון שבחרנו:** השקעה רבה בהנדסת הפרומפט הנשלח אל Gemini עם פרטי האתר (Prompt Engineering) תוך כדי בדיקות מרובות ושילוב אפשרות למשתמש להגדיר Safe Threshold & Caution Threshold בהתאם להעדפותיו האישיות.
- **פתרון עתידי:** פיתוח מנגנון למידה מתמשך (Learning Component) שיוסיף מאגר דומיינים לגיטימיים וחשודים ויבצע אופטימיזציה לאורך זמן.

4. אתרים מורכבים או דלי מלל

ניתוח מבוסס טקסט מתקשה באתרים ויזואליים או דינמיים.

- **האתגר:** אתרים המבוססים על תמונות ואנימציות מקשים על המודל להסיק תובנות.
- **הפתרון שבחרנו:** שימוש במנגנוני Fallback לאיסוף פרטי האתר הנסרק.

5. הנדסת פרומפטים (Prompt Engineering)

התאמת השאילתות ל־API קריטית לקבלת תוצאות עקביות ומדויקות.

- **האתגר:** ניסוח השאילתות דרש ניסויים רבים מאוד כדי להפחית רעש בתשובות, לקבל פורמט אחיד, לדייק את פירוט ה־Reasoning ולשפר את דיוקו של ה־Legitimacy Score.
- **הפתרון שבחרנו:** ביצוע בדיקות מרובות, שינויים הדרגתיים ושימוש במנועי בינה מלאכותית נוספים ליצירת פרומפט מותאם ספציפית למשימה, עם דרישה לפורמט JSON מובנה הכולל Legitimacy Score ו-Reasoning.
- **פתרון עתידי:** המשך אופטימיזציה על בסיס ניסויים עם פורמטים שונים ושקילת שימוש במהנדס פרומפטים מקצועי לשיפור איכות התשובות לאורך זמן.

6. הרשאות חסרות על סריקת עמודים

חלק מהאתרים בעלי הגבלות מחמירות יותר על הרשאות הסריקה שלהם.

- **האתגר:** הגבלות אבטחה בדפדפן מנעו במקרים מסוימים ניתוח מלא של דפים.
- **הפתרון שבחרנו:** יישום מנגנוני Fallback מתקדמים במקרה וסריקת העמוד נכשלת, המבצעים סריקה בטכניקה אחרת גם אם פחות מדויקת או מקיפה.

7. פרטיות ואבטחת מידע

שמירה על פרטיות המשתמש היא עקרון קריטי.

- **האתגר:** איזון בין איסוף מידע מספיק לניתוח אמין לבין הקפדה על פרטיות המשתמש.
- **הפתרון שבחרנו:** העברת מינימום נתונים אפשריים ל־API, ללא מזהים אישיים או מידע רגיש.
- **פתרון עתידי:** הטמעת מנגנון אנונימיזציה מתקדם ופילטור אוטומטי של מידע הנסרק בטעות (כגון, צינזור כתובות אימייל, שמות, כתובות, סיסמאות וכו׳).

8. חוסר עקביות בתוצאות ה־Confidence Level

במהלך פיתוח מנגנון דירוג האתרים, יישמנו שתי ישויות נפרדות:

Verdict – החלטת המודל האם מדובר באתר שהוא לגיטימי, או חשוד כניסיון פישינג.

Security Level – רמת הביטחון של המודל ב-verdict אשר נתן לאותו אתר.

- **האתגר:** בבדיקות גילינו שתשובות המודל לא היו עקביות – אותו אתר קיבל לעיתים תוצאות שונות, מה שהוביל לחוסר יציבות בקביעת רמת הביטחון ובחוויית המשתמש.
- **הפתרון שבחרנו:** ביטלנו את שני המשתנים הנפרדים ואיחדנו אותם למשתנה יחיד - **Legitimacy Score**, שמחזיר ערך מספרי בין 0 ל־100 לכל אתר. ציון זה משמש למדידת רמת הביטחון, והחלטת ה־Verdict נגזרת ממנו. שינוי זה שיפר משמעותית את אחידות התוצאות והקל על המשתמשים להבין את סיווג האתר.
- **פתרון עתידי:** נשקלת הטמעת מנגנון כיול אוטומטי (Auto Calibration) המבוסס על למידה מצטברת מנתוני אמת בשטח, על מנת לשפר את היציבות והדיוק של הציון לאורך זמן.

