נוהל פיתוח מאובטח למערכות ויישומים

בסביבות Android ו-iOS

תאריך עדכון: 26/10/2022

גרסה: 1.1

תאריך השינוי	שינוי ע"י	גרסה	פרטי השינוי
30/06/2019	קומסק	1.0	כלל התוכן
26/10/2022	אלכס לוין	1.1	בקרה טכנית

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

תוכן עניינים

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

1. מבוא

1.1 סוגי אפליקציות מובייל

אפליקציות מובייל שייכות בדרך כלל לאחת מהקטגוריות הבאות:

1. אפליקציית Native: אפליקציה אשר מפותחת ומקומפלת עבור פלטפורמה/מערכת הפעלה מסוימת ולרוב משתמשת ב-API מוגדר כ-iPhone. כלומר, אפליקציית Native שתפותח ל-iPhone לא תוכל לרוץ על Android, אלא אם פותחה לאפליקציה גרסת Android נפרדת (וכן להיפך). מסיבה זו ישנו צורך לפתח שתי אפליקציות נפרדות, אחת לכל מערכת הפעלה. אפליקציות אלה לרוב מותקנות דרך חנות אפליקציות.
2. אפליקציית Hybrid: אפליקציה הכתובה בקוד אחד (ולא כפול כב-Native) ויכולה לרוץ במספר מערכות הפעלה שונות. זאת אומרת שניתן לכתוב קוד פעם אחת ולהשתמש בו כדי להריץ את האפליקציה הן על מכשיר ה-Android והן ב-iPhone.
3. Web App: אתר אינטרנט המופעל דרך דפדפן (ניגשים לאתר באמצעות הזנת כתובת URL) ואשר מותאם מבחינה פונקציונאלית וגרפית עבור המכשיר הנייד. בד"כ נכתב ב-HTML5.

1.2 אבטחת מידע במכשירי סלולר

מכשיר סלולר מכיל מס' רכיבים שאחראים על מידע רגיש של המשתמש, ביניהם: מצלמה, מיקרופון, מסך מגע, GPS ועוד. גישה למכשירו של המשתמש עלולה לגרום לפגיעה בפרטיותו ולמעקב אחר השיחות הטלפוניות שלו, תמונות ווידאו שצילם ושקיבל (ובכלל – גישה ל-file system), לוח שנה המכיל פרטי פגישות ואירועים, מיקומי המכשיר, אנשי קשר, היסטוריית המקלדת (cache), הודעות ה-SMS, חשבונות בנק ומיילים, דואר אלקטרוני, היסטוריית הדפדפן וכו'.

ניתן לקטלג את הסיכונים המרכזיים לנושאים הבאים (עולם ה- Web הקלאסי ועולם המובייל דומים מאוד בהרבה אספקטים):

1. התקפות מבוססות רשת ו- Web.
2. וירוסים ו- Malware.
3. פגיעה באמינות ובסודיות המידע.
4. התקפות Denial of service.
5. התקפות הנדסה חברתית.
6. איבוד וזליגת מידע.
7. סיכונים הנובעים מאפליקציות ושירותים חיצוניים.

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

8. התקפות Spam של SMS ו- Voice והתקפות הנוצרות על ידי מקורות אלה.
9. סוגיות של פרטיות ורגולציה.
10. פעולות כגון Rooting ו- Jailbreak אשר יכולות לגרום לחשיפה להתקפות שונות.
11. התקפות המנצלות פגיעויות בשפת התכנות

1.3. אבטחת שכבת האפליקציה

אבטחת שכבת האפליקציה כוללת מספר נושאים מרכזיים אשר עליהם מתבסס כל תהליך הפיתוח והנדסת המערכת:

1. יישום מתודולוגיית בדיקות שמתייחסת למצבי קצה ו- abuse cases, לא רק use cases.
   - במטרה להתמודדות בין היתר עם runtime manipulation והתקפות המנצלות מקרי קצה שונים.
2. ביצוע בדיקות קלט מדויקות (הן בצד שרת והן בצד לקוח).
3. שימוש בשפות תכנות הנחשבות ל"בטוחות" או לחילופין התייחסות לבעיות מובנות בשפה, כגון: buffer, integer overflow.
4. ביצוע סקרי קוד טרם העלאת האפליקציה לאוויר ותקופתית – אנליזת static and binary code, שימוש ב- Fuzzers.
   - כמו כן, ביצוע מבדקי חדירות – Penetration tests.
5. הרצת אפליקציות תוך מתן מינימום הרשאות, כמו כן, ביטול/שליטה בהרשאות גבוהות מובנות שלעיתים מגיעות ב- API המובנה.
   - לא לאפשר ריצה תחת root או system admin.
6. ניהול נושא ההזדהות וה-sessions בצורה מבוקרת.
7. יישום אבטחת תווך והגנה על מידע אשר נשמר ע"ג המכשיר (שימוש מושכל בכלים קריפטוגרפים, שמירה על פרטיות המשתמשים ועל אמינות הנתונים).
8. שליטה בערוצי התקשורת הנכנסים והיוצאים (הן בצד השרת והן בצד האפליקציה).
   - ברמת פורטים פתוחים, משאבי רשת וספריות אליהם אנחנו פונים ו/או חשופים אליהם.
   - מומלץ לבצע תרשים מערכת/מודל איומים.
9. יישום מנגנון בקרה (Log) מבוקר, תוך מניעת שמירת מידע מסווג או לא הכרחי (אשר יכול גם להשפיע על ביצועים).
10. יישום מנגנוני code obfuscation ו-anti-tampering במטרה להתמודד עם Reverse Engineering.

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

2. הנחיות כלליות לפיתוח מאובטח בעולם הסלולר

2.1 ריצת האפליקציה במכשיר פרוץ (rooted/jailbreak)

מצב Rooted (באנדרואיד) / Jailbreak (ב-iOS) הינו מצב שבו מכשיר הסלולרי נפרץ על ידי המשתמש, ולכן מהווה בעיית אבטחה כשלעצמה. על מנת לצמצם את משטח התקיפה האפשרי של האפליקציה, יש לוודא כי המכשיר עליו היא מותקנת אינו פרוץ. אפליקציה שאינה בודקת את מצב המכשיר ומאפשרת את התקנתה על גבי מכשירים פרוצים עלולה לגרום לאפשרות של קריאת מידע רגיש מצד הלקוח.

לכן, יש לבדוק אם המכשיר נמצא במצב פרוץ לפני טעינת האפליקציה, וכן מומלץ שלא לאפשר את ריצת האפליקציה במידה שהמכשיר נפרץ.

2.2 בקרה על הלוג

מנגנון התיעוד (הלוג) של האפליקציה משמש לרישום פעולות שמתבצעות באפליקציה, בין אם באופן יזום על ידי המשתמש, ובין אם באופן אוטומטי על ידי רכיבים שונים. קיימות התקפות שונות שתכליתן פגיעה במידע הרשום במנגנון התיעוד, למשל על ידי הוספת רשומות "זבל", על ידי הצפת הלוג, על ידי שינוי הערכים וכו'. כמו כן, גורמים זדוניים עלולים לנסות לקרוא את הנתונים על מנת לחלץ מידע רגיש.

- על האפליקציה לוודא כי מידע רגיש אינו נרשם למנגנון התיעוד. במידה וישנו צורך לתעד מידע כזה, הוא ירשם בצד השרת.
- אם הרישום מתבצע על המכשיר הנייד, יש להצפין את המידע על ידי הצפנה חזקה, אולם יש להשתדל להימנע מכך.
- כמו כן, על האפליקציה להגביל את גודל הלוג ולבצע בקרת קלט, במטרה להתמודד עם ניסיונות Denial of Service והשחתת הלוג באמצעות החדרת תווים לא רלוונטיים.
- כאשר האפליקציה מועברת לסביבת הייצור, יש למחוק/לבטל הוראות תיעוד בהן היא משתמשת במצב DEBUG, וכן יש לוודא כי מצב ה- DEBUG מבוטל באופן כללי.

2.3 טיפול בשגיאות

חריגה המתרחשת בתוכנית נחשבת לפעולה יקרה מבחינת משאבים, ובנוסף עלולה לגרום לתקיעת התוכנית. לפיכך, כן תפיסת השגיאה וטיפולה הינה נדבך חשוב באבטחת מידע, אם על מנת לוודא את זרימת התוכנית והמשאבים המוקצים לה ואם על מנת למנוע את קריסת התוכנית (למנוע Denial of Service).

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

2.4. עירפול קוד המקור של האפליקציה (Obfuscation)

עירפול קוד המקור של אפליקציות הינו הליך אבטחתי חשוב הנועד למנוע מתוקפים ניתוח והבנה של קוד המקור של האפליקציה (באמצעות Reverse Engineering). הפעולה מחליפה את שמות המשתנים, שמות המחלקות ושמות המתודות בטקסט אקראי שאינו מובן לקריאה. פעולה זו גורמת לקוד המקור להיות בלתי קריא, ומקשה על הבנת אופן פעולת האפליקציה.

קוד מקור של אפליקציה שאינו מעורפל עלול להיקרא על ידי גורם זדוני, ועל ידי כך הוא יוכל להבין את דרכי פעולתה לצורך מיפוי החולשות הקיימות באפליקציה וניצולן (למשל, באמצעות manual Code Review ו/או Static Code Analysis).

כחלק משלבי הפיתוח הסופיים, מומלץ להוסיף את תהליך עירפול הקוד בטרם הפצתו ללקוחות בכדי להסתיר את מהות האפליקציה ואת האופן בו היא פועלת על מנת למנוע שינויים זדוניים בקוד.

2.5. יישום Two-Factor Authentication

אימות דו-שלבי הינו שיטת אימות שמחייבת שימוש ביותר ממנגנון אימות יחיד, המספקת אמצעי אבטחה נוסף לכניסה לחשבון של המשתמש. אימות בשיטה זו דורש שני סוגי הזדהות לחשבון המשתמש (או יותר, במקרה של Multi-Factor Authentication), ומבוסס על גורמי הזיהוי הבאים:

- Something you know (הסיסמא לחשבונו של המשתמש, PIN code)
- Something you have (טלפון חכם, כרטיס חכם וכו', שההוכחה שהם נמצאים ברשות המשתמש באה לידי ביטוי בכך שהמשתמש מזין את סיסמת ה-OTP[1] שנשלחה אליו כהודעת SMS, למשל)
- Something you are (כהתקן ביומטרי של טביעת אצבע, סריקת הפנים)

בעת ביצוע פעולות רגישות, יש לדרוש נתון נוסף אחד לפחות מהנ"ל, ע"מ לוודא בסבירות גבוהה יותר שהבקשה שנשלחה אכן הגיעה מן המשתמש הלגיטימי.

