ממשל זמין – פרויקט תהיל"ה

הנחיות פיתוח מאובטח עבור מערכות המאוכסנות בתהיל"ה

גרסא 1.2

ממשל זמין – פרויקט תהיל"ה

מאפייני מסמך

מחבר	נמרוד לוריא, גיל כהן, דניאל סמירנוף
מספר גירסה	1.2
סטטוס	הפצה
תאריך הוצאה	פברואר 2013
שם קובץ אלקטרוני	Application_dev_rules

תשומות / הערות

שם/תפקיד	הערה (אופציונאלי)	תאריך	חתימה

היסטוריה

מ. גרסה	ת. הוצאה	מחבר	שינויים מרכזיים בגרסה
1	יוני 2009	נמרוד לוריא	
1.1	נובמבר 2011	דניאל סמירנוף גיל כהן	הורדת פסקאות מיותרות, הוספת סעיפים נוספים וחידוד פרטים
1.2	פברואר 2013	גיל כהן	הוספת התייחסות למתקפות הצפה ומניעתן על ידי CAPTCHA או NoBot

ממשל זמין – פרויקט תהיל"ה

תוכן עניינים

1. כללי

ממשל זמין מספק שירות אירוח של אתרי אינטרנט עבור משרדי הממשלה.

עיקר פעולתם של האתרים הממשלתיים מתבטא במתן מענה לשלושה צרכים עיקריים:

- שירותים מקוונים במסגרת ממשל זמין (תשלומים, טפסים, פניות ציבור וכו')
- מקור מידע רשמי ועדכני של נתונים, פרסומים והודעות לכלל הציבור
- חלק ממערך ההסברה של מדינת ישראל

המידע המאוחסן על שרתי האינטרנט בממשל זמין, הינו רשמי ובדרך כלל רגיש.

מערכות אלה נתונות תחת נסיונות השחתה, החדרת תעמולה, גניבה, שיבוש מידע, השבתה ומניעת שירות קבועים.

חולשת אבטחה אחת יכולה להספיק בכדי להשתלט על מערכת, ולהשתמש בה כעוגן להמשך התקפות בסגמנטים פנימיים של רשת האירוח.

ללא קשר לחשיבות המידע אותו הם מציגים, כל אתרי האינטרנט המתארחים בפרויקט נתונים תחת ניסיונות פריצה והתקפה, ולכן שירות זה כפוף למדיניות אבטחת מידע נוקשה.

כלל המפתח ביישום מדיניות זו הוא, שכל פגיעה באתר או מערכת ממשלתיים, כמוה כפגיעה בנכס ממשלתי ולצורך העניין, פגיעה בממשלה. עקרונות אבטחת המידע נקבעו ואושרו ע"י החשב הכללי, מנהלת הרשת הממשלתית, אגף ביטחון באוצר וחברות אבטחת מידע.

מסמך זה מטרתו להדגיש את נושאי האבטחה השונים בעת פיתוח אפליקציה אשר תתאכסן במתקני תהיל"ה. מסמך זה מתאר באופן כללי את הגישות שבהם יש לנקוט על מנת לאפשר פיתוח מערכת מאובטחת. חשוב להדגיש כי מסמך זה הינו בסיסי והדרישות עלולות להשתנות בין המערכות השונות בהתאם לרגישות ולסווג המערכת.

1.1 תיאור האיומים

שלוש רמות עיקריות:

- **הרשת (Network)** (שימוש בפרוטוקולים אסורים, הצלבת רשתות...)
- **מערכת ההפעלה** (הרשאות לקויות, שימוש בשירותים אסורים...)
- **היישום (Application)** (עיבוד שגוי של קלט משתמש, הרשאות ...)

סכנות ואיומים נפוצים על יישומי אינטרנט:

- השחתה "רועשת" - החלפת דפים, מחיקת התוכן (defacement)
- השחתת מידע, זיוף והצגת מידע שגוי (ובכך פגיעה במשתמשים ובמהימנות של המערכת)
- מניעת שירות - האטת המערכת או השבתתה
- הונאה – הפעלת יישומים במירמה, גניבת כסף
- גניבת מידע, ריגול אחר משתמשים תוך גניבת זהות, התחזות
- השתלטות, חדירה פנימה לתוך הארגון
- פישינג

טכניקות פעולה:

- איסוף מידע – חשיפת כל פרט אפשרי על המערכת: סביבות עבודה, גרסאות תוכנה, תהליכים, פורטים פתוחים, שירותים פעילים וכו'.
  מתבצע בד"כ ע"י כלי סריקה (סקאנרים) אך גם בצורה ידנית
- למשל ערעור היציבות של המערכת, ותוך כך הפקת הודעות שגיאה
- המידע שהתקבל על המערכת מאפשר ניצול חולשות ברכיבים שנגלו
  - הרצת קוד מרחוק למשל ע"י ניצול זליגת חוצצים (Buffer Overflow)
  - ניצול לוגיקה בצד הלקוח להזרקת קוד ושיבוש המידע שמועבר למערכת:
    - Cross-site scripting
    - HTML, XML, LDAP, SQL injection
  - ניצול הרשאות לקויות
  - Directory traversal
  - ניצול הזדהות חלשה של משתמשים, ניצול מנגנון ניהול משתמשים לא מאובטח (להתחזות,ניחוש משתמשים וסיסמאות וכו')
  - הסנפת תעבורה לא מוצפנת Man in the Middle
  - וכו'...

