

מטלה תכנותית - קורס 20940

(ממ"ן 16, מבוא לאבטחת המרחב המקוון)

ניתוח השוואתי של מנגנוני אימות מבוססי סיסמאות

מבוא

בעבודה זו נתמקד בניתוח מנגנוני אימות מבוססי סיסמאות. כידוע, החוזק והעיצוב של מנגנוני האימות משפיעים באופן ישיר על עמידותן של מערכות בפני שיטות תקיפה נפוצות כגון Brute-Force ו-Password-Spraying. מטרת הפרויקט היא לבדוק כיצד שיטות שונות לאחסון סיסמאות ואימות משתמש מתפקדות מול תקיפות אלו, ולנתח את היתרונות הנוספים של אמצעי הגנה משניים כגון הגבלת קצב, נעילת חשבון, CAPTCHA ו-TOTP (Time-based One-Time Password).

פרויקט מחקר זה משלב אלמנטים תיאורטיים ומעשיים. תצטרכו לחקור מנגנוני הגנה ושיטות תקיפה שחלקם טרם הכרתם, ואח"כ לתכנן, לממש ולנתח ניסוי מבוקר המדמה תקיפות אימות בסביבה מבודדת. לשם כך תאספו ותנתחו מדדים כמותיים כגון זמן-לפריצה, מספר ניסיונות, ניסיונות לשנייה ושיעור הצלחה עבור רמות חוזק סיסמה שונות ותצורות הגנה מגוונות.

שימו לב: הפרויקט מתבצע בזוגות (בלבד). כל חריגה דורשת אישור של המנחה.
חשוב: אם קיבלתם אישור לבצע לבד, לא יתקבלו בקשות להקלה בסגנון: עשיתי את הפרויקט לבד ולכן לא הספקתי X Y Z וכדומה.
תאריך הגשה יפורסם באתר בנפרד.

מטרות הפרויקט

מטרה מרכזית

לבצע ניסוי הניתן לשחזור, המשווה בין מנגנוני גיבוב ואימות סיסמאות (למשל: bcrypt, Argon2, SHA-256+salt), בוחן את השפעת מנגנוני ההגנה השונים כגון salt, pepper, הגבלת קצב, נעילת חשבון, CAPTCHA ו-TOTP, ומבצע ניתוח סטטיסטי של הנתונים המתקבלים.

מטרות נוספות

- למדוד זמן-לפריצה ושיעור הצלחה תחת מגוון מנגנוני הגנה.
- לכמת את ההשפעה של כל מנגנון הגנה בנפרד ובשילוב.
- להעריך את הפשרה בין השימושיות והביצועים בכל גישה.
- להפיק דוח מחקר תמציתי המסכם את הממצאים הניסויים.

תיאור המשימות

עליכם לבצע (כאמור בזוגות) את המשימות הבאות:

- תכנון והקמה של שרת אימות וירטואלי התומך במספר מנגנוני אחסון סיסמאות.
- יצירת סט נתוני משתמשים בעלי סיסמאות חלשות, בינוניות וחזקות.
- סימולציה של תקיפות Brute-Force ו-Password-Spraying תחת תצורות שונות.
- הפעלת מנגנוני הגנה נוספים (Rate-Limiting, נעילת חשבון, CAPTCHA, TOTP, Pepper) וחזרה על מדידות.
- רישום וניתוח של מדדי ביצוע ותוצאות התקיפה.
- הכנת כל תוצרי החובה והגשתם.

תוצרי הפרויקט

כל צוות יגיש:

- **דוח מחקר (6-8 עמודים):** כולל מבוא, מתודולוגיה, תוצאות, ניתוח, דיון, שיקולים אתיים ומקורות.
- **לוגים גולמיים:** נתונים מובנים (JSON או CSV) של ניסיונות האימות והתוצאות.
- **קובצי קונפיגורציה:** תצורה אנונימית של סביבת העבודה (ללא מפתחות סודיים).
- **מצגת:** סיכום קצר של מטרות, תצורה וממצאים.
- **סרטון הדגמה (2-4 דקות):** הצגת הסביבה והתוצאות המרכזיות.