[1] One-Time Password

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

3. דגשים לפיתוח מאובטח ב-Android

3.1 הצפנה

- יש לבצע שימוש באלגוריתמים הבאים בלבד:
  - AES עבור הצפנה סימטרית (אורך מפתח מינימלי של 256-bit)
  - RSA עבור הצפנה א-סימטרית (אורך מפתח מינימלי של 4096-bit)
  - SHA-2 עבור hash חד כיווני (אורך מפתח מינימלי של 384-bit)

- להלן דוגמא להצפנת מחרוזת input באמצעות הצפנת AES:

```
cipher = Cipher.getInstance("AES/GCM/NoPadding");

cipher.init(Cipher.ENCRYPT_MODE, skeySpec);

encryptedBytes = cipher.doFinal(Utilities.nullPadString(input).getBytes());
```

- אין לבצע שימוש באלגוריתמים שפותחו בצורה עצמאית.
- יש לבצע שמירה מאובטחת של מפתחות ההצפנה הנמצאים בשימוש המערכת.
- אין להשתמש באלגוריתם AES עם Cipher Mode של ECB (Electronic Code Book, המצפין כל בלוק באופן נפרד).
- בשימוש ב-GCM mode באלגוריתם AES, חובה לייצר מחדש IV (Nonce) עבור כל פעולת הצפנה.
- לא די בכך שהמידע בין המשתמש לשרת יעבור בצורה מוצפנת, אלא על האפליקציה לבדוק את תקינות התעודה מהבחינות הבאות (אכיפת SSL Pinning):
  - יש לבדוק את התאריך, ה-hostname וה-trusted CA של ה-Certificate.
  - ניתן להשוות את ה-Public Key שחזר מהשרת עם ה-Key הצפוי.
  - ניתן לבצע Hash ל-SubjectPublicKeyInfo (SPKI) ולהשוותו ל-Hash המצופה לצורך obfuscation.

במידה והמשתמש הסכים להתקין את התעודה במכשיר שלו, יוכל התוקף להאזין לתווך התקשורת שבין המשתמש והשרת, ולקבל את כל המידע המועבר בצורה שאינה מוצפנת (כולל מידע רגיש), וזאת ללא צורך בהתחברות לאפליקציה.

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

3.2. הגנה על משאבים הכוללים תשלום כספי מצד המשתמש

מערכת ההפעלה Android מספקת APIs שמאפשרים גישה למשאבים שונים בתשלום – SMS, טלפון, ארנק דיגיטאלי, ממשקי NFC (Near field communication) וכדומה. ניצול לרעה של ממשקים אלה ע"י תוקף יכול לעלות למשתמש כסף רב וגם לפגוע במוניטין החברה שלה האפליקציה שייכת. על כן יש להגן על האפליקציה וממשקיה אשר מתוכנתים לעבוד מול רכיבים ושירותים אלה בצורה שיטתית. האפליקציה צריכה ליידע את המשתמש על שימוש במשאבים בתשלום, אם באמצעות הרשאות שמוגדרות ב- Android ואם באמצעות הרשאות פרטיות (custom permissions). במידה ונעשה שימוש בהרשאות הפרטיות, יש להגדיר אותן כ-dangerous ולספק למשתמש תיאור בשפה לא-טכנית, שיבהיר לו את ההשלכות של אפשור הגישה למשאבים, כמודגם בקטע הקוד הבא[2]:

```
<permission xmlns:android="http://schemas.android.com/apk/res/android"

            android:name="il.el-al.android.ACCESS_FLIGHTS"

            android:label="@string/accessperm_label_flights"

            android:description="@string/accessperm_desc_flights"

            android:protectionLevel="dangerous">

</permission>

<string name="accessperm_label_flights">read your next flights data</string>

<string name="accessperm_desc_flights">Allows an application to retrieve the details
    of the flights you haven't yet taken. Malicious applications may monitor your
    geographical location and absence from home.

</string>

<uses-permission android:name="il.el-al.android.ACCESS_FLIGHTS">

</uses-permission>
```

מומלץ לצמצם את מספר ההרשאות למינימום הנדרש. כמו כן, יש לאפשר למשתמש לראות את החיובים שבוצעו עקב השימוש שלו באפליקציה, ולשמור לוגים של הפעולות בצד השרת (ו/או על המכשיר הנייד, באופן מאובטח – חתומים דיגיטאלית ונגישים רק לאפליקציה עצמה).

3.3. שימוש ב-Shared Clipboard

אפליקציה יכולה לאפשר למשתמשיה להעתיק פרטים (כדוגמת שם המשתמש) באמצעות ה-Clipboard. אולם, על אף הנוחות שבכך, שימוש ב-Clipboard משותף עשוי לאפשר לאפליקציות אחרות המותקנות על מכשיר ה-Android לגשת למידע המועתק, ובכך לקבל גישה לנתונים רגישים כפרטי הזדהות למערכת

[2] מדמה מידע של זמני הטיסות של המשתמש, שיכול להוות נכס רב לפורצים לבתים (אין במתן דוגמא זו קריאה לעידוד המעשה).

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

מסוימת. אפליקציה זדונית עלולה לנצל זאת ולשמור פרטים אלו אודות המשתמשים. לפיכך, אם אין צורך לאפשר למשתמש להעתיק את הערכים או שהעתקתם כ-clear text תחשוף אותם בצורה מסוכנת, יש לבטל את אופציית copy-paste עבור שדה הקלט הרלוונטי.

ניתן להשתמש בקוד הבא עבור גירסא 3.0 (API level 11) ומעלה:

```
mEditText.setCustomSelectionActionModeCallback(new Callback() {

   public boolean onPrepareActionMode(ActionMode mode, Menu menu) {

      return false;

   }

   public void onDestroyActionMode(ActionMode mode) {

   }

   public boolean onCreateActionMode(ActionMode mode, Menu menu) {

      return false;

   }

   public boolean onActionItemClicked(ActionMode mode, MenuItem item) {

      return false;

   }

});
```

3.4. יישום מנגנון Logout ו-Session timeout

- לצורך סיום מפורש של פעילות המשתמש, מקובל ליישם כפתור יציאה (logout) אשר מנתק את ה-Session של המשתמש משרת המערכת, מוחק את מזהי המשתמש ומאלץ אותו לבצע הזדהות מחודשת בעת חזרה לעבודה מול האפליקציה. תפקידו של מנגנון ה-logout, בין היתר, הוא לוודא שאכן ה-Session הקיים אינו תקין יותר (כלומר אינו יכול לשמש להמשך הפעילות), וכן למחוק כל מידע רגיש שהיה בשימוש האפליקציה: מזהי משתמש, פרטי חשבון, נתוני טרנזקציות, מפתחות צופן וכדומה. המחיקה צריכה להתבצע גם עבור משתנים זמניים בזיכרון העבודה, וגם עבור קבצים זמניים או קבצי הגדרות, בהם נכתב המידע.
- בנוסף, מכשירים ניידים נוטים ללכת לאיבוד ולהישאר ללא השגחה במקומות שונים, לכן יש לנעול את האפליקציה לאחר מספר דקות שהיא אינה בשימוש (~7 דקות) ולמחוק את המידע אשר תואר מעלה. הנעילה למעשה תנתק את ה-Session של המשתמש, כך שכאשר הוא או גורם המנסה לפרוץ לחשבונו ינסו להמשיך לעבוד באפליקציה, יהיה צורך בביצוע הזדהות מחדש. הגדרה זו נקראת: Session Idle Timeout.

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

3.5. הגנה על תעבורת התקשורת

- בכל שימוש בתקשורת יש להצפין את רמת תווך התקשורת באמצעות TLSv1.2.
- אין להגדיר כי האפליקציה תעבוד במצב Debug.

3.6. הגבלת הרשאות

על מנת להגן על המידע של המשתמש וכן על הנתונים של האפליקציה עצמה, יש לשאוף למינימום ההרשאות הנדרשות לצורך הפעולה התקינה. כאשר משתמש מוריד אפליקציה למכשיר המריץ מערכת הפעלה Android, המערכת מציגה בפניו אפשרות לאשר או לדחות מתן גישה למשאבים שונים ואף רגישים של המכשיר. בצורה זו, המערכת מאפשרת למשתמש הפשוט לקבל החלטות שעלולות לחשוף אותו לפרצות. לכן, על האפליקציה לבקש מ- Android רק את הרשאות המערכת ההכרחיות.

- יש להגדיר הרשאות פרטיות (custom permissions) כ-dangerous במידה והן עלולות לסכן את המשתמש, את המידע שלו או את המכשיר הנייד שלו.
- יש הימנע מלהריץ את ה-GUI עם הרשאות גבוהות.
- יש להימנע ממתן הרשאות רחבות לגישה לקבצים ולהגדרות, ובייחוד הרשאות גורפות, כגון:
  - MODE_WORLD_READABLE
  - MODE_WORLD_WRITEABLE

  הרשאות מסוג זה מאפשרות לכל אפליקציה שעל המכשיר לגשת לקבצים של אותה האפליקציה.
- בדוק כי קבצים רגישים אינם נגישים מחוץ לאפליקציה, וכן קיים מידור גם בתוכה (אלמנטים של GUI לא יבצעו גישה אל קבצים כאלה). האפליקציה צריכה להגן על עצמה באופן פרו-אקטיבי מפני ניסיונות גישה של קוד זדוני.

3.7. הגנה על רכיבי האפליקציה

בספריה הראשית של הפרויקט יש לאפליקציה קובץ בשם AndroidManifest.xml, בו מוגדר מידע על האפליקציה. בקובץ זה מצהירים על הרכיבים (קומפוננטות) שמהם האפליקציה מורכבת:

- Intents
- Activities
- Content Providers
- Broadcasts
- Services

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

3.7.1. Intents

- **העברת מידע רגיש באמצעותם**

Intents הם המנגנון של Android להעברת מידע בין תהליכים, הם משמשים להפעלה של פעולות באפליקציות, יידוע אפליקציות אחרות על אירועים, תקשורת עם services הרצים ברקע, וגישה למידע דרך ContentProviders. Intents כשלעצמם אינם מחזיקים בהגדרות של אבטחה, ולכן זהו תפקידן של הקומפוננטות האחרות לוודא ולהגביל את הרשאות התקשורת הפנימית.

ב-Intent מפורש (Explicit) הכוונה היא שהרכיב באפליקציה ששולח את ההודעה מציין את השם של מקבל ההודעה באופן מפורש.

ב-Intent מרומז (Implicit) הרכיב השולח מציין את הפעילות שה-Intent אמור ליזום. כש-Implicit Intent נשלח, הוא מועבר דרך מסננת – Intent Filter שמכוון אותו לרכיב שיכול לטפל בו. מנגנון זה מאפשר קוד גמיש בו אין צורך לדעת מראש את זהות היעד של הודעה.

- יש להימנע מהעברת מידע רגיש באמצעות Intents שכן גורמי צד ג' יכולים ליירט ולקרוא אותם. במקרה בו מועבר מידע רגיש באמצעות Intent בין ה-components של אפליקציית המשרד, יש לדאוג להצפינם.
- אין לשלוח ו/או לקבל מידע רגיש מהאפליקציה אותה מפתחים לאפליקציה חיצונית.
- יש להשתמש ב-Explicit Intent בלבד.
- אין להשתמש ב-FLAG_GRANT_READ_URI_PERMISSION או ב-FLAG_GRANT_WRITE_URI_PERMISSION. סוגי הרשאות אלו מאפשרים לתכנית חיצונית לקרוא ולכתוב מ-URI מסוים שבתוך ה-Inetnt, במידה והתכנית מצילחה ליירט את אותו ה-Intent.

ב-Android קיים מנגנון בשם autocorrect אשר נועד להקל על המשתמש באמצעות תיקון של שגיאות כתיב נפוצות. בעוד שמנגנון זה עוזר למשתמש, הוא נחשב למסוכן משום שהמידע נשמר בו בצורה בלתי מאובטחת, וזאת כחלק ממנגנון הלמידה העצמית שלו. במידה וגורם זדוני יצליח להשיג גישה למכשיר (או לחלופין ע"י אפליקציה זדונית המותקנת על המכשיר), ניתן יהיה לקרוא את המידע שנשמר ב-autocorrect, לרבות פרטי המשתמש עבור ההתחברות לאפליקציה.

