מבוא:

הפרויקט שלנו הוא מוצר אבטחה אשר מאפשר למשתמשים הנרשמים אליו מעקב אחר ה-One Drive שלהם ולהודיע להם אם תוכנת כופרה השתלטה והצפינה קבצים בדרייב. אם כן, המוצר שולח מייל למשתמש ומתריע לו על כך.

המוצר שלנו הוא מוצר צד ל-Cloud. משמע, אין לנו גישה לתהליכים שרצים בשרת הענן עצמו או מידע שניתן לקבל ממערכת הפעלה. לפיכך, התמקדנו בפרויקט זה בניתוח נתונים על הקבצים כגון: אנטרופיית הקובץ, שינוי שם, שינוי סיומת והוספת קבצים חשודים ל-One Drive המכילים תוכן חשוד. כמו כן, הוספנו honeypots ב-One Drive של המשתמש לטובת זיהוי כופרה באופן מידי.

על תכנון הפרויקט וניתוח הקבצים נפרט בהמשך הפרקים.

סקירת ספרות:

על מנת להצליח לזהות התקפת כופרה, אנחנו צריכים תחילה להבין איך כופרה עובדת ואת ההשפעות שלה על קבצים של משתמש.

לפי המאמר Ransomware attacks: detection, prevention and cure שנכתב על ידי רוס ברואר [1] הרעיון של תוכנת כופר עלה לראשונה בשנת 1989 על ידי סוס טרויאני שנקרא PC Cyborg אשר הצפין קבצים ודרש תשלום לשחרורם. בהמשך השנים, תוכנות כופר הפכו לכלי הנפוץ ביותר להתקפת עסקים.

במאמר מצוינים ארבעה שלבים של הכופרה:

1. ניצול וזיהום – קובץ של תוכנת הכופר צריך להיות מורץ במחשב המותקף.
2. הפעלה – קובץ ההפעלה של הכופרה צריך להיות מופעל במחשב של הקורבן.
3. הריסת גיבויים – תוכנת הכופר מזהה גיבויים במחשב הקורבן ומסירה אותם כדי שהקורבן לא יוכל להשתמש בהם.
4. הצפנת קבצים – התוכנה מסכימה על מפתח הצפנה עם ה-C&C ומצפינה את הקבצים. ההצפנה יכולה לקחת פרק זמן לא מבוטל (בסטנדרטים של מדעי המחשב, דקות עד שעות). דבר זה נותן לנו את האפשרות לזהות את ההתקפה כשהיא מתרחשת.
5. הודעה למשתמש וניקיון – אחרי שהכופרה מסיימת את השלבים הקודמים, היא מודיעה למשתמש שהקבצים שלו הוצפנו. ההודעה עצמה כתובה בקבצים בעלי שמות כגון: "HELP_DECRYPT" או "HELP_YOUR_FILES" (אנחנו נשתמש בעובדה זו בפרויקט).

כעת, נפנה לחלק של זיהוי הכופרה. לפי המאמר על CryptoDrop [3] יש שלושה סוגי תוכנות כופר:

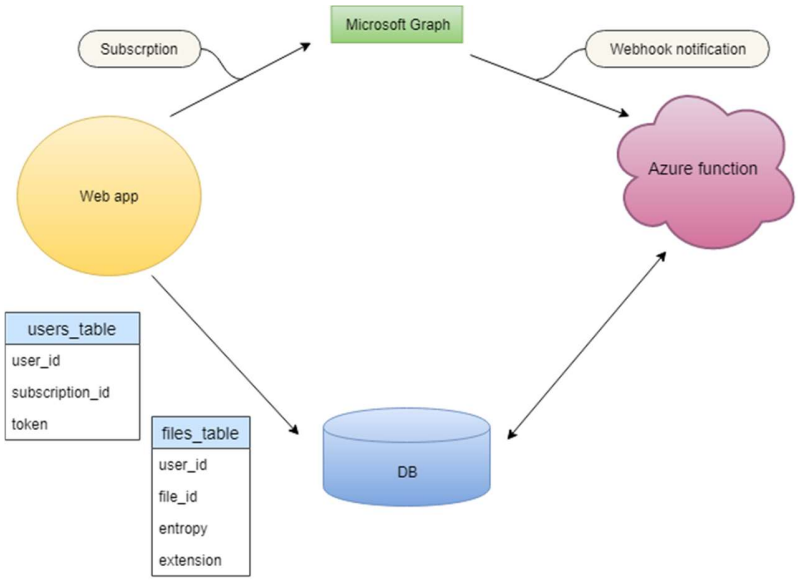
1. Class A – תוכנת הכופר דורסת את תוכן הקבצים המקוריים של המשתמש ועלולה גם לשנות את שמם.
2. Class B – פועלת בדומה לתוכנת כופר מסוג Class A רק שתהליך ההצפנה שונה: התוכנה מעבירה את כל הקבצים לתיקייה אחת, מצפינה אותם ומחזירה אותם למקומם המקורי מוצפנים.
3. Class C – תוכנת הכופר הזו מצפינה את הקבצים המקוריים על ידי יצירת קבצים חדשים עם התוכן המוצפן ומוחקת את המקוריים.

מאמר זה מדגיש כי יש מספר אלמנטים אשר תורמים לזיהוי התנהגות חשודה של קבצים:

1. שינויים של טיפוסי קבצים.
2. שינוי תוכן באופן דרסטי (הצפנה משנה בצורה מהותית את התוכן המקורי של הקובץ).
3. אנטרופיה של שאנון.
4. מחיקה עצומה של קבצים בבת אחת.

תכנון הפרויקט:

לפני שנתחיל לפרט על מימוש הפרויקט, נציג סכמה של המערכת לשם הנוחות:



הפרויקט כתוב כולו בפייתון, כאשר יש שלושה כלים שאנחנו משתמשים בהם לביצוע המשימה:

- Web application [3] אשר ממומש באמצעות Django framework[4] – האתר הוא נקודת הגישה של המשתמש להירשם לשירות שלנו. לאחר שהמשתמש נרשם לשירות, האתר אחראי על הוספת המשתמש ל-DB [] ואת המידע על הקבצים שלו גם כן ל-DB.
- Azure function [6] – הפונקציה בעצם אחראית על קבלת ה- Web hook notifications [7] מ-Microsoft. על כל קבלת נוטיפיקציה היא מריצה את אלגוריתם זיהוי הכופרה שממומש בפונקציה.
- Azure SQL Database [5] – בתוכו אנחנו שומרים את הפרטים של המשתמשים ואת הפרטים על הקבצים ב-OneDrive של המשתמשים שלנו.

כעת נרחיב על כל אחד מהכלים:

Web application:

כפי שצוין לעיל, אנחנו משתמשים ב-Django framework. תפקיד האתר הוא לספק נקודת גישה למשתמש להירשם לשירות שלנו.

ההרשמה לשירות שלנו כוללת קודם כל התחברות למערכת של מייקרוסופט באמצעות Microsoft OAuth 2.0 [8]. התחברות זאת מאפשרת לנו לקבל Token אשר באמצעותו נוכל להתחבר ל-OneDrive של המשתמש ולקבל מידע על הקבצים שלו ללא המשתמש והסיסמא שלו.

לאחר שהמשתמש מתחבר באמצעות הפרטים שלו ל-Microsoft Auth, האתר אחראי על המשימות הבאות:

- ביצוע Subscription לשירות ה-Webhook notification של Microsoft. את ביצוע ה-Subscription אנחנו מבצעים באמצעות GET http requests ל- Microsoft Graph RESTful API.
- הוספת המשתמש שהרגע נרשם לשירות לטבלת המשתמשים ב-DB (users_table) והוספת הקבצים שכבר קיימים ב-OneDrive של המשתמש עם הנתונים המתאימים להם לטבלת הקבצים ב-DB (files_table).
- מילוי ה-OneDrive של המשתמש ב-Honeypots.
- הוספת הנתונים על ה-honeypots לטבלה בשם honeypot_table.

- ביצוע Unsubscribe לשירות שלנו במקרה הצורך והסרת המידע הרלוונטי מה-DB.