הנחיות אתיות ובטיחותיות

- כל הניסויים יתבצעו אך ורק על מערכות מקומיות או וירטואליות השייכות לסטודנטים.
- אסור לחלוטין לבצע תקיפות על רשתות או מערכות חיצוניות.
- יש להשתמש אך ורק בנתוני משתמשים מלאכותיים ולא בנתוני אמת.
- הדוח חייב לכלול הצהרת עמידה בכללי האתיקה.

מושגים מרכזיים והסברים

TOTP – Time-based One-Time Password

הגדרה: מנגנון אימות דו-שלבי היוצר קודים זמניים הנגזרים מסוד משותף. כל קוד תקף לפרק זמן קצר (לרוב 30 שניות).

רלוונטיות: מוסיף גורם דינמי עצמאי שמונע כמעט לחלוטין הצלחת תקיפות Brute-Force.

היבטי מימוש: שלב רישום (החלפת סוד), סנכרון זמנים (סטיית שעון), טווח סבילות בבדיקה (עד כמה השרת מוכן לקבל קוד שמקורו בזמן מעט שונה).

מקורות: RFC 6238, וספריות נפוצות כגון pyotp.

Password-Spraying

הגדרה: ניסיון להשתמש במספר קטן של סיסמאות נפוצות על פני משתמשים רבים בניגוד ל-Brute-Force הממוקד בחשבון יחיד.

רלוונטיות: מדגים כיצד Rate-Limiting ברמת משתמש אינו מספיק ללא מגבלות גלובליות.

מוקדי ניתוח: שיעור הצלחה לחשבון, מספר ניסיונות כולל, והשפעת מדיניות ההגנה.

מקורות: OWASP, ובלוג Microsoft Security Research.

Pepper

הגדרה: ערך סודי גלובלי הנוסף לסיסמה לפני הגיבוב ונשמר בנפרד ממסד הנתונים.

הבדל מ-Salt: ה-Salt הוא ייחודי למשתמש ונשמר לצד הגיבוב; Pepper גלובלי ונשמר כסוד.

רלוונטיות: מחזק את אחסון הסיסמאות במקרה של דליפת מסד נתונים.

מקורות: OWASP Password Storage Cheat Sheet, NIST SP 800-63B.

פרוטוקול ניסוי (תהליך מוצע)

שימו לב: הפרוטוקול שלהלן הינו הצעה שמומלץ מאוד להיצמד אליה. אם מסיבה כלשהי אתם מעדיפים לבצע את הניסוי תוך שימוש בפרוטוקול שונה, עליכם להגדיר אותו במדויק ולקבל אישור ממנחה הקבוצה שלכם לפני המימוש.

- הגדרת GROUP_SEED ייחודי ושילובו בקבצי התצורה.
 - GROUP_SEED יהיה Bitwise XOR של שני מספרי הת.ז של חברי הקבוצה.
- יצירת סט משתמשים מבוקר (סיסמאות חלשות/בינוניות/חזקות).
 - עליכם להגדיר מה נכלל בכל קטגוריה.
- הרצת בדיקות בסיס (ללא מנגנוני הגנה).
- הוספת מנגנון הגנה יחיד בכל פעם והרצת התקיפות מחדש.
- רישום תוצאות עקבי בכל התצורות.
- סיכום ממצאים בטבלאות ותרשימים.

מטרת GROUP_SEED ושימוש

מטרות עיקריות:

- **ייחודיות ושלמות:** מבטיח הבדל ושחזור בין קבוצות הסטודנטים.
- **בדיקת מקוריות:** מאפשר לוודא שהניסוי בוצע עצמאית.
- **עקיבות:** מופיע בלוגים, בקונפיגורציה ובדוחות לקישור התוצרים לקבוצה מסוימת.
- **שחזור:** מסייע להריץ מחדש ניסויים במדויק.

הערה: GROUP_SEED אינו סוד (לא Pepper), אלא מזהה.