לפיכך, מומלץ לבטל את מנגנון ה-Autocorrect באמצעות שינוי שיטת הקלט (באחת משתי הדרכים להלן):

- Android:inputType="text|textNoSuggestions"
- Android:inputType="textVisiblePassword"

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

- **הגבלת השימוש בהם**

תקשורת בין תהליכים (IPC) מאפשרת לרכיבים השונים בתוך מערכת ההפעלה של מכשירים מבוססי Android לתקשר ביניהם. התקשורת יכולה להתבצע באמצעות מספר דרכים, כאשר האפשרות הנפוצה היא השימוש ב-Intents המהווים את ההודעות הנשלחות בין הרכיבים לצורך קבלת ושליחת מידע. בעזרת Intents האפליקציה יכולה להתחיל תהליכים או שירותים, וכן להאזין לפעולות במערכת.

בעת השימוש ב-Intents יש חשיבות עליונה על הגנתם מפני קריאות זדוניות. ללא הגנה מתאימה ניתן יהיה לשנות את המידע שעובר לאפליקציה באופן שעלול להשפיע על הלוגיקה הפנימית שלה. לכן, יש חשיבות שה-Intent של ה-Splash Screen יגביל את הקריאות אליו, אחרת גורם זדוני עלול לנצל פגיעות זו באמצעות שליחת קריאות ל-"SplashActivity", ובכך לאתחל את מסך הטעינה של האפליקציה. ביצוע פעילות זו מספר מרובה של פעמים ימנע מהמשתמשים מלהשתמש באפליקציה, תוך מחשבה שמקור הבעיה באפליקציה של אותה החברה, מה שעלול להסב נזק תדמיתי לאירגון הנ"ל.

- מומלץ לבדוק את זהות התהליך אשר קורא לכל Intent, ושלא לבצע שימוש ב-Implicit Intents.

ע"מ להגדיר Intent כ-Explicit, יש להחיל את ההגדרה הבאה בקובץ ה-Manifest :

```
<activity>

  android:name=".ActivityTest"

  android:exported="false"

  android:label="@string/app_name" >

       <intent-filter>

              <action android:name="android.intent.action.SEND" />

              <category android:name="android.intent.category.DEFAULT" />

       </intent-filter>

  </activity>
```

- Pending Intents

ניתן להגביר את רמת האבטחה על ידי שימוש ב- PendingIntent במקום Intent, אשר מונע שליחה של הודעות על ידי רכיבים זדוניים בשמם של רכיבים אחרים. PendingIntents נשלחים עם מזהה של התהליך שיצר אותם מלכתחילה, ולכן אם האפליקציה חושפת ממשק שמאפשר לרכיבים אחרים לקבל התרעת callback, מומלץ לעשות שימוש ב-PendingIntent על מנת להעביר את האחריות על ווידוא מקור ההודעה לרכיבים אלה.

יש לציין באופן מפורש את נמען ההודעה על ידי Intent.setComponent() .

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

3.7.2. Activities

Activity מאפשר להציג רכיבים על מסך המשתמש. ממשק המשתמש של אפליקציות ב-Android מורכב מ-Activities, שאותם הן מערכת ההפעלה והן אפליקציות אחרות יכולות להפעיל, עם אפשרות לציפייה לתוצאה. ההפעלה יכולה להתבצע ישירות או דרך Intent. ה-Activities מוגדרות בתוך קובץ ה-AndroidManifest.xml של האפליקציה, והנראות שלהן כלפי חוץ תלויה במספר גורמים.

- יש צורך להגדיר עבור כל רכיב של המערכת את סיווגו כפרטי או פומבי (Public/Private) בהתאמה, אין להשאיר רכיב לא מוגדר.
- על מנת לוודא כי התוכן של Activity רגיש ופנימי יהיה נגיש רק לאפלקיציה עצמה ולא כלפי אפליקציות אחרות, יש יש לקבוע את דגל הייצוא (exported) כשלילי באופן מפורש, כמו בהגדרה הבאה:

```
<activity android:name="SendMySMS" android:exported="false />
```

- מומלץ להגביר את רמת האבטחה של ה-Activities על ידי בקשת אישור מהמשתמש טרם ביצוע פעולות הגורמות לשינויים. אם הפעלה של Activity באמצעות Intent עלולה לגרום לנזק, יש לאכוף הרשאה ספציפית לביצוע ההפעלה. כמו כן, יש לזכור כי Intent זהו קלט לא בטוח, ואין לסמוך על התוכן שלו מבלי לבצע בדיקות קלט מדוקדקות.
- על מנת למנוע אפשרות של Activity Hijacking לוגי יש להשתמש ב-Explicit Intents בלבד.
- יש לקרוא ל-()getCallingPackage על מנת לוודא את שם ה-Package שהפעילה את ה-Activity.
- אין להשתמש ב-addJavascriptInterface .
- יש להשתמש ב-()clearCache על מנת לנקות כל קובץ אשר יכול היה להישמר על ידי האפליקציה.
- יש להגדיר רשימת עמודים המותרים לשימוש (Whitelist).

3.7.3. Content Providers

- בעת הגדרת רכיבי אחסון (Content Provider) יש צורך לבצע הפרדה בין הרשאות כתיבה לקריאה באופן רוחבי.

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

- יש לעשות שימוש ב- <grant-uri-permission> על מנת להגביל גישה ל-branch מסוים ברמת ה-DATA. בדוגמא הבאה מוצגת הגבלת גישה ל-messages בלבד:

```
<grant-uri-permission android:path="/messages/" />
```

- אין להשתמש בתכונות ה-pathPrefix או ה-pathPattern, אשר מגדילות את טווח הגישה המותר.
- ניתן לבצע הגדרה פרטנית על ידי שימוש ב-URI במידת הצורך. לדוגמא:

```
<provider

       android:authorities="SendMySMS" android:name="StatusProvider"

       android:readPermission="com.test.access.permission.READ_STATUS"

       android:writePermission="com.test.access.permission.WRITE_STATUS">

</provider>
```

- אין להגדיר את תכונת ה-grantUriPermissions כ-True מכיוון שמאפשרת לגשת לכל רכיב מידע תחת אותו provider.
- אין לשמור מידע רגיש ב-Persistent Content Provider.
- אין לתת הרשאות גישה לאפליקציות אחרות.
- יש לקרוא ל- getCallingUid() על מנת למנוע קריאה לא מורשית באמצעות וידוא ה-User ID של הקורא.

3.7.4. Broadcasts

התקשורת בין אפליקציות ב- Android מתבצעת באמצעות Broadcasts (תשדורות), כאשר ההודעות עצמן הן מסוג Intent. הצהרה על מחלקות שתפקידן לקבל Broadcasts מאפליקציות אחרות או מהמערכת מתבצעת באמצעות האלמנט <receiver> בקובץ AndroidManifest.xml. שימוש ב-<intent-filter> ובקטגוריות יכול להגביל את סוגי ה-Intents שנשלחים אל המחלקה, אולם אין להסתמך עליהם כמנגנון אבטחה, שכן ניתן לשלוח הודעה ישירות אל המחלקה תוך עקיפת הפילטר. כמו כן, הפילטר אינו משפיע על תוכן ההודעה, שיכול להיות זדוני.

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

- רכיב Broadcast אשר לא מגדיר הרשאות גישה ניתן לקריאה על ידי כל אפליקציה. על מנת להגביל את האפליקציות האחרות מומלץ להגדיר הרשאה פרטית. באופן כזה רק אפליקציה שמצהירה על שימוש בהרשאה הזו, תוכל לשלוח Broadcasts אל האפליקציה שלנו. בכדי להגדיר אילו Receivers יכולים לקבל את השדרים, יש לקבוע עבור כל התשדירים הרשאה אשר אותה יצטרכו להחזיק. לדוגמא:

```
Intent bdctIntent = new Intent(MESSAGE_RECEIVED);

myContext.sendBroadcast(bdctIntent,"com.example.test1.permission.MSG_NOTIFY_RECEIVE");
```

- בכיוון השני, Receivers אשר מקבלים תשדירים, צריכים לוודא כי השדר הגיע ממקור אמין ולכן יש להגדיר ב-AndroidManifest.xml כי השדרים יתקבלו ממקורות המכילים הרשאות מתאימות. לדוגמא, במקרה זה מוצג Receiver אשר מחכה לשדרים מסוג MESSAGE_RECIEVED שנשלחו ממקור בעל הרשאת MSG_NOTIFY_SEND:

```
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"package="com.example.test1">

    ...

    <receiver android:name=".UIMailBroadcastReceiver"

         android:permission="com.example.testapps.test1.permission.MSG_NOTIFY_SEND">

         <intent-filter>

                  <action android:name="com.example.test1.action.MESSAGE_RECEVIED">

          </intent-filet>

    </receiver>

     ...

</manifest>
```

- ניתן להגדיר Broadcast Receiver גם בקוד, ובמקרה זה יש להגדיר כמו כן הרשאות:

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

```
IntentFilter intentFilter = new IntentFilter(MESSAGE_RECEIVED);

UIMailBroadcastReceiver rcv = new UIMailBroadcastReceiver();

myContext.registerReceiver(rcv, intentFilter,

   "com.example.testapps.test1.permission.MSG_NOTIFY_SEND", null);
```

- יש לבצע בדיקת קלטים למידע שמתקבל ב- Broadcast Reciever.
- במידה ונשלח מהאפליקציה שלנו מידע רגיש בתור Broadcast, יש לוודא שרק נמענים מורשים יקבלו אותו. אפליקציות יכולות להירשם להאזנה ל-Intents מסוימים ללא צורך באישור, ולכן יש להשתמש בהרשאה פרטית כדי להגביל את הנמענים.

```
context.sendBroadcast(intent, "il.moh.android.DOWNLOAD_DATA");
```

- מערכת Android מאפשרת שליחת Sticky Broadcasts – הודעות שנשארות זמינות למשך זמן מסוים לאחר שליחתן (מטרתן בד"כ יידוע על שינוי במצב המערכת). מכיוון שהודעות כאלה לא ניתן לאבטח כמו Broadcasts רגילים, יש להימנע מחשיפת מידע רגיש באמצעותן.

3.7.5. WebView

המחלקה WebView מציגה אתר web-י על מסך המכשיר הנייד. כאשר Activity מציג WebView, האפליקציה יכולה לשלוח Intent עם URI לבחירה, ולהורות לאותו WebView להציג את דף האינטרנט הרצוי.

- יש לטעון רק כתובות url המתחילות ב-https בתוך ה-WebView.
- יש להסיר קוד המאפשר לאפליקציה אחרת במכשיר לטעון תוכן מסוים בתוך ה-WebView.

