A Project Report

On

Hashline - File Integrity Monitor

By

Yash Dagadkhair (BC212)
Akshay Dongare (BC218)
Jaydatta Patwe (BC223)

Under the guidance of

Dr. Girija Chiddarwar

Department of Computer Engineering

Marathwada Mitra Mandal’s College of Engineering

SAVITRIBAI PHULE PUNE UNIVERSITY

2023-2024

Marathwada Mitra Mandal’s College of Engineering

Department of Computer Engineering,Pune_51

“ येथे बहुतांचे हित ”

Date: 30/10/2023

CERTIFICATE

This is to certify that,

Yash Dagadkhair (BC212)
Akshay Dongare (BC218)
Jaydatta Patwe (BC223)

of class B.E Computer have successfully completed their project work on “Hashline-File Integrity Monitor’’ at MARATHWADA MITRA MANDAL'S COLLEGE OF ENGINEERING in the partial fulfillment of the Graduate Degree course in B.E Cyber Security & Digital Forensics Subject at the Department of Computer Engineering, in the academic Year 2023-2024 Semester – VII as prescribed by the Savitribai Phule Pune University.

Dr. Girija Chiddarwar	Dr. K. S. Thakre
Guide	Head of the Department
	(Department of Computer Engineering)

Acknowledgement

I feel great pleasure in expressing my deepest sense of gratitude and sincere thanks to my guide Dr. Girija Chiddarwar for her valuable guidance during the Project work, without which it would have been a very difficult task. I have no words to express my sincere thanks for the valuable guidance, extreme assistance and cooperation extended to all the **Staff Members** of my Department.

This acknowledgment would be incomplete without expressing my special thanks to **Prof. K. S. Thakre**, Head of the Department (Computer Engineering) for their support during the work.

I would also like to extend my heartfelt gratitude to my **Principal, Dr. V. N. Gohokar** who provided a lot of valuable support, mostly being behind the veils of college bureaucracy.

Last but not least I would like to thank all the Teaching, Non- Teaching staff members of my Department, my parents and my colleagues who helped me directly or indirectly for completing this Project successfully.

Name of Students

Yash Dagadkhair (BC212)

Akshay Dongare (BC218)

Jaydatta Patwe (BC223)

Contents

1. TITLE OF THE PROJECT

File Integrity Monitor

2. ABSTRACT

File Integrity Monitor (FIM) project aims to develop a comprehensive solution to protect critical systems and aid in post-incident analysis. This project combines cybersecurity and digital forensics to create a powerful tool for monitoring, detecting, and analyzing file system changes on computer systems, ensuring data integrity and aiding in the investigation of security breaches.

This project combines cybersecurity and digital forensics to create a powerful tool for monitoring, detecting, and analyzing file system changes on computer systems, ensuring data integrity and aiding in the investigation of security breaches.

3. INTRODUCTION

Problem Definition:
File Integrity Monitor (FIM) project aims to develop a comprehensive solution to protect critical systems and aid in post-incident analysis. This project combines cybersecurity and digital forensics to create a powerful tool for monitoring, detecting, and analyzing file system changes on computer systems, ensuring data integrity and aiding in the investigation of security breaches.

Introduction:
In an age dominated by digital information and the omnipresent threat of cyberattacks, the realm of cybersecurity and digital forensics has become more critical than ever. The File Integrity Monitor (FIM) project emerges as a beacon of innovation, bridging the domains of cyber defense and digital investigation. In this increasingly interconnected and perilous digital landscape, the need to preserve data integrity, detect and respond to security breaches in real-time, and conduct thorough post-incident analysis has never been more urgent. The FIM project aspires to address these pressing challenges, offering organizations a robust solution to safeguard their critical systems, data, and digital evidence. By combining advanced monitoring capabilities, forensic analysis tools, and compliance support, the project seeks to fortify the defenses of modern enterprises and empower them with the means to stand resilient in the face of cyber threats and digital breaches. This introduction serves as a gateway to the File Integrity Monitor project, which seeks to redefine the way organizations protect their digital assets and respond to the complex challenges of cybersecurity and digital forensics.

4. Problem Statement

Development and implementation of File Integration Monitoring system for evidence protection in digital forensics.

5. Technical Requirements

1. Software :

1.Python

2.Hashlib

3.Operating System

4.tkinter

5.custom tkinter

Algorithms Used:

1. SHA512: SHA-512, which stands for "Secure Hash Algorithm 512," is a cryptographic hash function that is part of the SHA-2 (Secure Hash Algorithm 2) family. It is designed to take an input (or message) and produce a fixed-size, 512-bit (64-byte) hash value, which is typically represented as a hexadecimal number.