אכיפת מדיניות אבטחת המידע מתבצעת כחלק מהפעילות השוטפת.

תנאי ראשון ביישום המדיניות הינו הקפדה על תכנות נכון ואיפיון של האפליקציות בהתאם לכללים המובאים.

2. הנחיות לפיתוח מאובטח

2.1 מדיניות סיסמאות

- הסיסמא לא תעבור גלויה ברשת אלה בצורה מוצפנת \ ב hash או על גבי תווך מוצפן.
- המערכת תספק למשתמש את היכולת להחליף את הסיסמא בעצמו, בצורה בטחה, בכל עת.
- אורכם של שמות המשתמשים של המערכת יהיה לפחות 8 תווים.
- לא יוגדרו במערכת משתמשים בעלי שם משתמש טריוויאלי, כגון 'admin'.
- המערכת תכיל בדיקה שתוודא ששם המשתמש והסיסמא יהיו שונים זה מזה ולא יכילו אחד את השני (שם המשתמש לא יהיה חלק מהסיסמא ולהיפך).
- סיסמת המשתמש לא תהייה קצרה מ-10 תווים ותהייה מורכבת לפחות מ-3 קבוצות תווים מתוך הארבע הבאות:
  - אותיות קטנות;
  - אותיות גדולות;
  - ספרות;
  - תווים מיוחדים.
- סיסמתו של מנהל המערכת (אדמיניסטראטור) תהייה באורך של 12 תווים לפחות.
- תוקפה של סיסמת המשתמש יפוג כל 60 יום ועל המשתמש יהיה להחליף את סיסמתו בהתאם למבנה המתואר לעיל.
- מנגנון החלפת הסיסמא ישמור היסטוריית הסיסמאות של 5 מחזורים לפחות ולא יאפשר למשתמש לחזור על אף אחת מהסיסמאות הללו בעת החלפת הסיסמא.
- טרם החלפת הסיסמא על המשתמש יהיה להקיש את סיסמתו הנוכחית.
- החלפת הסיסמא לא תתאפשר בטווח של 24 שעות מהחלפת הסיסמא האחרונה.
- הסיסמא הראשונית של המשתמש תהייה רנדומאלית ובהתאם למבנה שהוגדר לעיל.
- המערכת תחייב את המשתמש להחליף את סיסמתו הראשונית בעת ההתחברות הראשונה למערכת.

- תוקף הסיסמא הראשונית יהיה 3 ימים, ולאחר מכן המשתמש ינעל ולא יוכל להשתמש בה.
- במקרה בו המשתמש שכח את סיסמתו, המערכת תיצור לו סיסמא חדשה. כמו הסיסמא הראשונית, סיסמא זו תהייה מוגבלת בתוקף והמשתמש יהיה מחויב להחליפה בעת השימוש הראשון בה.
- יצירת סיסמא חדשה במקרה בו המשתמש שכח את הנוכחית תהייה אך ורק לאחר זיהוי המשתמש באמצעים אחרים, כגון כתובת דואר אלקטרוני, שאלות סודיות וכדומה. מומלץ להשתמש בשילוב של השיטות כגון: שליחת מייל למשתמש עם קישור חד פעמי ומוגבל בזמן המשמש לאיפוס סיסמא, ואחרי שהמשתמש גולש הוא נשאל שאלות בטחון ואם הוא מצליח אז ניתנת לו האפשרות לשנות את סיסמתו.
- אין להציג בשום שלב במחשבי המערכת, במחשבים של משתמשי המערכת, בקוד המקור של דפים וטפסים המועברים למשתמש, את מזהי האימות של המשתמשים השונים במערכת.
- סיסמת המשתמש תשמר בצורת hash בבסיס המידע.
- יש להשתמש באלגוריתמי Hash בטוחים כגון Sha-256 או Sha-512. יש להימנע משימוש באלגוריתמים לא בטוחים כגון Sha-1 ו-MD5.

2.2 נעילת משתמשים

- נעילת המשתמשים תתבצע לאחר 3 ניסיונות הזדהות כושלים.
- במקרה של מיעוט משתמשים אין להשתמש במנגנוני שחרור אוטומטי, אלא השחרור יבוצע על ידי מנהל המערכת לאחר קבלת הפניה מהמשתמש ווידוי זהותו.
- במקרה של מערכת פתוחה לציבור ו/או בעלת משתמשים רבים יש ליישם מנגנון שחרור אוטומטי לאחר פרק זמן קבוע (למשל 30-15 דקות).
- נעילת המשתמש תתבצע בצד שרת המערכת ולא ברמת ה-Session או ה-Client.
- משתמש ניהול המערכת לא ינעל לאחר ניסיונות זיהוי כושלים על מנת למנוע מצב של מניעת שירות של המערכת.
- במקרה של נעילת המשתמש אין להודיע על כך באופן מפורש בכדי למנוע מצב של מיפוי שמות משתמשים בעזרת נעילה מכוונת. ניתן לדווח על הנעילה ישירות

לדואר האלקטרוני של המשתמש או להטמיע אותה בהודעת הכישלון בהתחברות הרגילה ("שם משתמש וסיסמא לא נכונים, יתכן והמשתמש ננעל אם ניסית מספר רב של פעמים")