בברירת המחדל, מאופשר ל-WebView לגשת למשאבים פנימיים. ניתן להשתמש בהגדרות הבאות על מנת לחסום זאת:

- אם אין צורך לאפשר לקוד JavaScript להיות מורץ ב-WebView – יש לבטלו באמצעות:

```
webview.getSettings().setJavaScriptEnabled(false);
```

- למנוע מ-WebView להריץ תוספים חיצוניים:

```
webview.getSettings().setPluginState(PluginState.OFF);
```


Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

- יש להגביל את ה- WebView מלגשת ל-file system:

```
webview.getSettings().setAllowFileAccess(false);

webview.getSettings().setAllowFileAccessFromFileURLs(false);
```

- למניעת גישה לקובץ שנטען על ידי content provider:

```
webview.getSettings().setAllowContentAccess(false);
```

- באם קיים שימוש ב-WebView, אין להגדיר כ-true את:

```
setAllowUniversalAccessFromFileURLs
```

- יש לטעון ל-WebView תוכן שעבר בקרת קלט. ניתן לבצעה באופן הבא ע"י דריסת הפונקציה shouldInterceptRequest של WebViewClient:

```
@Override

public WebResourceResponse shouldInterceptRequest (final WebView view,

String url)

{

    Uri uri = Uri.parse(url);

    if (!uri.getHost.equals("www.health.gov.il") ||!uri.getScheme.equals("https"))

    {

        return new WebResourceResponse("text/html", "UTF-8",new
StringBufferInputStream("alert('Not happening')")

    }

    else

    {

        return super.shouldInterceptRequest(view, url);

    }

}
```

- רכיב WebView ב-Android (בגירסא 4.2.2 ומטה) עלול לאפשר הרצת קוד JavaScript. במיוחד בגירסא 4.1.2 הסיכון גדול למימוש Remote Code Execution. Android בגירסת API המוקדם מגירסא 17 אינו

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

מגביל את מתודת WebView.addJavascriptInterface, המאפשרת להזריק אובייקטי Java לתוך ה-WebView.

יש לוודא שרק משאבים מיועדים נטענים ל-WebView. אם ישנם ממשאבים נוספים, מומלץ לטעון אותם ע"י פתיחת Intent חדש ללא מתודת ה-JavaScriptInterface הנ"ל.

3.7.6. תקשורת עם Services

ה-Service הינו פעולה שרצה ברקע, באופן שאינו גלוי למשתמש, והוא יכול לקבל פקודות מ-Intents כתלות לפעולות שביצע המשתמש. רכיבים זדוניים יכולים להתחזות ל-Services לגיטימיים על מנת לגנוב את המידע שמועבר. לכן נדרש לוודא שאכן ה-Service אליו האפליקציה מדברת, הוא הנמען הרצוי. הבדיקה יכולה להתבצע בשני אופנים: ציון השם המלא של הרכיב בעת התחברות אל ה-Service, או בדיקת ההרשאה הפרטית הספציפית ב- ( )onServiceConnected. יתרונה של השיטה השנייה בכך שניתן יותר בקלות להחליף את ה- service provider, מכיוון שהבדיקה אינה תלויה בשם המדויק שלו.

על מנת להגביל גישה לשירותים רגישים יש צורך בבדיקת הרשאות בזמן ריצה של רכיב שירות (Service) רגיש, בכדי למנוע הפעלה מרמת הרשאה נמוכה, בדיקה זו נעשית על ידי קריאה למתודת checkCallingPermission():

```
private final IStatusTracker.Stub mBinder = new IStatusTracker.Stub() {

       public boolean isTracking() {return mTracking;}

       public boolean addNickname(String nick, int contactId) {

              if (StatusTracker.this.checkCallingPermission(PERMISSION_STATUS_SERVICE_ADD) !=
              PackageManager.PERMISSION_GRANTED) {

                     throw new SecurityException("Requires " + PERMISSION_FRIEND_SERVICE_ADD);

              }

          ...

        }

int canProcess = checkCallingPermission("com.example.test1.perm.READ_INCOMING_EMAIL");

if (canProcess != PERMISSION_GRANTED)

       throw new SecurityException();
```


Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

- בכל מקרה של שימוש בשירות יש להגדיר הרשאות מתאימות באמצעות android:permission על מנת למנוע הפעלה ישירה של service כגון ע"י שליחת intent או באמצעות interface binding.
- יש לקרוא ל- getCallingUid() על מנת לוודא את ה-User ID של הקורא על מנת למנוע קריאה לא מורשית.

3.8 הצגת האפליקציות שהיו בשימוש

כאשר משתמש לוחץ על כפתור ה-"בית" במכשיר, האפליקציה מתכווצת אך עדיין רצה מאחורי הקלעים. פעולת הכיווץ יוצרת תמונה ממוזערת המתארת את המצב האחרון שבה הייתה האפליקציה לפני לחיצה על כפתור הבית. ה-Android שומר בזיכרון המכשיר את אותן תמונות של האפליקציות שהיו פתוחות לאחרונה ושכעת הן רצות ברקע. במידה שאפליקציה ש"נסגרה" באופן חלקי במהלך צפיה במסך רגיש, עלול גורם זדוני לנצל חשיפה זו ע"י שליפת תמונת המצב האחרון של האפליקציה מן המכשיר, ועל ידי כך לדלות מידע רגיש אשר קיים באפליקציה.

- כדי להגן על אותם נתונים רגישים, יש להוסיף את הקוד הבא בפונקציית onCreate של ה-Activity:

```
getWindow().addFlags(WindowManager.LayoutParams.FLAG_SECURE);
```

- אפשרות נוספת – להוסיף לכל Activity רלוונטי בקובץ ה-Manifest של האפליקציה את המאפיין הבא:

```
android:excludeFromRecents=true
```

3.9 בדיקות קלט

נוהג אבטחת מידע מקובל באתרי אינטרנט, מערכות מידע ואפליקציות הינו לבצע בדיקות קלט מלאות וקפדניות לכל קלט המתקבל ממשתמשי הקצה. מטרת הבדיקות הללו הנה לוודא כי הקלט הינו חוקי ותקין, וכי אינו מכיל קוד זדוני אשר נועד לשרת גורמים זדוניים בניסיונם לתקוף את המערכת. מומלץ ליישם מנגנון בדיקות קלט בכל מקום בו מבוצע שימוש בקלט חיצוני, שיכלול את סוג הבדיקות הבאות:

- Constrain – הגבלה של קלטים מותרים בלבד באמצעות שימוש ב-White List. כלומר, יש לבדוק את הקלט עפ"י סוג הנתון (מספר, מחרוזת וכד'), גודל הנתון (טווח מספרים, אורך המחרוזת), בדיקת טווח התווים החוקיים (0-9, a-z, A-Z וכד'). לאחר בדיקה זו מתאפשר אך ורק קלט הנחשב כתקין. נחשב לשכבת ההגנה המועדפת.
- Reject – דחיית הקלט ע"י שימוש ב-Black List, כלומר חסימת קלט הידוע כמשמש למתקפות מוכרות. ככלל, אין להסתמך על שכבה זו לבדה, אלא ראוי לצרף אותה לזו הקודמת.

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

- Sanitize – 'טיהור' הקלט, הפיכת קלט שיש לו פוטנציאל להיות זדוני לקלט בטוח, כדוגמת קידוד (Encoding).

3.10. מניעת SQL Injection

- דוגמא לשימוש ב-Prepared Statement:

```
String[] userInput = new String[] {"book", "wiley"};
Cursor c = database.rawQuery("SELECT * FROM Products WHERE type=? AND brand=?", userInput);

//using the query() method
String[] userInput = new String[] {"book", "wiley"};
Cursor c = database.query("Products", null, "type=? AND brand=?",userInput, null, null, null);

//For actions other than querying, use the SQLiteStatement
SQLiteStatement statement = database.compileStatement("INSERT INTO
Products (type, brand) values (?, ?)");
statement.bindString(1, "book");
statement.bindString(1, "wiley");
statement.execute();
```

- דוגמא לשימוש ב-Parameterized:

```
NSString* safeInsert = @"INSERT INTO messages(uid, message, username) VALUES(?, ?, ?)";
if(sqlite3_prepare(database, [safeInsert UTF8String], -1, &statement, NULL) != SQLITE_OK)
{
       // Unable to prepare statement
}
if(sqlite3_bind_text(statement, 2, [status.message UTF8String], -1, SQLITE_TRANSIENT) !=
SQLITE_OK)
{
       // Unable to bind variabless
}
```

3.11. הגנה מפני Cross Site Scripting (XSS)

מתקפת XSS עלולה להתממש כאשר מידע שאיננו trusted נכנס לתוך UIWebView (היא גם תלות של אופן טעינת ה-WebView ורמת ההרשאות שיש לאפליקציה וכו').

- יש לבדוק את תקינות הפרמטרים אשר בשימוש לבניית התשובות הדינמיות, בין אם אותם פרמטרים מגיעים בצורה ישירה מהמשתמש, ובין אם הם עובדו לפני כן בחלקים אחרים של האפליקציה.
- יש לבדוק את תקינות הפרמטרים בצד השרת (אין להסתמך על בדיקות שנעשות בצד הלקוח).

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

- יש לאפשר אך ורק קלט המוגדר כחוקי.
- יש לבדוק כל פרמטר מבחינת קיום, סוג, אורך וטווח. יש לבצע בדיקה זו באופן מחמיר בה יוגדרו מהם התווים החוקיים (מתודולוגית White List), ולא מהם התווים הלא חוקיים (מתודולוגית Black List). שימוש בביטויים רגולאריים (regular expressions) יכול לסייע בבדיקות.
- יש לתעד כל בקשה אשר נדחית עקב פרמטרים לא תקינים, ובמקרה הצורך יש להעביר הודעה מתאימה למנהל המערכת.
- מנגנון Encoding יכול להקל במניעה של חשיפה זו, ולכן מומלץ לאוכפו באופן ריכוזי כך שלא תתאפשר הצגת פלטים ללא שימוש ב-Encoding. יש לבצע Encoding המתאים למיקום הפלט (למשל HTML Encoding עבור פלט HTML שמוצג ללקוח, URL Encoding עבור פלט שנכתב לתוך קישורים או פרמטרים העוברים ב-URL ו- JavaScript Encoding עבור פלט שנכתב אל תוך פונקציות JavaScript).
- ה-HTML Encoding צריך לנטר, לכל הפחות, את התווים הבאים: ' (גרש), " (מרכאות), > (קטן מ-), < (גדול מ-), & (אמפרסנד), / (לוכסן) ו-\ (לוכסן אחורי).

3.12. הגנה מפני התקפת TapJacking

התקפת TapJacking מבוססת על שימוש ב-Overlays. בזמן פעילות האפליקציה, ניתן להסתיר את האפליקציה באמצעות אפליקציה אחרת או תוכנה זדונית במטרה לגרום למשתמשים לגיטימיים לבצע פעולות זדוניות (ללא ידיעתו של המשתמש עצמו), כגון: שינוי סיסמא, העברת כספים לחשבון אחר, קניית ניירות ערך וכדומה. ראוי לציין, כי התקפת TapJacking גם יכולה לנצל את ה-Session הקיים של האפליקציה (כלומר לאחר תהליך הזדהות). ביצוע התקפת TapJacking אינה דורשת הרשאות Root (רמת הרשאות גבוהות ביותר) במכשיר. בשיטה זו גורם התוקף למשתמש לגיטימי לבצע פעולות זדוניות באפליקציה מבלי שיהיה מודע לכך שביצע אותן.

ניתן ליישם הגנה כנגד TapJacking בשתי דרכים:

- הגדרת ערך המשתנה filterTouchesWhenObscured כ-True עבור ה-Layouts הרגישים, או הגדרת ה-setFilterTouchesWhenObscured כ-true עבור ה-WebView הרלוונטי.
- מימוש פונקציית onFilterTouchEventForSecurity ע"י דריסתה ויישום מדיניות אבטחה חזקה.