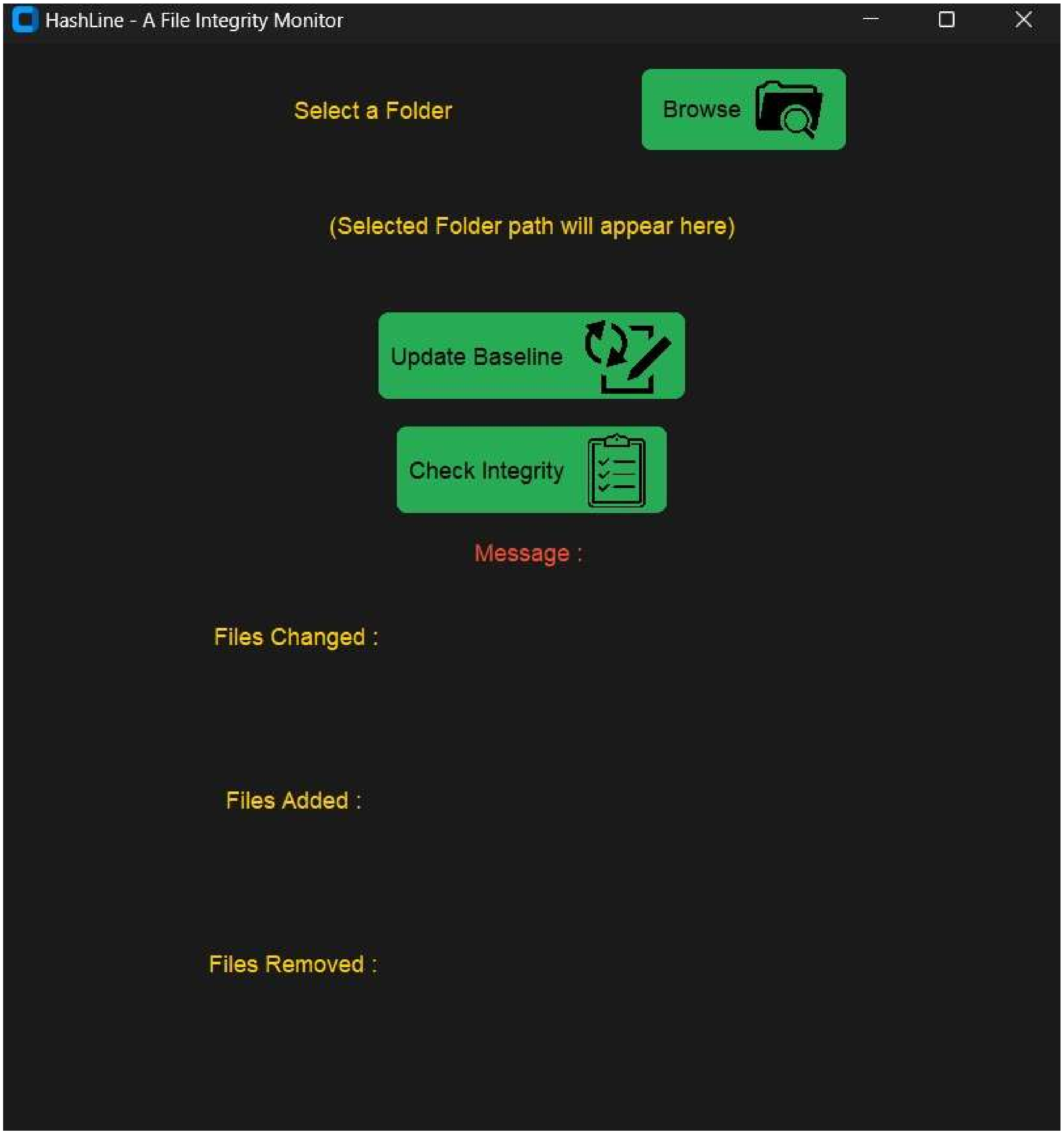
6. Sample Code

```
import glob

import hashlib

import os

import tkinter as tk

from tkinter.filedialog import askdirectory

import customtkinter as ctk


baselines = r"C:\Users\aksha\Desktop\Baselines" #Baseline.txt will be at this specified path

secure_path = ""


name_hash=""

baseline_path=""


files_changed = []

files_added = []

files_removed = []

files_all = []


spaces = "                                                  \n"


#Calculate hash from data in a file

def CalculateSha512Hash(file_name):

    # BUF_SIZE is totally arbitrary, change as per your requirement

    BUF_SIZE = 65536  # 65536 lets read stuff in 64kb chunks!

    sha = hashlib.sha512()


    with open(file_name,'rb') as file:

        while True:

            data = file.read(BUF_SIZE)

            if not data:

                break
            sha.update(data)
            # print("SHA: {0}".format(sha.hexdigest()))
        return sha.hexdigest()

#Calculate hash from name of a file
def CalculateNameHash(filename):
    md5 = hashlib.md5()
    md5.update(filename.encode())
    return md5.hexdigest()

#Updates baseline
def UpdateBaseline(dir,mode):
    if dir=="":
        label3.configure(text="Error : Folder not selected")

    elif os.path.isdir(baselines)==False:
        label3.configure(text="Message : Baselines Folder doesn't exists, so creating it")
        os.makedirs(baselines)
        label3.configure(text="Message : Updating Baseline...")
        UpdateBaselineHelper(dir,mode)
        label3.configure(text="Message : Updated Baseline Successfully")

    else:
        label3.configure(text="Message : Updating Baseline...")
        UpdateBaselineHelper(dir,mode)
        label3.configure(text="Message : Updated Baseline Successfully")

#Update Baseline Helper for [files in a folder] and [files in subfolders]
def UpdateBaselineHelper(dir,mode):

    global name_hash,baseline_path
    if(mode=='w'):
        name_hash = CalculateNameHash(dir)
        baseline_path = os.path.join(baselines,(name_hash+'.txt'))



    files = [os.path.abspath(f) for f in glob.glob(os.path.join(dir,'*')) if os.path.isfile(f)]
    with open(baseline_path,mode) as baseline:
        for f in files:
            hash = CalculateSha512Hash(os.path.join(dir,f))
            baseline.write(f)
            baseline.write("=")
            baseline.write(str(hash))
            baseline.write("\n")



    directories = [d for d in glob.glob(os.path.join(dir,'*')) if os.path.isdir(d)]
    for d in directories:
        UpdateBaselineHelper(d,'a')

#Returns dictionary containing keys as file name and values as their hashes
def getKeyHashesFromBaseline():
    global name_hash,baseline_path
    dict = {}

    with open(baseline_path,'r') as baseline:
        for line in baseline:
            key,value = line.split('=')
            dict[key] = value[:-1]

    return dict

#clears data in all 4 lists
def ClearData():
    files_changed.clear()
    files_added.clear()
    files_removed.clear()
    files_all.clear()

    fc.configure(text="Files Changed :"+spaces)
    fa.configure(text="Files Added :"+spaces)
    fr.configure(text="Files Removed :"+spaces)

#Calculates hashes and Checks with the baseline
def CheckIntegrity(dir,number):

    ClearData()#Clear data in all 4 lists

    if dir=="":
        label3.configure(text="Error : Folder not selected")