Azure function:

הפונקצייה היא בעצם החלק הארי של הפרויקט, בה ממומש אלגוריתם זיהוי הכופרה ב-OneDrive של המשתמש.

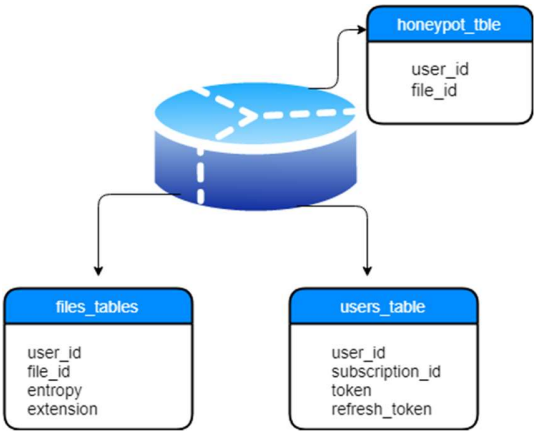
הפונקציה היא מטיפוס Http Trigger מהפונקציונליות של Azure. כלומר, היא מתבצעת כל פעם שמתקבלת נוטיפיקציה על שינוי ב-OneDrive מ-Microsoft Graph.

הפונקציה מקבלת מהנוטיפיקציה את ה- Subscription id שהשתנה, ניגשת ל-users_table, מוציאה את המשתמש שעבר את השינוי ומבקשת מ-Microsoft graph את השינויים עצמם על ידי קריאה לפונקציה delta [9]. לאחר זיהוי הקבצים שהשתנו, הפונקציה מפעילה את אלגוריתם הזיהוי עליהם. על לוגיקת האלגוריתם יפורט בהמשך.

כמו כן, הפונקציה דואגת לעדכן את ה-token במידה ופג תוקפו.

Azure SQL Database:

אנחנו מתחזקים 3 טבלאות שונות:



- טבלת משתמשים – משמשת אותנו לשמור את הקשר בין ה-Subscription id ל-User id. כמו כן, מכילה את ה-Token ואת ה-Refresh token במידה וצריך לעדכן את ה-Token של המשתמש (הוא פג תוקף לאחר פרק זמן מסוים).
- טבלת קבצים – משמשת אותנו לשמור מידע על קבצים ב-OneDrive של המשתמש. מתוך הטבלה הזאת נדע מה השתנה ואם יש דברים חשודים.
- טבלת המלכודות – משמשת אותנו לשמור את ה-Honeypots שהצבנו בדרייב כדי שלא יהיה ספק שאם הם השתנו, מדובר בכופרה.

לוגיקה:

BHKShield מבסס את בקרת הקבצים לפי 3 מאפיינים:

1. סריקת קבצים
2. Honeypots
3. סיומות ותכנים שהם Blacklisted

נפרט על כל מאפיין בנפרד.

סריקת קבצים:

אנו מחשבים תוחלת של מספר הקבצים החשודים.

תחילה, בודקים אם הקובץ הוא חדש. אם הוא אכן חדש, בודקים אם הוא בעל תוכן או סיומת חשודים. אם לא, מכניסים אותו ל-DB.

אנחנו שומרים ארבעה מונים אשר כל מונה סופר:

1. מספר סיומות חשודות
2. מספר תכנים חשודים
3. שינוי האנטרופיה [12]
4. מספר קבצים חשודים באופן כללי

אם הקובץ אינו חדש, אנחנו בודקים עבורו:

- האם הוא נמצא תחת הסיומות החשודות. אם כן, מוסיפים למספר הקבצים החשודים
- האם הוא בעל תוכן חשוד.
- השינוי באנטרופיה שלו ביחס לעדכון הקודם של הקובץ, מוסיפים למונה נפרד.
- האם הוא honeypot. אם כן, אנחנו מחזירים כי מדובר בכופרה.

בסיום מעבר על כל הקבצים שהשתנו ועל כל השינויים שעברו, אנחנו קוראים לפונקציה שתפקידה לשקלל את כל המונים שאגרנו עד עכשיו לערך מספרי אחד שיקבע האם כרגע התרחשה תקיפת כופרה או לא.