3.13. שימוש במחולל מספרים אקראיים

לעיתים ישנו צורך להגריל מספרים בצורה אקראית לצורך שימוש בסיסמאות או מפתחות קריפטוגרפים. ישנם שני סוגים של מחוללי מספרים אקראיים (PRNG - Pseudo-Random Number Generators) – סטטיסטי וקריפטוגרפי:

- ב-PRNG הסטטיסטי עלולה להיות בעיה המאפשרת לחזות מראש מהו המספר שיחולל, כלומר הערכים המופקים באופן זה אינם אקראיים לחלוטין, וניתן לצפות אותם מראש (predictable).
- ה-PRNG הקריפטוגרפי מייצר מספר שמקשה יותר על חיזויו מראש.

בשימוש בפונקציות מן הסוג הראשון עלול גורם זדוני להצליח לחזות מהי הסיסמא שיוצרה ע"י השרת, ולהשתמש בה. לכן, עבור נתונים רגישים שנדרשים להיות מאובטחים, כגון: סיסמאות ומפתחות הצפנה, יש

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

להשתמש במחולל הידוע כקריפטוגרפי. ע"מ ליצור מפתח בצורה מאובטחת, יש להשתמש במחלקה SecureRandom:

```
SecureRandom sr = new SecureRandom();

KeyGenerator generator = KeyGenerator.getInstance("AES");

generator.init(256, sr);

SecretKey key = generator.generateKey();
```

דוגמא נוספת – ליצירת סיסמא רנדומלית (חזקה) בעלת 12 תווים:

```
SecureRandom randPasswd = new SecureRandom();

byte bytes[] = new byte[12];

randPasswd.nextBytes(bytes);
```

3.14. שיחרור אובייקטים מהזיכרון

מערכת עשויה להשתמש מספר פעמים באפשרות לפתיחת קבצים ו/או חיבור לבסיסי נתונים באמצעות פונקציות. פונקציות אלו פותחות את הקובץ/מתחברות לבסיס הנתונים, ומחזירות handler, שבעזרתו ניתן לבצע פעולות על אותם המשאבים. בכל פעם שמתבצעת קריאה למשאב חיצוני (כבסיס נתונים, קובץ וכיו"ב) ללא שחרורו לבסוף, גורם זדוני יכול לנצל חשיפה זו ולגרום למניעת שירות (Denial of Service) ע"י הצפת משאבי הזיכרון בשרת שבו נפתחים כל אותם הקבצים. על כן, בסוף השימוש באותו משאב שהוקצה יש לשחררו, ולהודיע למערכת ההפעלה שהסתיים השימוש באותו מצביע למשאב. לדוגמא:

```
private SQLiteDatabase db;

...

db.close();
```


Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

3.15. טיפול במידע רגיש

3.15.1. שמירת מידע רגיש

- בגרסאות השונות של Android, רוב התוכן נשמר על גבי כרטיס זיכרון ה-SD (Secure Digital) בצורה גלויה. כלומר, ניתן לחלץ את המידע הרגיש שנשמר על ידי האפליקציה באמצעות הוצאת כרטיס הזיכרון מהמכשיר, תוך עקיפת מנגנוני האבטחה של מערכת ההפעלה (נעילת המכשיר) והאפליקציה עצמה (סיסמת גישה, למשל).
- על כן, מומלץ לא לשמור מידע רגיש ופרטי על גבי המכשיר הנייד, אלא רק בשרת האפליקציה. את המידע הנשמר על המכשיר עצמו יש להצפין על ידי אלגוריתם סימטרי חזק, כגון: AES 256-bit, כאשר על מפתח ההצפנה יש להגן עם סיסמת משתמש חזקה או מפתח אסימטרי (RSA 2048-bit). עם זאת, יש לזכור כי במקרה והמידע נגנב, תוקף יוכל לבצע התקפת Brute Force במצב offline בניסיון לפרוץ את ההצפנה, כלומר באופן לא מוגבל.
- עבור יצירת cryptographic hash יש להימנע משימוש באלגוריתמי הצפנה כגון: SHA-1 או MD5 (שנחשבים למיושנים וחלשים מבחינת אבטחה), ולהשתמש ב- SHA 256 ומעלה.
- אין לשמור מידע רגיש על המכשיר בקבצים לרבות סיסמאות, מפתחות הצפנה וכדומה.
- החל מ-API גירסא 18, מערכת Android Keystore מאפשרת לאחסן מפתחות הצפנה בתוך container כדי להקשות על החילוץ שלהם מפני גורם זדוני. המפתחות אינם ניתנים לייצוא אך כמובן שניתן להשתמש בהם לצרכי הצפנה. לכן, מומלץ לשמור את מפתחות ההצפנה וכיו"ב ב-Keystore.
- כמו כן, אין לשמור מידע רגיש בקוד (hardcoded), וזאת משום שניתן בקלות לבצע decompile לאפליקציית צד הלקוח (על קובץ ה-jar), ולכן יש לצאת מנקודת הנחה שקוד המקור קריא לחלוטין.
- ההרשאה המותרת ליצירת קבצים הינה MODE_PRIVATE בלבד. אין לעשות שימוש בהגדרות אחרות אשר מאפשרות לאפליקציות אחרות לגשת לתוכן הקבצים.
- יש לשמור את קבצי האפליקציה ב-internal storage בלבד, בתיקיית האפליקציה.

3.15.2. מנגנון תיקון שגיאות בשדות עם מידע רגיש

לעיתים המשתמש נדרש להזין מידע רגיש לאפליקציה, כגון: מספר כרטיס אשראי, בעוד שמטעמי נוחות המשתמש, שדה קלט עבור מידע זה בד"כ לא יהיה ממוסך. עבור שדות קלט שאינם ממוסכים, Android מספקת יכולת המובנית במכשיר ברמת מערכת ההפעלה של תיקון אוטומטי של שגיאות כתיב (מנגנון הנקרא Autocorrect). יכולת זו ממומשת על ידי ניתוח אוטומטי של תוכן שדות הקלט ושמירתו במנגנון system cache. בעוד שמנגנון זה עוזר למשתמש, הוא נחשב למסוכן משום שהמידע נשמר בו בצורה בלתי מאובטחת, וזאת כחלק ממנגנון הלמידה העצמית שלו. במידה וגורם זדוני יצליח להשיג גישה למכשיר (או לחלופין ע"י אפליקציה זדונית המותקנת על המכשיר), ניתן יהיה לקרוא את המידע שנשמר באותו מילון של autocorrect, לרבות פרטי המשתמש הרגישים.

על מנת למנוע ממכשיר ה-Android את שמירת הערכים הרגישים, יש להפעיל את האופציה NoSuggestions עבור השדות הרלוונטיים בממשק האפליקציה (באחת משתי הדרכים הבאות):

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

כ- Attribute ב- XML:

```
android:inputType="textNoSuggestions|textVisiblePassword"
```

בקוד האפליקציה:

```
mEditText.setInputType(InputType.TYPE_TEXT_FLAG_NO_SUGGESTIONS |
                       InputType.TYPE_TEXT_VARIATION_VISIBLE_PASSWORD);
```

3.15.3. מיסוך מידע רגיש ב-GUI

כאשר נדרש להציג למשתמש מידע רגיש, אין להשתמש בערך המלא שלו כ- clear text. הצגה כזו חושפת את המידע להתקפות כגון "shoulder surfing" (הצצה מעבר לכתף), ושמירה שלו שלא בכוונה בתמונות המסך אותן יוצרת Android באופן אוטומטי (או אף בתמונת מסך שהמשתמש עצמו לוקח). על כן יש להציג תמיד מידע רגיש באופן ממוסך, אם על ידי שימוש בשדות Password ואם על ידי החלפת חלק מהערך בכוכביות. לדוגמא, סטנדרט ה-PCI קובע שבעת הצגת נתוני הכרטיס, נדרש להסתיר חלק מהספרות של מספר כרטיס האשראי בכוכביות.

יש למסך את תווי הסיסמא שמזין המשתמש באמצעות הקוד הבא (עבור שדה EditText):

```
android:inputType="textPassword”
```

3.15.4. איפוס מידע ב-RAM

זיכרון העבודה של האפליקציה עלול להישמר ב-memory snapshot של Android או בקובץ memory dump במקרה והאפליקציה קורסת. ערכים רגישים שהאפליקציה שומרת בזיכרון העבודה, כגון סיסמת המשתמש, ייכתבו אז כ-clear text באופן לא מאובטח. על כן יש לאפס את המשתנים ששומרים ערכים רגישים מיד כאשר כבר אין בהם צורך. למשל, לאחר ביצוע הזדהות מוצלחת, האפליקציה אינה זקוקה יותר לערך ה-clear text של סיסמת המשתמש. יש לבצע את האיפוס גם למשתנים פנימיים של האפליקציה וגם למשתנים של שדות קלט ב-GUI.

3.15.5. מידע גיאוגרפי

על מנת לשמור על פרטיותו של המשתמש ועל מנת לא לחשוף מיקומים רגישים, יש להימנע מלשמור מידע גיאוגרפי. לעיתים מיקום GPS נשמר באופן אוטומטי (במצלמה לדוגמא) ולכן נרצה להסיר מידע זה; למשל, עבור תמונות אשר צולמו, יש לשנות או לאפס את מידע ה-EXIF אשר נכלל בתמונה. המיקום הגיאוגרפי לעיתים גם מסונכרן ומדווח לשרתים של גוגל, ולכן יש להימנע משימוש ב-GPS כאשר ניתן.

3.16. שימוש בהגדרות מאובטחות של ה-Cookie

בעת תהליך ההתקשרות מול המערכת נוצר Session ID פרטי למשתמש התקף כל עוד המשתמש איננו מתנתק מהאתר. ה- Session ID מזהה את המשתמש מול האתר, ועל כן כל זמן שהוא תקף כל משתמש יכול להשתמש במזהה שיחה זה על מנת להזדהות מול האתר. נתוני ה-Session נשמרים בצד הלקוח

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

באמצעות Cookies אשר מועברים בכל בקשה מהלקוח אל השרת. בשל כך, יש להגן על נתונים אלה כדי למנוע מצבים של גניבת זהות.

- מאפיין ה-HTTPOnly נועד כדי למנוע גישה לערך ה-Session ע"י סקריפטים שונים, כגון JavaScript.
- מאפיין ה-Secure של ה-cookie נועד כדי למנוע העברה של הערך כאשר התקשורת מתבצעת בצורה לא מוצפנת (בתווך HTTP).
- מאפיין ה-MaxAge קובע את משך החיים של ה-Cookie עד שתפוג ותימחק. יש להגדיר פרק זמן סביר וקצר במידת האפשר, כדי לא להגדיל את משטח התקיפה.
- מאפיין ה-Domain קובע את דומיין ה-URL שרק דפים שככתובותיהם כלולים תחתיו יוכלו לגשת אל ה-Cookie שנשלחים ע"י הדפדפנים. במידת האפשר, צריך להכיל את הדומיין הנדרש בלבד.
- מאפיין ה-Path מאפשר להפריד בין Cookies הנשלחים לאתר הציבורי לבין ה-Cookies הנשלחים אל ממשק ה-admin. מומלץ שלא להגדירו כנתיב של ה-root.

```
HttpCookie myHttpCookie = new HttpCookie("LastVisit", DateTime.Now.ToString());

myHttpOnlyCookie.setHttpOnly(true);

myHttpOnlyCookie.setSecure(true);

myHttpOnlyCookie.setMaxAge(86400);

myHttpOnlyCookie.setDomain("health.gov.il");

myHttpOnlyCookie.setPath("/root/app/web");
```

3.17. מצב Debug

- במהלך תהליכי הפיתוח נוהג מקובל הוא לעבוד ולקמפל את המערכת במצב Debug על מנת שמתכנתי המערכת יוכלו לבדוק את המערכת. יחד עם זאת, במסגרת תהליכי העברת המערכת לסביבת הייצור, יש לקיים מספר פעולות אשר יאפשרו את העברת המערכת באופן מאובטח לסביבה הייצורית (Production), כולל הסרת כל הגדרות ה- Debug מהקוד:

```
<application android:debuggable="false“ …>
```

יצויין שברירת המחדל של הגדרה זו היא false, אולם מצוי שבשלבי הפיתוח וה-Testing צוותי הפיתוח מעדיף לשנותה ל-true.