2.3 ניהול משתמשים והרשאות

- הרשאותיהם של המשתמשים יקבעו לפי עקרון ההרשאות המינימאליות הדרושות, כלומר כל משתמש מערכת יקבל את הרשאותיו בהתאם לדרישות עבודתו במערכת ולא מעבר לכך.
- הרשאות המשתמש ייבדקו בכל השכבות ובכל הרכיבים של המערכת.
- יש לבצע בדיקת הרשאות משתמש בכניסה לכל דף במערכת.
- יש לבצע בדיקת הרשאות משתמש טרם ביצוע פעולות במערכת לרבות פעולות צפייה במידע, מחיקה, עדכון או הוספה.
- אין להסתמך על מנגנון זיהוי כמנגנון הרשאות. משתמש מזוהה במערכת אינו בהכרח מורשה לכל חלקיה.
- בקרת הגישה תתבצע בצד השרת בלבד ולא תסתמך על נתונים השמורים במחשבו של הלקוח, לדוגמא cookies.

2.4 אימות קלט

- אימות קלט משתמש יבדק בשכבות השונות בהתאם לסוג המידע שאמור להתקבל שימוש ב regular expression – white list, קרי, סינון על פי ערכים מותרים ידועים מראש ולא שלילת ערכים, וזאת משום שניתן להציג קלטים ביותר מצורה אחת על ידי שימוש בקידוד שונה.
- הבדיקות יתבצעו גם בצד המשתמש וגם בצד הלקוח.
- מומלץ לבצע אימות קלט בכל אחת משכבות האפליקציה – למשל קוד, מסד נתונים, שכבות Web Services וכו'.
- בגישה ל WS וגישות SOAP באמצעות XML, יש לבצע בדיקות לקלט שמועבר למערכת לפי סכמות XSD מוגדרות מראש לכול פעולה \ מתודה בשרות אליו מתבצעת הגישה, על פי נוהל פיתוח סכמות מאובטחות WS-Gov.il.

2.5 בקרת קלט

- בנוסף למנגנונים התשתיתיים בממשל זמין הבאים למנוע התקפות הצפה, יש להטמיע גם ברמת האפליקציה מנגנוני בקרת קלט המונעים הצפה.
  לצורך כך ישנם 2 מנגנונים כללו הנמצאים כיום בממשל זמין:
  - מנגנון CAPTCHA המציג למשתמש תמונה המכילה אותיות ומספרים ומחייבת אותו להקלידם על מנת לוודא שמדובר באדם הגולש באתר ולא בקוד אוטומטי הבא להציף את האתר.
  - מנגנון NoBot של מיקרוסופט המגביל את כמות הבקשות הנשלחות מכתובת IP מסויימת בחלון זמן קבוע. מנגנון זה נגיש כיום לטכנולוגיית NET. בלבד.
- יש להשתמש במנגנונים הסטנדרטיים הקיימים בממשל זמין ולא לפתח מנגנוני הגנה באופן עצמאי או להשתמש במוצרים של צד שלישי ללא התייעצות קודמת עם צוות אבטחת מידע וצוות תפעול של ממשל זמין. רק במקרים מיוחדים ולאחר התייעצות יאושרו מנגנונים שאינם הסטנדרטיים.
- יש להטמיע את מנגנוני בקרת הקלט על כל עמוד ושדה קלט המשמש להכנסת נתונים למאגר כלשהו (כגון DB) או גורם לשליחה של הודעה במערכת מסרים כלשהו (כגון שליחת Email) הנגיש באופן פומבי ולא מחייב הזדהות בטרם הגישה אליו. ניתן אך לא חובה לממש את מנגנונים אלה גם במקומות נוספים כמו בשדות הנמצאים מאחורי מנגנוני הזדהות, במנגנון Login עצמו וכו'.

2.6 הגנה על מידע רגיש

- יש להצפין נתונים רגישים במערכת. כגון:
  - נתונים רגישים וחסויים של משתמשי המערכת.
  - במידה וקיימים קבצים (כגון קבצי Word, PDF, תמונות וכדומה) אשר מכילים מידע רגיש בבסיס הנתונים יש להצפינם גם כן.
  - נתוני זיהוי של רכיבי תוכנה שונים, כגון נתוני הזיהוי של שרת האפליקציה לשרת בסיס הנתונים (Connection String) וכדומה.

- מפתח ההצפנה יישמר במקום מאובטח על שרת המערכת, כגון ה-Registry. הגישה למפתח תוגבל לאפליקציה ולאדמיניסטראטור של השרת בלבד.
- יש לבצע הצפנה של המפתח ע"י שימוש בהצפנת DPAPI, יש לשמור עותק של המפתח במקום מוגן נפרד (פיזי) למקרה שלא ניתן לשחזר את המפתח המקורי.