פונקציה זו מחשבת תוחלת עבור השינויים, כאשר ניתן משקל כבד במיוחד לשם חשוד, תוכן חשוד וסיומת חשודה מתוך הנחה שאלה בדיקות הסף שכדאי לבצע. אם אכן נפלנו בבדיקות אלו סימן שמתרחשת תקיפה.

אם לאחר הסריקה של כל הקבצים שהשתנו, התוחלת שקיבלנו גדולה מ-0.6 אזי מדובר בכופרה.

Honeypots:

קבצים שהוכנסו ל-honeypot_table ושונו מעידים על כופרה בוודאות (הלקוח לא אמור לשנות אותם).

סיומות ותכנים שהם Blacklisted:

הגדרנו שני Blacklists:

- Ransomware extension blacklist
- Ransomware file content blacklist

Ransomware extension blacklist:

סיומות של קבצים שלפי Crypto malware [11] נמצאו חשודות להתקפת Ransomware.

Ransomware file content blacklist:

תכנים של קבצים שלפי Crypto malware [11] נמצאו חשודים להתקפת Ransomware.

Microsoft Graph:

כחלק מתהליך הפיתוח של השירות, היינו צריכים לבקש את ההרשאות הבאות מ-Microsoft לאפליקציה שלנו:

1. Files.ReadWriteAll – ההרשאה מאפשרת לנו לקרוא ולכתוב קבצים ל- OneDrive של המשתמש.
2. User.ReadWrite – ההרשאה מאפשרת לנו גישה לפרטי המשתמש לטובת שליחת המייל.
3. Offline_access – ההרשאה מאפשרת לנו גישה לפרטי המשתמש כשהוא לא מחובר.

נקודות לציון:

דרישות הפרויקט:

בעת הפרויקט עמדנו בדרישות הבאות:

מבחינת Front-end:

- Enabling the user to configure the service, user credentials and cloud storage provide - ✓
- Notifies the user when needed - ✓

מבחינת Back-end:

- Manage multiple users and their credentials - ✓
- Register for notifications from the cloud storage API - ✓
- Process the events and detect - ✓
- Remediate when needed - ✓
- Notify - ✓

מבחינת נקודות בונוס:

- Detection accuracy - ✓
- Manage multiple cloud storage providers - ☹
- Add Windows 10 toast notifications - ☹
- Add rollback function for the user - ☹
- Protect SharePoint and OneDrive for Business - ✓

מימשנו את נקודות הבונוס הללו כיוון שדיוק הזיהוי הוא קריטריון חשוב לדעתנו בפרויקט ולכן היה חשוב לנו לממש אותו.

כמו כן, שילוב SharePoint ו-OneDrive for Business במערכת שלנו דרש מאיתנו לבקש הרשאות לגשת לנתונים האלה. מבחינת האלגוריתם לא היה שינוי. לפיכך, החלטנו לממש גם את בונוס זה.

סקיילביליות:

האלגוריתם רץ באופן Serverless וכיוון שהוא תחת שירותי ה-Azure functions של Microsoft, ריבוי משתמשים לא אמור להשפיע על הביצועים.

נקודות לשיפור:

Honeypots:

- המשתמש יכול לשנות את ה-Honeypots בטעות ולכן יקבל False-Positive. לכן, היינו רוצים להכניס בדיקות נוספות גם על ה-Honeypots שלאו דווקא מסתמכות על שינויו.
- ה-Honeypots אינם מופצים בצורה דינמית. לפיכך, כאשר משתמש מוחק Honeypot, הוא לא מוצב מחדש. כמו כן, כאשר משתמש יוצר תיקייה חדשה אז לא נוצר Honeypot בתיקייה החדשה.
- התוכן של ה-Honeypots קבוע כך שתוכנת כופר חכמה יכולה ללמוד את התכנים של קבצים אלה ולדלג עליהם בהתקפה שלה.
- ה-Honeypots לא נמחקים מה-OneDrive של המשתמש כאשר הוא מפסיק את השירות.