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

3.18. מתן הרשאות עודפות לאפליקציה

- אפליקציות המותקנות על מכשירי Android דורשות הרשאות שונות על מנת לקבל גישה למידע וליכולות של המכשיר. מתן גישה להרשאות האפליקציה נחוץ בד"כ לפעילות תקינה של האפליקציה. עם זאת, מתן הרשאות עודפות לאפליקציה עלול להעמיד את מכשיר הלקוח בסיכון וזאת משום שהאפליקציה עלולה לגשת למידע רגיש על המכשיר שאין לה צורך בו. הרשאות שאינן נצרכות לתפקודה התקין של האפליקציה – יש להסירן מן הקובץ AndroidManifest.xml (הן ברמת האבטחה, והן בפן העסקי – אפליקציה הדורשת הרשאות שאין לה בה צורך תגרום לכך שפחות משתמשים יהיו מעוניינים להתקין אותה). דוגמא להרשאות כאלו אותן האפליקציה מבקשת לקבל:

```
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />

<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />

<uses-permission android:name="android.permission.INTERNET" />
```

- בין ההרשאות הנחשבות למסוכנות נמצאת הרשאת GET_ACCOUNT. במקרה של תוכנה זדונית המותקנת על מכשיר Rooted, ניתן יהיה לגשת באמצעותה לנתונים רגישים השמורים במכשיר.

3.19. חשיפת פרטי המפתח (Developer) בקבצי האפליקציה

אפליקציות הנכתבות למערכת הפעלה מסוג Android חייבות להיות חתומות (signed) באמצעות תעודה הדיגיטלית. החתימה נועדה להבטיח את זהות האפליקציה, וכן להבטיח כי גורם זדוני לא יצליח להתקין אפליקציה זדונית במקום האפליקציה הנוכחית (בהנחה שהגדרות המכשיר לא מאפשרות התקנת אפליקציות שהן untrusted).

עם זאת, השימוש בתעודה דיגיטלית עלול לחשוף פרטים אודות הארגון, וזאת כתלות ב-Meta data השמור בתעודה הדיגיטלית. בין היתר, יש לשים לב שהתעודה הדיגיטלית לא תחשוף פרטים של מפתחי האפליקציה, וזאת מכיוון שגורם זדוני עלול לנצל זאת למתקפות Phishing ו/או Social Engineering. לכן, מומלץ להסיר את פרטי המפתח מקבצי החתימה, ולהשתמש במידע גנרי בלבד.

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

4. דגשים לפיתוח מאובטח ב-iOS

4.1 עבודה בטוחה עם קבצים זמניים ו- cache

- אחת הדרכים של מערכת הפעלה להעביר נתונים בין תוכניות הינה באמצעות יצירת קובץ זמני. קבצים זמניים נשמרים בדר"כ בספרייה ייעודית. כדי למנוע מגורם זדוני לגשת לקבצים הללו שנוצרים, מומלץ לייצר קבצים עם שמות אקראיים וקשים לצפייה מראש, כדי שהסבירות לגישה אליהם תקטן. יתרון נוסף לשימוש בפונקצות בטוחות הוא הימנעות מ-Race Condition, שבו נמנע מצב של פתיחת קובץ קיים ועריכה שלו. בעת יצירת קבצים זמניים יש להשתמש במתודות mktemp() ו-mkstemp(), שמייצרות קובץ חדש בעל שם ייחודי, ורק לאחר מכן התכנית פותחת אותו.
- בנוסף, יש ליצור את הקבצים הזמניים באיזור מוגן ע"י שימוש במודל ה-sandbox של המכשיר באמצעות NSTemporaryDirectory(), כך שקבצים זמניים יהיו נגישים לאפליקציה בלבד.
- אין לשמור מידע רגיש כגון סיסמאות ומפתחות צופן ב- NSUserDefaults.
- יש להימנע מ-http caching על ידי מימוש של NSURLConnection וביטול (disabling) של newCachedResponse:

```
NSCachedURLResponse *newCachedResponse = cachedResponse;
if ([[[[cachedResponse response] URL] scheme] isEqual:@"https"])
{
   newCachedResponse = nil;
}
return newCachedResponse;
```

- במערכת ההפעלה iOS, קיים מנגנון cache למקלדת, הנועד להקל על המשתמש בעת הכנסת מילים שאינן במילון, אבל עדיין בשימוש על ידי המשתמש. המנגנון שומר את הערך שהכניס המשתמש (כדוגמת שם המשתמש או דואר אלקטרוני), כך שבפעם הבאה שהמשתמש יקליד את האותיות הראשונות של אותו הערך, יציע המנגנון מספר ערכים אפשריים. בעוד שמנגנון זה מקל על השימוש במכשיר הנייד, השימוש במנגנון זה עלול להיות מסוכן במידה שהערכים השמורים בו הינם רגישים. אי לכך, יש לכבות את אפשרות האיות האוטומטי עבור שדות כ-UITextView ו-UITextField המכילים נתונים רגישים:

```
UITextField* userPassword = [[UITextFieldalloc] init];
userPassword.autocorrectionType = UITextAutocorrectionTypeNo;
```


Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

4.2 בדיקת קלט

יש ליישם מנגנון בדיקת קלט בכל מקום בו מבוצע שימוש בקלט חיצוני, שיישם את סוג הבדיקות הבאות:

- Constrain – הגבלה של קלטים מותרים בלבד באמצעות שימוש ב-White List. כלומר, יש לבדוק את הקלט עפ"י סוג הנתון (מספר, מחרוזת וכד'), גודל הנתון (טווח מספרים, אורך המחרוזת), בדיקת טווח התווים החוקיים (0-9, a-z, A-Z וכד'). לאחר בדיקה זו מתאפשר אך ורק קלט הנחשב כתקין. נחשב לשכבת ההגנה המועדפת.
- Reject – דחיית הקלט ע"י שימוש ב-Black List, כלומר חסימת קלט הידוע כמשמש למתקפות מוכרות. ככלל, אין להסתמך על שכבה זו לבדה, אלא ראוי לצרף אותה לזו הקודמת.
- Sanitize – 'טיהור' הקלט, הפיכת קלט שיש לו פוטנציאל להיות זדוני לקלט בטוח, כדוגמת קידוד (Encoding).

4.3 SQL Injection

בעת כתיבה לבסיס הנתונים המקומי (SQLite) יש להימנע מכתיבת שאילתות דינמיות אשר עלולות להוביל להתקפות SQL Injection.

להלן שתי דרכים המסייעות להתמודד מול פגיעות ה-SQL Injection:

```
    1)  Parameterized:


NSString* safeInsert = @"INSERT INTO messages(uid, message, username) VALUES(?, ?, ?)";

if (sqlite3_prepare(database, [safeInsert UTF8String], -1, &statement, NULL) != SQLITE_OK)

{

       // Unable to prepare statement

}

if (sqlite3_bind_text(statement, 2, [status.message UTF8String], -1,SQLITE_TRANSIENT) != SQLITE_OK)

{

       // Unable to bind variabless

}
```


Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

```
    2)  Prepared Statements:


const char *sql = "SELECT username FROM users where uid = ?";

sqlite3_prepare_v2(db, sql, -1, &selectUid, NULL);

sqlite3_bind_int(selectUid, 1, uid);

int status = sqlite3_step(selectUid);
```

4.4 Cross Site Scripting (XSS)

- יש להימנע מלהשתמש בפרוטוקול file:// .
- יש לבצע Input Validation על הקלטים שנכנסים לאפליקציה (ראה הדגשים מסעיף 4.2).
- כמו כן, יש לוודא שפלט ה-HTML מקודד היטב לפני הצגתו ב-UIWebView .

4.5 הרצת אפליקציות תוך מתן מינימום הרשאות

- יש להימנע משימוש ב-kernel extensions מכיוון שתוקף יוכל לנצל פגיעויות שונות וליצור מצב בו תוכנה זו מתפקדת כרוגלה זדונית בעלת הרשאות מקסימליות (יש לקרוא תיעוד של חברת אפל בנושא).
- בעת קריאת נתוני מיקום גאוגרפיים (לדוגמא ע"י CLLocationManager), יש להשתמש ברזולוציה דיוק הנמוכה ביותר האפשרית. לדוגמא על מנת לקבוע את העיר בה נמצא המשתמש די להשתמש באפשרויות כגון:
  - kCLLocationAccuracyKilometer
  - kCLLocationAccuracyThreeKilometers

4.6 ניהול זיכרון

- ישנן פונקציות שתפקידן להקצות זיכרון לאובייקטים באופן דינמי. בשפת Objective-C למשל, נהוג לנהל ידנית את הזיכרון, ולשחרר במפורש זיכרון שהוקצה בצורה דינאמית על מנת לשחרר את משאבי המערכת ולנצלם מחדש. תוקף או גורם זדוני הבקיא בקוד המערכת, יכול לייצר שגיאות אשר יגרמו להקצאה בלתי נגמרת של זיכרון, מה שעלול בסופו של דבר למימוש Denial of Service (מתקפת מניעת שירות) ולהביא לקריסת האפליקציה. לכן, יש לשחרר כל אובייקט שהוקצה על ידי הפונקציה alloc בעזרת הפונקציה release.

להלן דוגמא של שיחרור המשתנה myVar מן הזיכרון בסוף השימוש בו (בפונקציה dealloc):

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

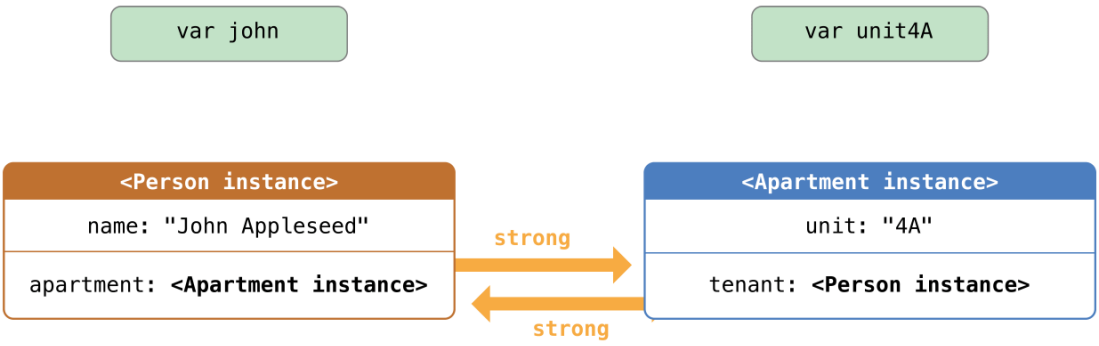
```
- (void)init
{
    myVar = [NSString alloc] init];
    ...
}

- (void)dealloc
{
    [myVar release];
    [otherVar release];
    [super dealloc];
}
```

- ב-Swift ישנו מנגנון לניהול זיכרון הנקרא ARC (Automatic Reference Counting), המכיל גם פונקציונליות של Garbage Collector. ה-ARC קיים החל מגירסת iOS SDK 5.0, והוא מפנה את המתכנת מלעסוק בנושא ניהול הזיכרון. מבחינת אבטחת מידע, ישנו יתרון ל-ARC בכך שהוא עשוי למנוע בעיות (Use After Free, Double Free) שנבעו בעבר מחוסר תשומת לב של המפתחים לנושאים הללו.