שימוש מומלץ:

- הכללת ה-Seed במטא-נתונים של קבצי תצורה/נתונים.
- ציון ה-Seed ב-README.
- שימוש בו לסימון תרשימים או קובצי תוצאות.
- אחת מהסיסמאות תהיה GROUP_SEED.

מדדים ולוגים

- מדדים נדרשים: סך ניסיונות, זמן כולל, ניסיונות לשנייה, זמן-לפריצה, שיעור הצלחה, השהיה, שימוש ב-CPU/זיכרון.
- פורמט לוגים: CSV או JSON הכולל Timestamp, שם משתמש, סוג גיבוב, תוצאה (הצלחה/כישלון), מצב מנגנוני ההגנה.
- תיעוד: יש לכלול תיאור קצר של אופן איסוף הלוגים ואימותם.

ניתוח נתונים וסטטיסטיקה

- זמני תקיפה:
 - ממוצע
 - חציון
 - אחוזנים (לדוגמה, האחוזון ה-90 הוא הזמן שמתחתיו נמצאים 90% מהתקיפות)
- השוואת התפלגויות בין מנגנוני ההגנה:
 - כלומר להשוות את צורת ההתפלגות של זמני התקיפה (או מדדים אחרים) בין תצורות הגנה שונות - למשל, ללא הגנות, עם Rate-Limiting, עם TOTP, עם Pepper וכו'.
- דיון בתוקף הניסוי:
 - מגבלות: מה היו המגבלות של הניסוי? לדוגמה: עוצמת חומרה מוגבלת, מספר קטן של משתמשים, שימוש בכלי תקיפה פשוטים, זמן ניסוי קצר, הנחות פשטניות שלא מייצגות סביבה אמיתית, וכו'.
 - מקורות טעות: מה יכול היה לפגוע בדיוק של התוצאות? לדוגמה: חוסר יציבות ברשת או בשרת, אי-סנכרון זמן (במיוחד ב-TOTP), טעויות לוגיקה בקוד התקיפה או ההגנה, נתונים מזהמים או לוגים לא עקביים, וכו'.
 - שחזור הניסוי: כיצד מישוהו אחר יוכל להריץ את אותו ניסוי ולקבל תוצאות דומות? זה כולל: תיעוד מלא, שימוש בקונפיגורציות ברורות וחוזרות, שמירה על GROUP_SEED, רשימת גרסאות של כל כלי, קוד עקבי ולוגים מסודרים.

מקוריות העבודה

- ניהול היסטוריית גרסאות (רצוי אך לא חובה ב-Git) עם התקדמות רציפה.
- צירוף צילומי מסך או קטעי וידאו קצרים של שלבי העבודה.
- תיעוד ניסיונות כושלים או לא-מכריעים.

קריטריוני הערכה

קריטריון	תיאור	משקל
מתודולוגיה ותכנון	בהירות התצורה הניסויית, שליטה במשתנים (כלומר שמירה על כל הפרמטרים קבועים בזמן בדיקת גורם אחד), תיעוד	25%
תהליך המימוש	נכונות, יכולת שחזור, שלמות הנתונים (כלומר הבטחת דיוק ושלמות הנתונים לאורך כל הניסוי, ללא שינויים או אובדן מידע)	20%
ניתוח ודין	איכות ההסברים, עומק סטטיסטי, פירוש התוצאות	25%
איכות הדוח	מבנה, בהירות, הצגה, מקורות	20%
הדגמה	איכות הסרטון ו/או המצגת	10%

מבנה דוח מוצע

• תקציר ומוטיבציה	• ניתוח ודין
• רקע	• שיקולים אתיים
• תצורה ניסויית	• מסקנות ועבודה עתידית
• מתודולוגיה	• מקורות
• תוצאות	• נספחים (לוגים, טבלאות, תרשימים)

תכנית עבודה מוצעת

שבוע	משימה	תוצר
1	סקירת ספרות, בחירת כלים	מתווה ראשוני
2-3	הקמת שרת, יצירת מערך משתמשים	לוגים בסיסיים
4	בדיקות Brute-Force	דוח ביניים
5	הוספת הגנות (Rate-Limit, Pepper, TOTP)	לוגים מעודכנים
6	ניתוח נתונים וכתיבת דוח	טיוטת דוח
7	מצגת והדגמה	הגשה סופית

טיפים מעשיים (Do / Don't)

:Do

- לשמור לוגים מסודרים, מתויגים ובעלי חותמת זמן.
- לוודא סנכרון זמן עבור TOTP.
- לבדוק מקרי קצה (סיסמאות חזקות, שילובי הגנות).