סריקת קבצים:

- גודל של קובץ לא בא לידי ביטוי בחישוב האנטרופיה שלו.
- קבצים דחוסים לא נפתחים לטובת חישוב האנטרופיה של כל קובץ שנמצא בתוך הקובץ הדחוס.
- פונקציית האנטרופיה עלולה לקחת המון זמן ולכן המשתמש יקבל את ההודעה על ההצפנה לא באופן מידי בעת ההתקפה אלא ב-delay.

הוראות להפעלת הפרויקט:

תחילה יש להעלות את פונקציית ה-Azure לפני שמעלים את האתר.

העלאת ה-Azure function:

1. צור Azure function בפורטל Azure
2. התקן Azure CLI במחשב
3. היכנס לתיקייה azure function בתיקיית הפרויקט הכללית BHKShield
4. הרץ את הפקודה func azure functionapp publish APP_NAME כאשר במקום APP_NAME מלא בשם הפונקציה שיצרת ב-1
5. שמור את כתובת הפונקציה (לדוגמא: https://rans.azurewebsites.net/api/httptrigger?code=Z6iRrVa6jqg3iFUuzAP==pwT3T8Z7qkVDJULYhrYTd1oeRuSPOTJop4A)

לפני שמעלים את התוכן של האתר ל-Azure, יש ליצור Web app בפורטל Azure.

שינוי ב-Web app לפני ההעלאה ל-Azure:

1. כנס לתיקייה azure website\webapp
2. בקובץ graph_helper.py יש לשנות את המשתנה url_notification בשורה 8 למחרוזת שהיא כתובת הפונקציה שקיבלת בסעיף 5 למעלה
3. בקובץ oauth_settings.yml יש לשנות את השדה redirect לתוכן:
https://YOUR_WEB_APP_NAME.azurewebsites.net/callback
כאשר יש להחליף את YOUR_WEB_APP_NAME בשם האתר שבחרת ליצור לפני שהתחלת את הסעיפים הללו

כעת, נעלה את האתר.

העלאת ה-Web app:

1. בפורטל Azure בכרטיסייה של ה-Web app, היכנס ללשונית Deployment center
2. בחר Local git להיות האופציה ל-Deployment
3. אתחל git בתיקייה והוסף את azure בתור remote
4. בצע git push
5. כנס לאתר והתחל להשתמש בשירות

תודות:

נרצה להודות לצוות הקורס שהנחה אותנו במהלך הסמסטר.

נרצה להודות ל-Microsoft שמימנו את הפרויקט ואת הגישה לענן Azure.

מקורות:

[1] Ransomware attacks: detection, prevention and cure by Ross Brewer

https://grades.cs.technion.ac.il/grades.cgi?cadgdhaca09eaf6fd8fa1f55d0d9f4+2+236499+Spring2019+ho/WCFiles/Ransomwareattacks%20detection%20prevention%20and%20cure%20Ross%20Brewer.pdf

[2] CryptoLock (and Drop It): Stopping Ransomware Attacks on User Data

https://www.cise.ufl.edu/~traynor/papers/scaife-icdcs16.pdf

[3] Azure Web App Service

https://azure.microsoft.com/en-us/services/app-service/web/

[4] Django framework

https://www.djangoproject.com/

[5] Azure SQL Database Service

https://azure.microsoft.com/en-in/services/sql-database/

[6] Azure Function Service

https://azure.microsoft.com/en-in/services/functions/

[7] Microsoft Graph Subscription

https://developer.microsoft.com/en-us/graph/graph/docs/api-reference/beta/resources/subscription

[8] Microsoft OAuth 2.0 API

https://docs.microsoft.com/en-us/azure/active-directory/develop/v1-protocols-oauth-code

[9] Microsoft Graph Delta Query

https://docs.microsoft.com/en-us/graph/delta-query-overview

[10] Shannon's Entropy Calculator

https://github.com/ambron60/shannon-entropy-calculator

[11] List of ransomware extensions and known ransom files created by Crypto malware

https://www.reddit.com/r/sysadmin/comments/46361k/list_of_ransomware_extensions_and_known_ransom/

[12] Shannon's Entropy Calculator

https://github.com/ambron60/shannon-entropy-calculator

BHKShield