מנגנון זה מצמיד לכל אובייקט שהוקצה על המחסנית מונה מצביעים (references), וזה עוקב אחר כמות היישויות בתוכנית אשר מפנות אל האובייקט. כאשר מונה זה מתאפס, אין צורך יותר באובייקט הזה, והוא משוחרר מהזיכרון. אולם, עדיין ייתכן מצב בו האובייקטים לא ישתחררו מן הזיכרון, מכיוון שקיימים מצביעים פנימיים לאובייקט אחר המוגדרים כ-Strong References (זהו מצב ברירת המחדל).

בדוגמא הבאה, גם לאחר שיחרור האובייקטים john ו-unit4A, האובייקטים המוצבעים ישארו בזיכרון מכיוון שקיימים להם שדות פנימיים המצביעים אחד על השני:



בכדי לשחרר בצורה מיטבית אובייקטים – יש להבחין בין שני מצבים אפשריים:

- במקרה בו לשדה באובייקט המשוחרר יש זמן חיים קצר מלשדה באובייקט המצביע עליו, יש להגדיר את המשתנה כ-weak:

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

```
weak var tenant Person?
```

במקרה זה, לכל אובייקט יש את היכולת להיות בעל ערך nil.

- במקרה בו לשדה באובייקט המשוחרר יש זמן חיים שווה או ארוך מלשדה באובייקט עליו הוא מצביע, יש להגדיר את המשתנה כ-unowned:

```
unowned let tenant: Person
```

במקרה זה, האובייקט המוצבע חייב להיות בעל ערך בכל זמן נתון, ואינו יכול להיות בעל ערך nil.

- אין להשתמש ביכולת ה-unsafe של מצביעים מסוג unowned.
- יש להשתמש ב- Address Sanitizer וב- Undefined Behavior Sanitizer על מנת לזהות דליפות זיכרון ו-Overflows פוטנציאליים.
- Refer הינו מנגנון המגדיר פונקציות שריצתן תתבצע בסוף ריצת הפונקציה הנוכחית. יש להשתמש בו על מנת לבצע פעולות אשר חיוני כי יבוצעו תחת המתודה הנוכחית, למשל: סגירת קובץ לאחר פתיחתו.
- יש לשחרר משאבים שהוקצו במהלך התוכנית באופן מפורש;
  - עבור אובייקטים מסוג NSStreams שהוקצו באפליקציה – יש להפעיל את המתודה close().
  - עבור אובייקטים מסוג CFStreams שהוקצו באפליקציה – יש להפעיל את המתודה CFReadStreamClose().

לאחר מכן יש לשחרר את המצביע לאותו ה-stream באמצעות קריאה לפונקציה CFRelease, או לבצע השמה ל-NULL עבור האובייקט:

```
CFReadStreamClose(readStream);
CFRelease(readStream);
readStream = NULL
```

4.7. התקנה והרצה מאובטחות של האפליקציה

- שלב זה הינו חשוב ומשמעותי בתהליך הפיתוח וההטעמה של האפליקציה. הרבה מאוד פגיעויות נוצרות בתהליך זה, כאשר האפליקציה אינה מותקנת ו/או מקונפגת בצורה נכונה.
- יש לוודא שלא להתקין רכיבי תוכנה וקוד תחת הספריות הבאות:
  - /Library/StartupItems

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

- /System/Library/Extensions

קוד אשר ממוקם בספריות אלה רץ עם הרשאות root. כמו כן, דגש זה הינו חשוב בהמשך למה שכתבנו על אי-שימוש ב- kernel extensions.

- טען / הרץ ספריות וקוד מתיקיות ומקורות הנחשבים למאובטחים בלבד, ספריות אשר אוכפות מנגנוני גישה ואשר מנוטרות.
  - תיקיה אשר מאפשרת למשתמש רגיל לכתוב לתוכה (writeable), איננה נחשבת למאובטחת.
- יש להשתמש בפונקציות ומבנים המיועדים ל-Swift ולהימנע ככל האפשר משימוש בפונקציות ומבנים אשר מיועדות ל-Objective-C אשר יחשפו את האפליקציה לפגיעויות רבות הקיימות ב-Objective-C ואשר נחסמו ב-Swift. לדוגמא, שימוש או ירושה מ-NSObject יחשפו את האפליקציה לפגיעויות זמן ריצה הקיימות ב-Objective-C. יש לשים לב כי ב-Swift4 מתודות בספריות היורשות מספריות הכתובות ב-Objective-C חשופות לפגיעויות זמן ריצה במקרה בו המתודות או הספריות עצמן מסומנות עם @attribute מסויים (כגון @objc).

4.8. שימוש בהצפנה חזקה

- לעיתים נדרש לבצע Hash על נתונים על מנת שיהיו בלתי קריאים לגורמים שאינם רשאים לקרוא אותם. ישנם אלגוריתמי hashing הנחשבים כיום חלשים, כדוגמת MD5 ו-SHA1. בדוגמא הבאה מוצג שימוש באלגוריתם hash חזק:

```
NSData *imageData = [NSData dataWithContentsOfFile:file];
CC_SHA256 (imageData, [imageData length], result);
```

- iOS Keychain הינו כלי לניהול סיסמאות של מערכת ההפעלה. מטרתו היא שמירת מידע רגיש (כפרטי הזדהות, תעודות ומפתחות הצפנה) באופן מוצפן. ישנן מס' רמות של גישה ל- Keychain שניתן להגדיר, כגון: ה-keychain נגיש תמיד, הוא נגיש רק כשהמכשיר איננו נעול וכו'.

בעת שמירת מידע ל-keychain, יש להצפין אותו על ידי מימוש של הצפנה מותאמת (CCCrypt) ולהגביל את הגישה אליו לאפליקציה בלבד, ולא לאפשר לאפליקציות נוספות גישה (למעט מקרים בה נדרשת גישה מפורשת).

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

כעיקרון, מומלץ לשמור את מפתח ההצפנה ב-keychain. עם זאת, יש לתת את הדעת על כך שישנם מצבים בהם ה-keychain עלול להיפרץ[3], ואז אותו מידע רגיש יוכל להיות מפוענח. עבור מכשירים פרוצים (jailbroken), ה-keychain אינו מספק הגנה נוספת שכן למשתמש/תוקף ישנה גישה למפתחות ההצפנה. לכן, עבור מידע רגיש, נמנע משימוש ב- keychain. במקום זאת, נדרוש ממשתמש הקצה להקיש סיסמא אפליקטיבית כאשר האפליקציה מתחילה לפעול ו/או נגן באמצעות מפתחות הצפנה הנשלחים (בצורה מאובטחת) משרת המערכת לאחר שמשתמש מזדהה לאפליקציה ובשלבים שונים בהם נדרש להגן על מידע.

4.9 ביטול אפשרות copy-paste עבור נתונים רגישים

יש לבטל תכונה זו במידת הצורך, במיוחד במסכים בהם מוצג מידע רגיש.

- ניתן למחוק את ה- pasteboard בעבודה עם applicationWillTerminate, ע"י ההגדרה הבאה: pasteBoard.items = nil.
- אפשרות נוספת:

```
-(BOOL)canPerformAction:(SEL)action withSender:(id)sender {

  UIMenuController *menu = [UIMenuController sharedMenuController];

  if (menu) {

       menu.menuVisible = NO;

  }

  return NO;

}
```

4.10 יצירת ערכים אקראיים (Random)

- יש להימנע משימוש בפונקציות כגון rand(), random() כשמעוניינים לייצר מידע רגיש (כמו מנגנון ליצירת סיסמא אקראית). במקום זאת, יש לעשות שימוש בפונקציה כגון: SecRandomCopyBytes או ספריות חיצוניות מוכרות ובדוקות.

דוגמא לעבודה עם הפונקציה הנ"ל:

[3] קיימים מספר כלים ומערכות אשר מטרתן לפרוץ את ה- keychain (למשל, ע"י הכלי Keychain Dumper), אם כאשר נשמר מקומית על גבי המכשיר ואם כאשר נשמר כגיבוי ב- iTunes.

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

```
int result = SecRandomCopyBytes(kSecRandomDefault, sizeof(int),

   (uint8_t*)& randomResult);
```

4.11. מנגנון איות (תיקון שגיאות) בשדות עם מידע רגיש

ב-iOS קיים מנגנון בשם autosuggest אשר נועד להקל על המשתמש באמצעות תיקון של שגיאות כתיב נפוצות. בעוד שמנגנון זה עוזר למשתמש, הוא נחשב למסוכן משום שהמידע נשמר בו בצורה בלתי מאובטחת, וזאת כחלק ממנגנון הלמידה העצמית שלו. במידה וגורם זדוני יצליח להשיג גישה למכשיר (או לחלופין ע"י אפליקציה זדונית המותקנת על המכשיר), ניתן יהיה לקרוא את המידע שנשמר ב-autosuggest, לרבות פרטי המשתמש עבור ההתחברות לאפליקציה.

לפיכך, מומלץ לבטל את מנגנון ה-Autosuggest באמצעות הוספת הערך Autocomplete=Off לשדות מסוג UITextField.

4.12. תעבורת תקשורת מאובטחת

- ע"מ לשלוח לשרת בקשה המכילה מידע רגיש, יש להשתמש בפרוטוקול HTTPS:

```
NSString * const USER_URL = @"https://www.health.gov.il";

NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:[NSURL
URLWithString:USER_URL]];
[[NSURLConnection alloc] initWithRequest:request delegate:self];
```

- פרוטוקול HTTP משמש לתעבורת מידע בין משתמש הקצה והשרת. במתודת POST המידע יכול לעבור כחלק מגוף הבקשה לשרת, וכך הוא מוסתר מהמשתמש. ואילו במתודת GET המידע מועבר תמיד כחלק מה-URL. מתודת GET נחשבת לפחות מאובטחת מכיוון שגישות ל-URL-ים נשמרות לעיתים תכופות בהיסטורית הגלישה, בזיכרון המטמון בקבצי הדפדפן של המשתמש ובמנגנוני תיעוד שונים (כגון לוגים של proxy ולוגים של שרת האפליקציה).אין לשלוח מידע רגיש באמצעות מתודת HTTP GET, אלא ע"י HTTP POST:

```
NSString * const USER_URL = @" https://www.health.gov.il/login/user";
NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:[NSURL
```


Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

```
  URLWithString:USER_URL]];
[request setHTTPMethod:@"POST"];
```

- ATS (App Transport Security) הינו מנגנון אבטחה הזמין החל מגירסת iOS 9 ואילך, האוכף תעבורה מוצפנת בין האפליקציה לצד השרת (HTTPS + TLS1.2). שימוש בו עשוי לסייע במניעת מתקפת Man in The Middle (MiTM) ומתקפות רשת נוספות. יש לוודא כי הגנת ברירת המחדל הנ"ל מופעלת[4], והערך NSAllowsArbitraryLoads בקובץ Info.plist לא מוגדר כ-True:

```
<key>NSAppTransportSecurity</key>

<dict>

  <key>NSAllowsArbitraryLoads</key>

  <false/>

</dict>
```

- אין לקבוע Exceptions לתעבורה מוצפנת בקובץ Info.plist
- יש לאמת Certificate על ידי המספר סידורי שלו או חתימתו בטרם החלת ההתקשרות.
- בכל מקרה בו מועברים פרטים רגישים ברשת (כגון סיסמאות, פרטי לקוח וכדומה), יש להשתמש בהצפנה. הצפנה יכולה להיות מבוססת על רמת תווך התקשורת (כגון SSL) או ברמת המסר (מחרוזת מוצפנת).
- בעת שימוש ב-SSL ע"י NSURLSession, יש לוודא שה-certificate שהתקבל מהשרת הינו תקין.
- לעתים בשלבי הפיתוח משתמשים המתכנתים בתעודה שיוצרה באופן פנימי באירגון לצרכי נוחות (self-signed certificate). בטרם העליה לייצור עלול נושא זה להישכח, מה שעלול להוביל למתקפת Man in The Middle. על כן, יש לוודא כי ערך allowsAnyHTTPSCertificateForHost לא נקבע כ-True:

```
implementation NSURLRequest (IgnoreSSL)
```

[4] ה-ATS מאופשר כברירת מחדל כאשר משתמשים ב-class-ים מסוג NSURLSession , NSURLConnection ו-CFURL.