:Don't

- לא לשתף סיסמאות אמיתיות או להשתמש בשרתי אימות חיצוניים.
- לא להשתמש בכלי AI ללא ציון מפורש.

הרחבות אפשריות

- בדיקת השפעת פרמטרים של Argon2 (זיכרון, איטרציות).
- בדיקת ביצועים תחת מגבלות משאבים (למשל, הגבלת CPU).
- מימוש התקפות נוספות (SQLi, XSS וכו').

שימו לב: אם אתם מממשים את ההרחבות (או חלק מהן) יש לציין זאת במפורש בדו"ח. הדבר עשוי להעלות את הציון לפי שיקול דעתו של המנחה. ניתן להציע הרחבות מעניינות שלא מצויינות כאן. עם זאת, ההרחבות אינן תחליף ויבדקו רק אם בוצעו כל המטרות העיקריות.

נספח - הגדרות שימושיות

שימו לב: בדומה לאמור בתיאור הפרוטוקול, ההגדרות וההנחיות המופיעות כאן הינן אופציונליות ונועדו להקל על התכנון שלכם. עם זאת כאמור לעיל מומלץ להשתמש בהן כמות שהן. לשם פיתוח האתר ניתן להשתמש בטכנולוגיות כגון Flask ו-SQLite (או דומות להן). הבהרה: כל הפרמטרים הקריפטוגרפיים בפרויקט זה הם ברמת מעבדה בלבד, ומכוונים להדגמה ולמידה. הם אינם מתאימים לשימוש בסביבת פרודקשן.

1. שטח התקיפה וסוגי התקיפות

- **שטח התקיפה:** ONLINE בלבד: REST API עם נקודות קצה `register`, `/login/`, `/login_totp`. אין גישה ל-DB dumps. אין התקפות נוספות (SQLi, XSS וכו').

• סוגי התקפות:

- **Brute-force** (מכוון לחשבון בודד דרך קריאות REST).
- **Password-spraying** (מספר סיסמאות נפוצות על פני חשבונות רבים).
- אופציונלי: **Distributed spraying** (סימולציה דרך threads/VMs).
- אופציונלי: תקיפות אופליין.

הערה: למרות האמור בסעיף זה, צוות המעוניין לפתח סביבה וירטואלית המדמה שרת (ולא סביבה וובית) רשאי לעשות כן.

[הכוונה היא שבמקום לבנות ממשק ווב, הצוות יכול ליצור סימולציה של שרת באמצעות תוכנה וירטואלית או סביבה מקומית שמחקה את פעולת השרת בפועל. זה מאפשר לבדוק את המערכת בסביבה מבוקרת ללא צורך בדפדפן או ב-UI web. כמובן, יש לציין זאת במפורש בדו"ח.]