- אחסון סיסמאות באופן מאובטח תעשה באופן הבא:
  - הסיסמאות אינן דורשות הצפנה דו כיוונית כיוון שאין צורך באחזורם, לפיכך הסיסמאות ישמרו בבסיס הנתונים לאחר ביצוע HASH על ערכן.
  - לכל משתמש בעת יצירת סיסמא ייבחר ערך רנדומאלי אשר ישורשר לסיסמא טרם ביצוע ה-HASH. ערך זה נקרא ערך SALT.
  - ערך ה-SALT ישמר בבסיס הנתונים יחד עם פרטי המשתמש.
  - על מנת לבדוק כי הסיסמא שהמשתמש הזין הינה נכונה, משרשרים אליה את ערך ה-SALT מבסיס הנתונים ומבצעים על הערך החדש את פעולת ה-HASH שבוצעה בעת שמירת הסיסמא. אם ערך ה-HASH החדש תואם את ערך ה-HASH אשר שמור בבסיס הנתונים, הרי שהסיסמא נכונה.
- יש להשתמש באלגוריתמי Hash בטוחים כגון Sha-256 או Sha-512. יש להימנע משימוש באלגוריתמים לא בטוחים כגון Sha-1 ו-MD5.
- יש להשתמש בהצפנות מקובלות כיום בשוק, כגון RSA, ולא לבנות אלגוריתם הצפנה ייחודי למערכת.
- אין לאפשר שמירת נתונים רגישים של המערכת במחשבו של המשתמש.
- יש למנוע את שמירת נתוני המערכת בספריית הקבצים הזמניים ובמנגנוני ה-Cache במחשב המשתמש.

2.7 הגנה על מידע בתעבורה

עקב רגישות המידע והעברתו בתווך אינטרנט, יש להעביר את כלל המידע הרגיש באופן מוצפן, לרבות כל המידע והמסכים המוצגים לאחר הזדהות כלשהי בפני מערכת בתהיל"ה.
כל התעבורה תתבצע בתווך מוצפן.
מומלץ להוסיף הגבלות אלו ברמת האפליקציה.

אין חובה להוסיף הגנה באמת התעבורה על אתרים פומביים שאינם מכילים מידע רגיש או הזדהות.

2.8 ניהול מופעי משתמשים (Session Management)

- יש להבטיח כי נתוני session נשמרים בצורה בטוחה במהלך חיי המערכת ובפעולת המערכת השונות המתבצעות עם האובייקטים \ משתמשים.
- יש להבטיח כי קיימת הפרדה בין ניהול הזהויות לבין שימוש ב session כך שלא יתכן מצב כי משתמש שלא ביצע הזדהות יוכל להשתמש ב session פעיל של משתמש שביצע הזדהות כנדרש (גניבת זהות), כלומר יש להבטיח כי המערכת אינה מסתמכת על נתוני session בכדי לאפשר למשתמש חשיפה למידע ופעולות רגישים במערכת.
- יש להשתמש ברכיבי session רק עבור שמירת מצב משתמש בין בקשות http שונות במערכת וכן לצורך ביצוע personalization עבור משתמש.
- אין לשמור מידע רגיש ב SESSION , במידה ונדרש יש לבצע הצפנה של מידע זה.
- בכל מצב שבו נשמר מידע רגיש ב session יש להבטיח כי המידע נשמר בצורה בטוחה ולא תתאפשר גישה אליו שלא דרך מקור מוסמך ומאושר (כלומר מהאפליקציה שייצרה את המידע) .
- על המערכת להימנע במידת האפשר בשימוש ב client – side state management כגון view state, hidden files,cookies לצורך קבלת נתונים עבור session .
- האפליקציה תעשה שימוש רק בזהות אשר נתקבלה בתהליך ההזדהות בכניסה לאפליקציה ואשר מבצעת שימוש ב- Session ID ייחודי וזמני.
- יש למנוע ביצוע גישה למערכת ללא SESSION תקין.
- יש למנוע ביצוע גישות מרובות מאותו SESSION למערכת.

הנחיות נוספות לגבי ניהול מופעי משתמשים כפי המופיע במסמך האפיון (פרק 5) :

- אין להעביר את נתוני הזיהוי של המשתמשים בין מחשב המשתמש לשרתי המערכת, למעט דף הכניסה למערכת.

- יש לקיים מנגנון Idle Timeout אשר יסיים את ה-Session של המשתמש לאחר מספר דקות מוגדר, כ-15 דקות, של חוסר פעילות במערכת.
- יש לקיים מנגנון Session Timeout אשר יסיים את ה-Session לאחר זמן ארוך של פעילות במערכת, כ-8 שעות. מנגנון זה נועד למנוע שימוש במערכת באמצעות סקריפטים וכדומה.
- יש לשקול את ניתוק Session במצבי שגיאה מסוימים.
- ניתוק ה-Session יבוצע על ידי סיום תוקף ה-Session בצד השרת, ולא על ידי העברת הלקוח לדף הכניסה בלבד.

2.9 ניתוק מערכת

- האפליקציה תאפשר יציאה מסודרת ונוחה מהמערכת בכל דף החל מדף הכניסה (Login).
- ניתוק זה יבטיח כי משתמש לא יוכל לבצע שימוש חוזר במערכת ללא ביצוע הזדהות מלאה מחדש וזאת על ידי סגירת ה-Session שלו וכלל המשאבים שהוקצו לו בטרם הניתוק.
- במקרה של זיהוי פעילות חשודה במערכת (כפי שהוגדרה במידול הסיכונים) כגון ניסיונות לביצוע sql injection או הזנת סקריפטים זדוניים בשדות קלט, נדרש לבצע ניתוק כפוי של המשתמש, לבצע רישום ללוג וכן להתריע על כך למנהל המערכת.