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

```
+ (BOOL)allowsAnyHTTPSCertificateForHost:(NSString *)host
{
    return NO;
}

@end
```

- אין לקבוע כ-true את הערך של setAllowsAnyHTTPSCertificate.
- בעת השימוש באוביקט מסוג NSStream, מומלץ להגדיר את ערכו של NSStreamSocketSecurityLevel ל- NSStreamSocketSecurityLevelTLSv1.

4.13. שימוש בהגדרות מאובטחות של ה-Cookie

4.13.1. Secure

בעת תהליך ההתקשרות מול המערכת נוצר Session ID פרטי למשתמש התקף כל עוד המשתמש איננו מתנתק מהאתר. ה- Session ID מזהה את המשתמש מול האתר, ועל כן כל זמן שהוא תקף כל משתמש יכול להשתמש במזהה שיחה זה על מנת להזדהות מול האתר. נתוני ה-Session נשמרים בצד הלקוח באמצעות Cookies אשר מועברים בכל בקשה מהלקוח אל השרת. בשל כך, יש להגן על נתונים אלה כדי למנוע מצבים של גניבת זהות.

מאפיין ה-Secure של ה-cookie נועד כדי למנוע העברה של הערך כאשר התקשורת מתבצעת בצורה לא מוצפנת (בתווך HTTP).

יש להגדיר את המאפיין של NSHTTPCookieSecure כ-True בכדי למנוע שליחה של ה-cookies בפרוטוקול HTTP:

```
NSDictionary *cookieProperties = [NSDictionary dictionary];
 ...
 [cookieProperties setValue:@"TRUE" forKey:NSHTTPCookieSecure];
 ...
 NSHTTPCookie *cookie = [NSHTTPCookie cookieWithProperties:cookieProperties];
 ...
```

4.13.2. Persistent

כאשר לא הוגדר זמן פקיעה מוגדר של ה-Session (Session Timeout), במידה ומשתמש לא בחר להתנתק באופן יזום, המערכת אינה מנתקת אותו גם לאחר זמן ארוך של אי-פעילות באתר. באותו פרק הזמן תאופשר שליחת בקשות לאתר בתור המשתמש, תוך שימוש באותו מאפיין שהוקצה עבורו ושמגדיר

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

אותו כמחובר למערכת. הגדרה כזו מעלה את פרק הזמן במהלכו ה-Token נשאר תקף, וממילא מגדילה את חלון ההזדמנויות של התוקף לנחש את מזהה השיחה.

בדוגמא הבאה מוצג כיצד ניתן לקצוב את תפוגת ה-Session ל-7 דקות:

```
NSDictionary *cookieProperties = [NSDictionary dictionary];
  ...
  [cookieProperties setValue:[[NSDate date] dateByAddingTimeInterval:(60*7)]
forKey:NSHTTPCookieExpires];
  ...
  NSHTTPCookie *cookie = [NSHTTPCookie cookieWithProperties:cookieProperties];
  ...
```

יתר ההגדרות הוסברו לעיל (3.16), ניתן להגדירן בצורה הבאה:

```
▪ Set-Cookie: id=a3fWa; Expires=Wed, 21 Oct 2015 07:28:00 GMT; Secure; HttpOnly
▪ Set-Cookie: qwerty=219ffwef9w0f; Domain=somecompany.co.uk; Path=/;
  Expires=Wed, 30 Aug 2019 00:00:00 GMT
```

4.14. מנגנון Snapshot Protection

כאשר משתמש האפליקציה לוחץ על כפתור הבית (Home button) במכשיר מסוג iOS, האפליקציה מתכווצת, אך עדיין רצה מאחורי הקלעים. פעולת הכיווץ יוצרת תמונה ממוזערת המתארת את המצב האחרון שבו הייתה האפליקציה לפני הלחיצה על כפתור הבית. התמונה הממוזערת נשמרת בתיקיית האפליקציה (בד"כ בתיקיה /private/var/mobile/Containers/Data/Application/<App-UID>/Library/Caches/Snapshots/ , תלוי בגירסת ה-iOS), ועלולה להכיל מידע רגיש. אחרת, במידה ולא קיים מנגנון נגד צילום מסך האפליקציה, עלולה להתרחש דליפת נתונים רגישים מתוך האפליקציה על ידי צילום המסך השמור בצורה שאינה מאובטחת על מכשירי iOS. גורם זדוני יכול לנצל חשיפה זו ולשלוף מהמכשיר את תמונת המצב האחרון של האפליקציה, ועל ידי כך לדלות מידע רגיש אשר קיים בה.

על כן, יש ליישם מנגנון Snapshot Protection על ידי הסרת כל מידע רגיש בעת המעבר של האפליקציה ל-background. פעולה זו ניתן ליישום ע"י שימוש בפונקציה applicationDidEnterBackground או applicationWillResignActive.

- יש לסמן את UITextView ו-UITextField המכילים נתונים רגישים כ-secure באמצעות הגדרת המאפיין secureTextEntry כ-yes:

```
UITextField* userPassword = [[UITextField alloc] init];

userPassword.secureTextEntry = YES;
```


Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

- אפשרות נוספת – ניתן לסמן את אותם שדות רגישים כ-hidden (מה שיגרום לאותם שדות לא להיות מוצגים כלל במסך של ה-snapshot. כאשר האפליקציה חוזרת להיות 'פעילה' יש להחזיר אותם להיות מוצגים כרגיל):

```
[[NSNotificationCenter defaultCenter] addObserver:self

    selector:@selector(didEnterBackground:) name:@"didEnterBackground" object:nil];


[[NSNotificationCenter defaultCenter] addObserver:self

    selector:@selector(didBecomeActive:) name:@"didBecomeActive" object:nil];

- (void)didEnterBackground:(NSNotification *) notification
{
    self.dbPassword.hidden = YES;
    self.creditCardNumber.hidden = YES;
}

- (void)didBecomeActive:(NSNotification *) notification
{
    self.dbPassword.hidden = NO;
    self.creditCardNumber.hidden = NO;
}
```

4.15. טיפול בשגיאות

- כברירת המחדל, במקרה ומתרחשת שגיאת זמן ריצה שלא נתפסה ע"י האפליקציה, יופעל ה- global exception handler אשר יגרום לסגירת האפליקציה.
- יש להשתמש ב-try&catch על מנת לתפוס שגיאות.
- יש להשתמש במילת הקוד throws על מנת להגדיר אילו פונקציות ומתודות יכולות לזרוק שגיאה. במקרה שהמילה throws הוגדרה, יש להשתמש ב-try, catch, do על מנת לטפל בשגיאה הפוטנציאלית.
- במקרה ורוצים לנהל שגיאה ולהמשיך את ריצת האפליקציה, במקום לזרוק את השגיאה, ניתן להחזיר את הערך nil על ידי שימוש ב-try?
- אין להשתמש ב-try! אשר יגרום לאפליקציה לא לטפל בשגיאות אפשריות.
- יש לייצר הודעת שגיאה גנרית שאינה חושפת מידע למשתמש, ומספקת לו קוד שגיאה כללי בלבד.

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570

4.16. שימוש בפונקציות בשפת C

- ב-Swift יש את היכולת לקרוא לפונקציות C על ידי שימוש ב-Unsafe pointers. מומלץ שלא להשתמש ביכולת זו מכיוון ועל ידי שימוש בהן, נחשפת האפליקציה לפגיעויות מסוג Overflow למיניהן (Stack, Heap) הנובעות מכך שאין למצביעים ומערכי מחרוזות יכולת לבדוק טווח ערכים תקין.
- במידה ויש שימוש ב-Unsafe pointers, כגון במקרה של שימוש בספריות כגון Core Foundation API הכתובות ב-C, יש להיזהר מבעיית Buffer Overflow בעת העתקת מידע לזיכרון מבלי לבדוק האם יש במקום אליו מעתיקים די מקום לתוכן המועתק. מקור התוכן המועתק יכול להיות שדות (TextBox), מידע משורת הכתובת (URL), מוזיקה, גרפיקה וכל משאב שבו יש למשתמש שליטה והאפליקציה לוקחת את התוכן ומעתיקה לתוך משתנה וכד'.

4.17. הזדהות והרשאות

- במקרה בו על נתוני הזדהות לעבור ברשת מול דף login, יש להשתמש בתווך מוצפן.
- בתום תהליך הזיהוי, יזוהה המשתמש באמצעות מזהה ה-Session (ניהול Session יבוצע בצד השרת). יש להימנע מלהעביר את נתוני הזיהוי בכל בקשה.
- אין לממש מנגנון הזדהות אשר מתבסס על ערך שאינו אמין דיו כגון כתובת IP, שם מכשיר הטלפון, user agent וכדומה.
- יש לאכוף מורכבות סיסמא, כלומר: על הסיסמא לעמוד במדיניות הסיסמאות של המשרד ולכלול לפחות 8 תווים, המכילים אות גדולה, אות קטנה, ספרה וסימן.
- בעת ביצוע פעולות רגישות, דרוש אימות נוסף של המשתמש כחלק מתהליך MFA (Multi Factor Authentication). יש לוודא כי האימות הנוסף הינו בלתי תלוי באימות הראשוני של המשתמש (למשל, דוגמאות לאימות נוסף: באם המשתמש הזין סיסמה, האימות הנוסף יהיה באמצעות משלוח קוד חד-פעמי OTP למספר הטלפון של המשתמש המשוייך לחשבונו). אימות שכזה מספק רמת וודאות גבוהה יותר כי המשתמש אינו מתחזה.
- יש לבצע בדיקות אבטחה ואכיפת הרשאות בצד השרת בלבד. כל הבדיקות אשר מבוצעות בצד הלקוח אינן רלוונטיות מבחינת אבטחה, ולכן אין להתחשב בהן כגורם אכיפה.
- אין לתת הרשאות לקבצים מעבר להרשאות ההכרחיות. את ההרשאות יש לקבוע באמצעות המחלקה NSFileManager, ע"י הפונקציה createFileAtPath, תוך קביעת ערך attributes בהרשאות מינימליות הנדרשות לפעילות האפליקציה.

Information Systems
Ministry of Health
80 Jeremiah street,9101002 Jerusalem
Tel: 02-6748265 Fax: 02-6715570

אגף מערכות מידע ומחשוב
משרד הבריאות
רח' ירמיהו 80, ירושלים מיקוד 9101002
טל: 02-6748265 פקס: 02-6715570