בס"ד

Project Structure

הסבר על המחלקות ותפקידיהן

ProcessMonitor

המחלקה הראשית. תחום אחריותה הוא לתפעל את המוניטור, לתעד לקבצים את מצב המוניטור ולאבטח אותם. ע"מ לבצע משימות אלו, המחלקה משתמשת בספריות ובמחלקות עזר נוספות.

המחלקות והספריות שהמחלקה משתמשת בהן:

time, psutil, ProcessList, StatusLog, FilesHandler

ProcessList

המחלקה אחראית לטפל בקובץ "processList.txt". המחלקה תתעד לקובץ את כל התהליכים שנדגמו כל דגימה מחדש.

הספריות שהמחלקה משתמשת בהן:

datetime, os, stat

StatusLog

המחלקה מיועדת לטיפול בקובץ "Status_Log.txt". המחלקה תתעד לקובץ עבור כל דגימה אילו תהליכים חדשים התחילו לרוץ ואילו תהליכים הפסיקו לרוץ מאז הדגימה האחרונה.

הספריות שהמחלקה משתמשת בהן:

datetime, os, stat

FilesHandler

המחלקה יורשת מהמחלקה PatternMatchingEventHandler ששייכת לספריה watchdog ותגדיר את הפעולות שיש לבצע בעת שמתבצעים שינויים לא סבירים באחד מקבצי התיעוד שהמוניטור מייצר.

הספריות שהמחלקה משתמשת בהן:

time, whatchdog

הסבר על ספריות מיוחדות שהיו בשימוש:

Psutil

ספריה זו איפשרה לי לבצע דגימות של תהליכים שרצים. הספריה הייתה הכרחית למימוש של המוניטור. בתוך הספרייה יש מחלקה בשם Process ובתוכה מאפיינים חשובים כמו pid, name ועוד. במימוש המוניטור נעזרתי בשתי פונקציות שהספריה מספקת. האחת היא process_iter() שמחזירה אוסף של אובייקטים מסוג Process שמייצגים את כל התהליכים שרצים כרגע. השנייה היא pids() שמחזירה אוסף של כל ה-pid של התהליכים שרצים כרגע.

ספריה זו תומכת גם ב-windows וגם ב-linux.

Watchdog

ספריה זו סיפקה לי שירות חשוב בהגנה על קבצי התיעוד שהמוניטור כותב אליהם. השירות שהיא מספקת הוא שירות watch service. ספריה זו מאפשרת לי לדווח על מקרה של מחיקת הקבצים, העברתם ויצירתם. כמו כן ניתן לדווח גם על מקרה של שינוי בקבצים, אולם מכיוון שלא רציתי לקבל דיווח גם על שינויים שהתוכנה של המוניטור עצמה ביצעה בקבצים העדפתי לממש ולבדוק האם בוצעו שינויים בקבצים הללו באופן ידני ללא עזר בספריה זו.

לצורך כך דגמתי כל פעם את מועד השינוי האחרון שביצע המוניטור בקבצים ובין שינוי לשינוי שהמוניטור מבצע בדקתי שלא התבצע שינוי נוסף (ע"י גורם חיצוני...).

ספריה זו תומכת גם ב-windows וגם ב-linux.

Stat

בעזרת הקבועים של המחלקה S_IREAD, S_IRGRP, S_IROTH, S_IWUSR יכולתי לשנות את מאפייני הקובצים ולהפוך אותם ל-read only בין דגימה לדגימה. יש לציין ששינוי המאפיינים הוא מחסום קל למי שירצה לחבל בקבצים, אך גם זה עוזר לשמור על הקבצים (לדוגמא, לא יתבצעו בטעות שינויים לא רצויים, או כאשר הפורץ לא הספיק לסיים לערות את הקובץ והוא חזר להיות read only).

ספריה זו תומכת גם ב-windows וגם ב-linux.

שאר הספריות הן סטנדרטיות ותומכות גם הן ב-windows וב-linux (ביצעתי נסיון הרצה בשתי מערכות ההפעלה).

הסבר על אופן ההגנה על הקבצים:

הקבצים שהמוניטור מייצר יהיו מוגנים על ידי ההגבלה read only ודיווח למסך הconsole אודות שינוי, מחיקה, יצירה או העברה של הקבצים.

דרכים נוספות שחשבתי עליהן ע"מ להגן על הקבצים, אולם לא מצאתי דרך לממש אותן cross platform (או בכלל) בהתחשב לזמן שעמד ברשותי ביחס למורכבות שבמימוש ההגנות הללו:

- סיסמאות על הקבצים (לא מצאתי דרך פשוטה לעשות זאת בפייתון).
- העלאת הקבצים לאחר כל עידכון לגיבוי בענן עם שם משתמש וסיסמה (גיטהאב זו אפשרות טובה כי ניתן לעיין בגירסאות קודמות).
- הצפנת הקבצים. אך צריך לאפשר למורשים צפייה בקבצים... לכן צריך לממש ממשק משתמש שבהזנת סיסמה יוצגו על המסך הקבצים לאחר פיענוח (או פיענוח הקבצים לזמן מוגבל והצפנתם מחדש).
- להגדיר את הקבצים להיות במצב hidden.