יתרונות וחסרונות של הכלי

יתרונות:

- **זיהוי בזמן אמת:** הכלי מספק שכבת הגנה מיידית בעת גלישה לאתרים חדשים ומפחית חשיפה לאיומי פישינג בזמן אמת.
- **ממשק משתמש ברור:** התראות ויזואליות בולטות (באנרים צבעוניים) והסבר מפורט על סיבת האזהרה מאפשרים למשתמש להפעיל שיקול דעת נוסף.
- **שמירה על פרטיות:** הכלי פועל בצד הלקוח (Client-side) ואינו שולח מידע אישי לשרתים חיצוניים, למעט פרטי דף לצורך ניתוח.
- **גמישות בסריקה:** מאפשר למשתמש לבחור בין סריקה ידנית מתפריט התוסף לבין הפעלה מהירה באמצעות קליק ימני, עם הצגת התראה ויזואלית מתאימה.
- **שקיפות:** הצגת הסיבות לסיווג האתר (חשוד או לגיטימי) מגבירה את אמון המשתמש ומאפשרת הבנה טובה יותר של החלטת המערכת.
- **היסטוריית סריקות:** הכלי שומר סריקות אחרונות ומאפשר למשתמש לעיין בתוצאות עבר ולוודא האם אתר כבר נותח בעבר.
- **מערכת דירוג מתקדמת:** Legitimacy Score מספרי (0-100) עם הסברים מפורטים ו-color coding ברור.
- **הגדרות מותאמות אישית:** אפשרות להגדיר ערכי סף Safe/Caution Thresholds ובחירת מודל AI המותאם להעדפות המשתמש.
- **מנגנון Fallback חכם:** מעבר אוטומטי בין מודלי AI במקרה של כישלון.
-

חסרונות:

- **תלות במפתח API:** הצורך בהגדרת מפתח חיצוני מהווה חסם כניסה עבור חלק מהמשתמשים ודורש ידע טכני בסיסי.
- **תלות בתקשורת אינטרנט:** הכלי דורש חיבור רציף לשרתי Google Gemini, כך שזמינות ה־API מהווה נקודת תורפה אפשרית.
- **הגנה לא הרמטית:** תוקפים מתוחכמים עשויים לעקוף את מנגנון הזיהוי, לדוגמה ע"י הזרקת טקסטים מטעים ("אתר זה אינו אתר פישינג") שיכולים להשפיע על מודל השפה אם אינו מותאם כראוי.

- **ביצועים ועלויות:** סריקות רקע וקריאות API מרובות עשויות להשפיע על מהירות הגלישה ולצרוך משאבי מערכת. שימוש מוגבר גם עלול לגרור עלויות גבוהות בשירותי API בתשלום.

- **מורכבות הגדרות:** ריבוי אפשרויות ההגדרה עלול לבלבל משתמשים מתחילים.

תיאור אופן פעולת המערכת

תהליך הפעולה של התוסף מחולק למספר שלבים עיקריים:

1. התקנה והגדרה

לאחר התקנת התוסף, המשתמש נדרש להזין בחלון ההגדרות את מפתח ה־API לשירות Google Gemini. שלב זה כולל אימות מפתח API בזמן אמת ובדיקת נגישות למודלים השונים. המשתמש יכול גם להגדיר ערכי הסף המתאימים לו אישית (Safe/Caution Thresholds) ולבחור את המודל בו רוצה להשתמש.

2. הפעלת סריקה

ניתן להפעיל סריקת דף בשתי דרכים:

- **סריקה ידנית:** לחיצה על כפתור "Analyze Page" בחלון הקופץ (Popup) של התוסף.
- **סריקה מהירה:** שימוש בקליק ימני על הדף ובחירה באפשרות "Scan page for phishing".

3. איסוף מידע

קובץ ה־`content.js`, שמוזרק לכל דף אינטרנט בזמן הטעינה, אוסף את הנתונים הנדרשים לניתוח:

- כתובת ה־URL של הדף ופרטי המטא־דאטה שלו (metadata).
- כותרת העמוד (Page Title).
- הטקסט הגלוי למשתמש (Visible DOM Text).
- Suspicious Elements: זיהוי iframes, קישורים חיצוניים, טפסי כניסה ומילוי פרטים.
- Content Flags: ניתוח מילות דחיפות, איומים, שגיאות כתיב נפוצות.

4. שליחה לניתוח

קובץ ה־`background.js` מקבל את המידע מקובץ ה־`content.js` ומבצע קריאת API מאובטחת לשירות Google Gemini. המערכת מנסה תחילה את המודל המועדף על המשתמש, ובמקרה של כישלון עוברת אוטומטית למודל הבא בשרשרת (Pro → Flash → Flash Lite).
הקריאה כוללת **Prompt מובנה** המורה למודל לנתח את התוכן ולאתר סימני פישינג בהתאם לפרמטרים שנקבעו, עם דרישה לקבל תשובה בפורמט JSON שיותאם לניתוחה בקלות.