2.10 שימוש בתעודות והצפנות

עבור מידע המוגדר כרגיש, יש לאפשר טיפול באמצעי מידור הן ברמת מנהלי המערכת והן ברמת המשתמש. כולל:

- תמיכה בסוגי מידע שונים.
- יכולת הגדרה במערכי ה-Audit לרישום גישה או ניסיונות גישה למידע המוגדר כרגיש. רישום ה-Audit יבוצע באופן מלא בכל שכבה, ובביצוע האחזור ניתן יהיה להפריד באופן מובהק בין התהליכים ובין השכבות השונות שבהם בוצע ה-Audit.

במערכת מידע נדרש לבצע הצפנה לפי הכללים הבאים:

כאשר מתבצעת הצפנה למידע רגיש יש לממש אלגוריתמי הצפנה לפי הכללים הבאים:

- אין לבצע שימוש באלגוריתמים שפותחו בצורה עצמאית.
- יש לבצע שימוש באלגוריתמים מוכרים כגון:
- AES עבור הצפנה סימטרית
- RSA עבור הצפנה א-סימטרית
- Sha-2, Sha-256 או Sha-512 עבור hash חד כיווני
- עבור יצירת מספרים רנדומאליים יש להשתמש במנגנון מבוסס crypto random generator

הגנה על מפתח\ות הצפנה:

- יש לאבטח את מפתח \ מפתחות ההצפנה הנמצאים בשימוש המערכת מפני גישה \ שימוש זדוני ללא הרשאה בהתאם לסוג המפתח – ציבורי \ פרטי.
- יש להגן על המפתח מפני הרס או שינוי בצורה לא מורשת.
- יש לנהל בקרה ודיווח לגבי ביצוע גישות ושימוש במפתחות הצפנה.
- יש להבטיח יכולות שיחזור וגיבוי בשימוש במפתחות הצפנה (כדי להבטיח שיהיה ניתן לשחזר מידע רגיש שהוצפן עם מפתח שאבד).
- על המערכת להימנע משמירת מידע רגיש בקובצי הגדרות, קבצים זמניים, cookies, זיכרון מטמון וכו'. במידה ומידע נשמר במקומות אלו, נדרש לוודא כי לאחר סיום עבודה במערכת מידע שיעורי זה ימחק.

2.11 ניהול שגיאות

- הודעות שגיאה שיוצגו למשתמש כתוצאה משגיאות המתרחשות באפליקציה יהיו הודעות שאין בהן כדי לחשוף את אמצעי האבטחה במערכת. יש לוודא כי הודעות שגיאה אינם חושפות מידע רגיש בנוגע למבנה המערכת ומשאבי המערכת. הודעות השגיאה שיוצגו יהיו ג'נריות וכלליות.
- הודעות שגיאה שיוצגו למשתמש יהיו הודעות שאין בהן כדי לחשוף את התשתית האפליקטיבית לגרסאותיה השונות כגון: מערכות הפעלה, שרתי web, שרתי אפליקאציה, בסיסי נתונים, פרוטוקולים בשימוש, Web Services בשכבות נמוכות וכדומה.

- אין להציג כל מידע רגיש (כולל : מספרי אשראי, סיסמאות, מפתחות הצפנה וכו') בהודעות שגיאה המוצגות למשתמש.
- כאשר קלט המשתמש אינו מתאים לתבנית הנדרשת בשדה קלט, יש להציג למשתמש הודעת שגיאה המפרט מהי התבנית בה נדרש להשתמש.
- על המערכת לנהל מערך ללכידת שגיאות בזמן ריצה :
  - יש לצפות שגיאות מראש וללכוד אותן בקוד המערכת.
  - בשגיאות שהוגדרו כשגיאות כתוצאה מפעילות הקשורה באבטחת מידע יש לנהוג לפי מה שהוגדר במידול הסיכונים של המערכת, כולל דיווח למנהל המערכת, חסימת משתמש וכו'.
- יש לדאוג לכך שמידע משגיאות יהיה מתועד ע"י המערכת בדפי ה log שלה.
- ניתן להציג למשתמש קוד המתאים לרשומה בלוג לשם טיפול בשגיאה. קוד הרשומה לא ירמוז בשום צורה על קוד השגיאה והסיבה להתרחשותה.
- על המערכת להתמודד עם שגיאות בהיבט של זמינות כך שאם למשתמש מסוים מתרחשת שגיאה הוא אינו חוסם גישה למשתמשים אחרים שמריצים את המערכת (קריסה כללית).
- במערכות רגישות ובסיכון בינוני ומעלה, המערכת תכלול יכולת לאחזור הודעות שגיאה (אחזור מלא, אחזור חלקי לפני פרמטרים שונים).
- פרמט הדיווח של הלוגים צריך להתאים לפורמט מערכת SIM \ SOC כך שיהיה ניתן לאסוף את הודעות השגיאה.