2. דרישות מינימום לשרת
 - נקודות קצה: `register`, `/login`, `/login_totp/` (או מקביל HTTP REST).
 - תמיכה במצבי אחסון סיסמאות הניתנים להגדרה: `SHA-256+salt`, `bcrypt (cost=12)`, `Argon2id`.
 - תמיכה בוויסותים: `CAPTCHA`, `rate-limiting`, `lockout`, (באמצעות `token` סימולציה), `TOTP` (לכל משתמש), `pepper` (מטעינה מהסביבה ולא מאחסן בבסיס הנתונים).
 - רישום כל ניסיון אימות ל-`JSON lines` (`attempts.log` או `CSV`) עם שדות: `timestamp`, `group_seed`, `username`, `hash_mode`, `protection_flags`, `result`, `latency_ms`.
3. מודל תוקף ומגבלות
 - יכולות התוקף: לקוח HTTP אונליין בלבד. אין פיצוח אופליין (אין DB dumps). כל ההתקפות מופנות לשרת הבדיקה.
 - ניתן להוסיף בדיקות אופליין כהרחבה.
 - מגבלות משאבים/זמן (לכל תצורת ניסוי):
 - ברירת מחדל: עד 50,000 ניסיונות לכל תצורה.
 - גבול: עד 1,000,000 ניסיונות או עד 2 שעות ריצה לפי המוקדם מביניהם.
 - קיום תגובות השרת: אם השרת מחזיר `captcha_required`, הלקוח צריך לפעול לפי אוטומציית ה-`CAPTCHA` (סעיף 5).
4. פרמטרים ברירת מחדל ל-`Hash`
 - `SHA-256 + per-user salt`.
 - `bcrypt: cost = 12`.
 - `Argon2id: time = 1, memory = 64 MB, parallelism = 1`.
 - הערה: השתמשו בערכים אלו כברירת מחדל להשוואה אך מומלץ לנסות ערכים נוספים.
5. אוטומציה ל-`TOTP` ו-`CAPTCHA` (אך להריץ ניסויים אוטומטיים)
 - **TOTP אוטומציה:** `totp_secret` יסופק ב-`users.json` לחשבונות בדיקה. סקריפטים יכולים להשתמש בסוד כדי לייצר טוקנים תקפים (`pyotp` או מקבילה). לסימולציה של סנכרון זמן, ניתן להוסיף לשעון המקומי סטייה מבוקרת של ± שניות, להריץ את מנגנון הסנכרון, ולדווח על הסטייה שהוטלה, התיקון שבוצע והשגיאה הסופית.
 - **סימולציית CAPTCHA:** השרת לא ידרוש פתרון CAPTCHA אנושי. במקום זאת: אחרי X ניסיונות כושלים יחזיר השרת `{ "captcha_token": "captcha_required": true, "...": "..." }`. לטעינה אוטומטית הלקוח יכול להשיג טוקן תקף דרך `endpoint` מנהל: `admin/get_captcha_token?group_seed=<GROUP_SEED/>` (מותר רק לבדיקות ודורש `GROUP_SEED` נכון). שימוש בטוקן ממשיך את התקיפה האוטומטית, כך ניתן למדוד את ההשפעה בלי חסימה אנושית.
6. `GROUP_SEED` (מעקב ומקוריות)
 - לכל קבוצה יהיה `GROUP_SEED` ייחודי שהוא כאמור XOR של מס' ת.ז. של מחברי הקב'. הקבוצה חייבת להטמיעו ב-`users.json`, לכלול אותו בלוגים (שדה `group_seed`), ולהזכירו ב-`README` ורצוי בלפחות אחד מה-`git commit messages`. המטרה: אימות מקוריות ו-`traceability`.

7. דרישות ל-dataset

- יצירת **30 חשבונות בדיקה**: 10 חלשים, 10 בינוניים, 10 חזקים. לתעד סיבה לסיווג של כל חשבון ב-README. אין להשתמש בנתוני משתמשים אמיתיים.

8. דרישות דיווח מינימליות

- יש להגיש, עבור כל שילוב של מצב ה-hash והגנות מופעלות/כבויות (toggles):
- קובץ **attempts.log** גולמי עם timestamps ו-GROUP_SEED.
- טבלת סיכום תוצאות: total_attempts, attempts_per_second, time_to_first_success (אם יש), success_rate_by_category.
- לחשב את הזמן הממוצע (average latency_ms) עבור כל מצב hash נפרד.
- אם פיצוח מלא אינו מעשי בגבולות, יש לספק **extrapolation**: חישוב זמן משוער לפיצוח על בסיס measured attempts/sec והנחות keyspace; יש לפרט הנחות.
- הצהרה אתית קצרה המאשרת ציות לכללי הבטיחות.