5. קבלת תוצאה

Google Gemini מחזיר תשובה בפורמט JSON הכוללת:

- **Legitimacy Score (רמת לגיטימיות):** מספר בין 0 ל-100 המדרג את ציון הלגיטימיות שהמודל נתן לאתר שנסרק בהתאם לפרומפט שהוגדר מראש ולפרטים שנסרקו מהעמוד.
- **Reasoning (נימוקים):** רשימת הסיבות והאינדיקטורים שהובילו לרמת הלגיטימיות שניתנה.

6. הצגת התרעה למשתמש

קובץ ה־`content.js` מקבל את התוצאה מ־`background.js` ומציג באנר ויזואלי בראש הדף:

- **Legitimate**: האתר קיבל רמת לגיטימיות גבוהה מאוד.
- **Uncertain**: האתר קיבל רמת לגיטימיות בינונית.
- **Phishing**: האתר קיבל רמת לגיטימיות נמוכה מאוד.

הבאנרים כוללים הסברים מפורטים, ציון לגיטימיות, ופעולות מומלצות למשתמש.

7. עדכון ממשק המשתמש

המערכת מעדכנת את אייקון התוסף עם אינדיקטורים ויזואליים:

- ✔️ (ירוק): אתר בטוח
- ❓ (כתום): אתר חשוד
- ❗ (אדום): אתר פישינג

היסטוריית סריקות

התוסף שומר את הסריקות האחרונות ומאפשר למשתמש לצפות בתוצאות עבר בחלון התוסף (Popup UI) לצורך מעקב ובקרה, כולל פרטי דומיין, תוצאה, וציון לגיטימיות.

תיעוד הקוד והרכיבים המרכזיים

הפרויקט בנוי ממספר רכיבים עיקריים, כל אחד אחראי על חלק ספציפי במערכת התוסף:

`manifest.json`

- **תפקיד:** קובץ התצורה הראשי של התוסף.
- **אחריות מרכזית:**
  - הגדרת הרשאות נדרשות (storage, tabs, scripting).
  - הצבעה על קובץ הרקע (Background Script), קבצי התוכן (Content Scripts) וממשק המשתמש (Popup, אייקונים).
  - הגדרת הגישה ל־Google Gemini API.
  - הגדרת תפריט הקשר (Context Menu) לסריקה מהירה.

`background.js`

- **תפקיד:** "המוח" של התוסף – מנהל את הלוגיקה המרכזית ופועל ברקע.
- **אחריות מרכזית:**
  - ניהול מצב התוסף ותקשורת פנימית בין הרכיבים.
  - טיפול בבקשות מה־Popup ומה־Content Script.
  - ביצוע קריאות ה־API לשירות Google Gemini עם מנגנון Fallback בין מודלים.
  - עדכון דינמי של אייקון התוסף
  - יצירת תפריט ההקשר (Context Menu) עבור קליק ימני לסריקה מהירה.
  - ניהול Storage Management להגדרות והיסטוריית סריקות.

`popup.html` | `popup.css` | `popup.js`

- **תפקיד:** רכיבי ה־UI של חלון ה־Popup שנפתח בעת לחיצה על אייקון התוסף.
- **אחריות מרכזית:**
  - `popup.html`: מבנה התוכן (HTML) של הממשק.
  - `css/popup.css`: עיצוב הממשק וחוויית המשתמש.
  - `src/popup.js`: לוגיקת הממשק המתקדמת – שמירת והגדרת מפתח ה-API, הפעלת סריקה ידנית, הצגת היסטוריית סריקות אחרונות.

`content.js` | `styles.css`

- **תפקיד:** אחראי על איסוף מידע מתקדם מהדף, ניתוח תבניות תוכן חשודות, הצגת באנרים ויזואליים, מעקב אחר שינויי URL דינמיים, הצגת התראות זמניות, ותקשורת דו-כיוונית עם Background לטיפול בשגיאות.
- **אחריות מרכזית:**
  - איסוף נתוני הדף (URL, כותרת, טקסט גלוי).
  - שליחת המידע ל־Background Script לניתוח.
  - הצגת באנרים ויזואליים בראש הדף (אדום/צהוב/ירוק) בהתאם לתוצאת הסריקה.
  - הצגת התראות זמניות שונות (שגיאות API, התרעות שונות למשתמש).