2.12 חיווי ובקרה

- האפליקציה תתעד את הנתונים הבאים, במידה והם מוגדרים, עבור כל פעולה במערכת:
  - Timestamp.
  - זיהוי המשתמש (ללא סיסמא)
  - מיקום המשתמש (מחשב/IP).
  - מיקום המשתמש במערכת (מסך, טופס, טבלה וכדומה).
  - פרטים מלאים של הפעולה המבוקשת.
- בנוסף יתועדו הפעולות הבאות:

- צפייה במידע במערכת.
- עדכון מידע במערכת.
- כתיבה ומחיקה של מידע במערכת.
- כל פעולות הניהול במערכת.
- כל פעולות הזיהוי במערכת, כולל כישלונות של פעולות אלו והסיבה לכך.
- כל פעולות ההרשאות במערכת, כולל כישלונות של פעולות אלו.
- שגיאות מערכת.
- ועוד, בהתאם לצורך.

- התיעוד יתבצע בשתי שכבות: תיעוד פעולות משתמשי מערכת באפליקציה, תיעוד גישה לנתוני המערכת בבסיס הנתונים.
- חשוב להדגיש כי התיעוד לא יכיל את נתוני הזיהוי של משתמשים או נתונים רגישים אשר שמורים בבסיס הנתונים של המערכת, בדגש על נתונים רגישים.
- כל פעולת תיעוד, בכל הרמות של המערכת, חייבת להכיל את המשתמש המבצע את הפעולה בפועל על מנת למנוע התכחשות משתמשים לפעולותיהם.

מעקב:

- יש לוודא כי נתוני התיעוד והמעקב נשמרים באופן מאובטח במערכת.
- יש לוודא כי רישומי התיעוד אינם נגישים למשתמשים ללא הרשאות מנהל מערכת.
- יש לוודא כי נתוני התיעוד מגובים יחד עם שאר נתוני המערכת.
- יש לקיים מנגנון ארכיב לנתוני תיעוד ישנים אשר אינם נחוצים לשם פעולתה התקינה של המערכת.
- יש להגדיר בתיאום עם מזמין המערכת את תקופת שמירת נתוני התיעוד של המערכת.

2.13 חתימת קבצים וקוד

- יש לבצע שימוש ב strong name ולחתום את קוד הפרויקט לאחר יצירת גרסת ייצור יציבה.
- מומלץ להחליף חתימה זאת בכל שחרור של גרסה חדשה.

CAS 2.14

- מומלץ לממש מנגנון CAS במערכת על מנת להגביל את גישת האפליקציה למקורות מידע שלא נדרש לבצע אליהם גישה כגון FTP, Unmanaged code וכו'.

2.15 ניהול הגדרות

- על המערכת לפרט באפיון את כל אמצעי גישות ניהול ההגדרות שבמערכת:
- יש להגדיר תחת איזה חשבון רצה המערכת (משתמש , מנהל , חשבון מערכת ...) , הדרישה היא כי המערכת תרוץ תחת חשבון עם רמת הרשאות הנמוכה ביותר הניתנת כך שלא תאפשר ביצוע פעולות לא רצויות ע"י משתמשים רגילים. לא יופעל שום רכיב עם זיהוי SYSTEM ו/או הרשאות ADMIN או מקבילות להם.
- במידה והמערכת דורשת הרשאות גבוהות רק בחלק קטן מהמערכת, יש להריץ את המערכת תחת משתמש נמוך הרשאות ולהתחזות למשתמש בעל הרשאות גבוהות יותר רק בקטע הקוד הרלוונטי.
- יש להגדיר דרכי גישה מאובטחות למשאבים חיצוניים כגון בסיס מידע, מערכת קבצים וכו' (למשל ע"י הצפנת מחרוזת קישור לבסיס מידע).
- יש להגדיר גישה מאובטחת לאדמיניסטרציה במערכת כולל זיהוי חזק והגבלת הגישה למורשים בלבד. יש לשקול הגדרת כתובות IP מסוימות שרק מהן ניתן לגשת לממשק הניהול.
- יש לדאוג לכך שלא יהיה ניתן לחשוף\לגשת למידע רגיש הקיים בקבצי הגדרות למשתמשים ללא הרשאות מתאימות. לשם כך יש להצפין הגדרות רגישות (ברמת האפליקציה ו/או ברמת מערכת הקבצים) ולהגביל אליהם גישה ברמת מערכת ההפעלה.
- אין לשמור מידע רגיש בקבצי הגדרות. במידה ונדרש יש להצפין אותו ע"י שימוש ב DPAPI.

2.16 התחברות למסד נתונים

- יש להצפין את הגדרות החיבור למסד הנתונים בקבצי הקונפיגורציה (Connection String)

- יש להשתמש בזיהוי מבוסס מערכת הפעלה (Windows Authentication) ולהעדיפו על פני זיהוי מסד נתונים במידה והדבר נתמך (למשל ב-SQL Server)

2.17 הגנה מפני מתקפות אפליקטיביות

מניעת התקפות cross site scripting

יש לבצע בדיקות תקינות בצד השרת על כל הקלט המגיע מצד המשתמש. בדיקת הקלט תכלול את הבדיקות הבאות:

- יש לבדוק את קיומו של הקלט ולא לאפשר הזנת ערכים ריקים.
- יש לבדוק ולהגביל את אורך הקלט (עפ"י האפיון שימוש ברשימות white list וב regular expression לפני הכנסת קלט למערכת).
- יש לבדוק שטיפוס הקלט המתקבל הוא מהסוג המצופה.
- יש לבדוק כי טווח הערכים שמתקבל מתאים להגבלות שנקבעו.
- יש לבדוק את הרכב התווים בקלט, ולוודא שהוא אינו מכיל תווים אסורים. ככלל יש להימנע ככל האפשר מקבלת קלט שאינו מכיל ערכים אלפא נומריים, למעט רווחים.
- יש לוודא כי ערכו של הקלט תואם ללוגיקה העסקית של רכיב היעד.
- יש לוודא כי הקלט ב encoding המתאים למערכת.
- יש להעביר את כלל התווים שאינם אלפאנומריים קידוד HTML בטרם הצגתם למשתמש. תהליך הקידוד יבטיח כי קוד שתול יוצג כטקסט ולא ירוץ על הדפדפן. מומלץ להשתמש בתשתית אפליקטיבית מוכרת (כגון AntiXSS ב-.NET).
- אין להכניס לבסיס הנתונים תווים הנובעים מקלט ישירות לתחום הפעולה של client side scripting (תגי script, אירועי HTML וכדומה).

מניעת הזרקות SQL

- אין לאפשר גישה ישירה לבסיס הנתונים. גישה לבסיס הנתונים תתבצע באמצעות שיכבה מתווכת כגון WS או DAL בפרויקט נפרד \ תשתית .
- בכל מקרה יש לבצע סינון מסודר של תווים למניעת הזרקת שאילתות SQL.
- כל תעבורת השאילתות תבוצע ע"י שימוש ב- stored procedures באופן הנכון וללא שימוש בהעברת פרמטרים בקריאה ל- stored procedure.
  בגישה לבסיס הנתונים (בשכבות ה- DAL או בכל מקום אחר) יש להשתמש בתשתית האפליקטיבית המתאימה והמומלצת לשם מניעת התקפות SQL

Injection:

ב-.NET: Parameterized Queries, (Linq to SQL, Entity Framework) LinQ.

ב-Java: Prepared Statements.

בכל מקרה יש להימנע משימוש בשרשור מחרוזות בעת הרכבת שאילתות או פקודות מסד הנתונים (לרבות פקודות עדכון נתונים).

מניעת מתקפות חסימת שירות

- יש לקחת בחשבון את כלל האיומים העלולים לגרור מתקפות Denial of service מניעת שירות) ולגבש בקרות כנגדם.
- יש להטמיע מנגנון NoBot או CAPTCHA כבקרת קלט לשם מניעת מתקפות הצפה.
- במנגנון נעילת משתמשים יש לקחת בחשבון את אלמנט הזמינות כך שיהיה ניתן לשחרר משתמש שננעל בצורה מהירה יחסית.

הגנה מפני buffer overflow

- יש לאמת פרמטרי מחרוזות כקלט ופלט – יש לוודא את אורך המחרוזת שלא תחרוג מהמקסימום.
- יש לאמת גבולות של מערכים.
- יש לאמת אורך נתיב לקבצים.
- בקוד unmanaged (C, C++) יש להימנע משימוש בפונקציות שאינן מומלצות כגון strcpy ותחתם להשתמש בגרסאות הבטוחות של אותן פונקציות כגון strncpy (פונקציות מתוך Strsafe.h).

הגנה מפני מתקפות Network eavesdropping

- הצפנת תווך תקשורת\ הודעה בזמן ביצוע הזדהות.
- הצפנת תווך תקשורת \ הודעה בזמן העברת מידע הקשור בפרטי זיהוי משתמש כגון החלפת סיסמא וכו'.
- הצפנת תווך תקשורת \ הודעה בזמן העברת מידע עסקי/אישי רגיש.

הגנה מפני מתקפות Brute force & Dictionary attacks:

- מימוש מדיניות סיסמאות חזקה.

- מימוש מנגנון נעילת משתמשים.
- שמירת סיסמאות ע"י שימוש ב hash בתוספת ערך רנדומאלי (Salt)

הגנה מפני מתקפות session hijacking ו session replay.

- אין לאפשר פתיחה של יותר מ session אחד עבור משתמש (במערכות רגישות בלבד).
- יש לבצע בדיקות אימות ל session לפני מתן גישה כלשהי.
- שימוש לבצע שימוש בתווך מוצפן כדי שלא יהיה ניתן לגנוב cookie המועבר לאפליקציה.
- למניעת מתקפות replay יש ליצור ערך חד ערכי עבור כל הודעה הנשלחת, כמו כן מומלץ לשלב חתימה בגוף ההודעה – timestamp.

מניעת שמירת נתונים במטמון הדפדפן

- אופציית אחסון הדפים (Caching) תהיה מבוטלת עבור כל הדפים באפליקציה ולכל סוגי הדפדפנים.

מניעת חשיפת תוכן תיקיות השרת

- יש לבטל את מאפיין ה-Directory Listing בכל אחת מהתיקיות הוירטואליות על שרת האפליקציה.

מניעת אפשרות אחסון פרטי הזדהות בדפדפן

- יש לבטל אפשרות ה- Password Auto complete ע"י שליחת מאפיין מתאים בתגי ה-Password וה-Form בדף ה-HTML. דוגמא:

```
<INPUT TYPE="password" AUTOCOMPLETE="off">
```

מניעת גישה לדפי בדיקות ודפים שאין אליהם קישורים נדרשים באפליקציה

- יש למנוע גישה לדפי סביבת הבדיקות ולהסיר אותם מסביבת הייצור של המערכת.
- יש לפעול לפי נוהל העברה ליצור מסודר.

3.נספחים

General Checklist 3.1

Check	Check Item
	If using Visual Studio .NET, application is compiled with the latest Visual Studio .NET /GS buffer overrun protection flag.
	If using Visual Studio.NET, debug builds are compiled with the /RTC1 flag.
	Any un-trusted input is validated for size, type and length prior to use or storage.
	Buffer management functions are safe from buffer overruns.
	Where possible, Strsafe.h (click here) is used.
	All DACLs are explicitly defined, not set to NULL or Everyone (Full Control)
	No hard-coded passwords or secrets (such as keys used in cryptosystems) are present.
	References to any internal resources such as user names and UNCs have been removed.
	Temporary file names are unpredictable.
	Error messages do not give too much information to an attacker (such as stack trace and extraneous drive information).
	Processes running with high privileges have been reviewed by at least one other person and business justification for privilege level has been validated.
	No user data is written to HKLM in the registry or “c:\program files”.
	Where impersonation is used, function return values are always checked.
	For every impersonation instance, there is a corresponding revert.
	Unauthorized connections are limited in the amount of resources (CPU, memory, disk space, etc.) they can use.
	Where sockets are used, application bind to explicit IP address instead of 0 or INADDR_ANY.

	At least one automated tool is used to identify code weaknesses before deployment.
	No sensitive data is embedded in configuration or XML files.
	Security decisions based on filenames are avoided where possible. When used, business justification is established and the code is reviewed by a security subject matter expert.

Web & Database Checklist 3.2

Check	Check Item
	Web pages do not output un-trusted data that has not been encoded first or filtered by AntiXSS Library (cross-site-scripting, or XSS, attacks).
	SQL statements are executed through parameter stored procedures, and not generated dynamically from un-trusted user input.
	The SA account, or any highly privileged account, is not used to create connections to the SQL server.
	Server validates access rights and does not rely on client-side verification.

Cryptography Checklist 3.3

Check	Check Item
	Secrets are not hard-coded into application code.
	SecureZeroMemory is used in place of ZeroMemory/memset when dealing with sensitive data such as session keys and key pairs.
	Crypographically strong random number generators are used in place were security decisions or processes are executed based on random data.
	Cryptograpic functions are implemented through the use of standard libraries such as CryptoAPI or System.Security.Cryptography and not developed in-house.

	Secret data is protected (DPAPI, crypto-stores, etc.)

Developer Security Checklist 3.4

Security Checklist	Link
Security Question List: Managed Code	http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnpag2/html/pagquestionlist0002.asp
Security Question List: ASP .NET	http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnpag2/html/pagquestionlist0001.asp
.NET Framework 1.1	http://msdn.microsoft.com/library/en-us/dnnetsec/html/CL_SecRevi.asp
.NET Framework 2.0	http://msdn.microsoft.com/library/en-us/dnpag2/html/pagck0003.asp
ADO.NET 1.1	http://msdn.microsoft.com/library/en-us/dnnetsec/html/CL_SecuDat.asp
ADO.NET 2.0	http://msdn.microsoft.com/library/en-us/dnpag2/html/pagck0002.asp
ASP.NET 1.1	http://msdn.microsoft.com/library/en-us/dnnetsec/html/CL_SecuAsp.asp
AS.NET 2.0	http://msdn.microsoft.com/library/en-us/dnpag2/html/pagck0001.asp
Enterprise Services	http://msdn.microsoft.com/library/en-us/dnnetsec/html/CL_SecuEnt.asp
Web Services .NET 1.1	http://msdn.microsoft.com/library/en-us/dnnetsec/html/CL_SecuWeb.asp
Building Secure Web Services	http://msdn.microsoft.com/library/en-us/secmod/html/secmod85.asp
Building Secure Service Components	http://msdn.microsoft.com/library/en-us/secmod/html/secmod84.asp
Building Secure Data Access	http://msdn.microsoft.com/library/en-us/secmod/html/secmod87.asp
Building Secure ASP.NET	http://msdn.microsoft.com/library/en-

Pages	us/secmod/html/secmod83.asp

Developer Security Tools 3.5

Tool Name	Tool Scope		Link
PRE-FAST	Unmanaged languages source code scanner (C/C++)		http://www.microsoft.com/whdc/archive/PREfast-drv.mspx
FxCop	.NET Framework languages assembly scanner (C#, VB.NET, etc.)		http://www.gotdotnet.com/Team/FxCop/
Visual Studio 2005 Team System Code Analysis Tool	Scanner for both unmanaged languages (C/C++) and .NET Framework languages (C#, VB.NET, etc.)	Available on Visual Studio 2005 Team Systems (Developer and Team Suite editions)	
Threat Modeling and Analysis Tool	Threat modeling tool for IT applications		http://msdn.microsoft.com/security/securecode/threatmodeling/acetm/
Anti-Cross Site Scripting Library V1.0	Managed library that implements white-list techniques for mitigations against cross-site scripting attacks.		http://www.microsoft.com/downloads/details.aspx?familyid=9a2b9c92-7ad9-496c-9a89-af08de2e5982&displaylang=en