    else:
        CheckIntegrityHelper(dir,number)

        # print("Files Changed:",files_changed)
        # print("Files Added:",files_added)
        # print("Files Removed:",files_removed)



        fc.configure(text="Files Changed :"+spaces + '\n'.join(files_changed))
        fa.configure(text="Files Added : "+spaces + '\n'.join(files_added))
        fr.configure(text="Files Removed : "+spaces+ '\n'.join(files_removed))

        '''
        fc.configure(text=fc.set_text + '\n'.join(files_changed))
        fa.configure(text=fa.text + '\n'.join(files_added))
        fr.configure(text=fr.text + '\n'.join(files_removed))
        '''
        label3.configure(text="Message : Integrity Checked Successfully")

#Helper () for Check Integrity
def CheckIntegrityHelper(dir,number):
    global name_hash,baseline_path

    if(number):
        name_hash = CalculateNameHash(dir)
        baseline_path = os.path.join(baselines,(name_hash+'.txt'))
        try:
            with open(baseline_path,'r') as baseline:
                random=99
        except IOError:
            label3.configure(text='Error : Baseline file for specified folder not present')
            return

    files = [os.path.abspath(f) for f in glob.glob(os.path.join(dir,'*')) if os.path.isfile(f)]
    for x in files:
        files_all.append(x)
    dict = getKeyHashesFromBaseline()

    for f in files:
        #Checking for changed files
        temp_hash = CalculateSha512Hash(os.path.join(dir,f))
        if str(os.path.join(dir,f)) in dict.keys() and temp_hash!=dict[f]:
            files_changed.append(os.path.abspath(f).replace(os.path.abspath(folder),"."))

        #Checking for added files
        if str(os.path.join(dir,f)) not in dict.keys():
            files_added.append(os.path.abspath(f).replace(os.path.abspath(folder),"."))



    directories = [d for d in glob.glob(os.path.join(dir,'*')) if os.path.isdir(d)]
    for d in directories:
        CheckIntegrityHelper(d,0)

    if number==1:
        #checking for removed files
        for x in list(dict.keys()):
            if x not in files_all:
                files_removed.append(os.path.abspath(x).replace(os.path.abspath(folder),"."))




################################# GUI #################################

ctk.set_appearance_mode("dark")  # Modes: system (default), light, dark
ctk.set_default_color_theme("dark-blue")  # Themes: blue (default), dark-blue, green

#Some Variables
font_data = ("Raleway",14)
label_text_clr = "#FFCF00"
btn_fg_clr = "#27ab55"
btn_text_clr = "#000000"
btn_hover_clr = "#148f3f"
error_label_clr = "#E94F37"

folder=""

#initialising root window
root = ctk.CTk()
root.title("  HashLine - A File Integrity Monitor")
root.geometry("650x700")#(x,y)
# root.geometry("")
#root.resizable(False,True)
```

```
root.wm_iconphoto(False,tk.PhotoImage(file="images/window_icon.png"))

#label1 : Monitor folder
label1 = ctk.CTkLabel(master=root,
                      text="Select a Folder",
                      font=font_data,
                      text_color=label_text_clr)
label1.place(relx=0.35, y=40,anchor=tk.CENTER)#absolute placing



#browse button
def open_file():
    label3.configure(text="Message : ")
    label2.configure(text="(Selected Folder path will appear here)")
    global folder
    folder = askdirectory(parent=root, title="Choose a folder")
    if folder:
        label3.configure(text="Message : Folder Selected Successfully")
        label2.config(text=folder)
        ClearData()

browse_btn = ctk.CTkButton( master=root,
                            text="Browse",
                            image=tk.PhotoImage(file="images/browse.png").subsample(10,10),
                            compound=ctk.RIGHT,
                            command=open_file,
                            fg_color=btn_fg_clr,
                            text_color=btn_text_clr,
                            font=font_data,
                            hover_color=btn_hover_clr,
                            height=40,
                            width=125 )
browse_btn.place(relx=0.7,y=40,anchor=tk.CENTER)
```

```
#Label2 : Selected Folder Path
label2 = ctk.CTkLabel(master=root,
                text="(Selected Folder path will appear here)",
                wraplength=500,
                font=font_data,
                text_color=label_text_clr)
label2.place(relx=0.5,y=110,anchor=tk.CENTER)

#Button : Update Baseline
update_baseline_btn = ctk.CTkButton( master=root,
                text="Update Baseline ",
                image=tk.PhotoImage(file="images/updatebaseline.png").subsample(18,18),
                compound=ctk.RIGHT,
                command=lambda:UpdateBaseline(folder,'w'),
                fg_color=btn_fg_clr,
                text_color=btn_text_clr,
                font=font_data,
                hover_color=btn_hover_clr,
                height=40,
                width=150 )
update_baseline_btn.place(relx=0.5, y=190, anchor=tk.CENTER)

#Button : Check Integrity
check_integrity_btn = ctk.CTkButton( master=root,
                text="Check Integrity",
                image=tk.PhotoImage(file="images/checkintegrity.png").subsample(9,9),
                compound=ctk.RIGHT,
                command=lambda:CheckIntegrity(folder,1),
                fg_color=btn_fg_clr,
                text_color=btn_text_clr,
                font=font_data,
                                hover_color=btn_hover_clr,
                                height=40,
                                width=125 )
check_integrity_btn.place(relx=0.5, y=260, anchor=tk.CENTER)

#label3 : Message Label
label3 = ctk.CTkLabel( master=root,
                    text="Message : ",
                    font=font_data,
                    text_color=error_label_clr)
label3.pack(fill="both", expand=True)
label3.place(relx=0.5,y=310,anchor=tk.CENTER)



#label4 : Changed Files
fc = ctk.CTkLabel(master=root,text="Files Changed :"+spaces,font=font_data,text_color=label_text_clr)
fc.pack(fill="both", expand=True)
fc.place(relx=0.5,y=370,anchor=tk.CENTER)

#label4 : Added Files
fa = ctk.CTkLabel(master=root,text="Files Added : "+spaces,font=font_data,text_color=label_text_clr)
fa.pack(fill="both", expand=True)
fa.place(relx=0.5,y=470,anchor=tk.CENTER)

#label4 : Removed Files
fr = ctk.CTkLabel(master=root,text="Files Removed : "+spaces,font=font_data,text_color=label_text_clr)
fr.pack(fill="both", expand=True)
fr.place(relx=0.5,y=570,anchor=tk.CENTER)

root.mainloop()
```

7. Results/Visualizations



9. References

1. Python Documentation - https://docs.python.org/3/
2. Towards a Dynamic File Integrity Monitor through a Security Classification" by Zul Hilmi Abdullah et al. (2014)
3. A New Real-Time File Integrity Monitoring System for Windows-based Environments" by Imad I. Ali et al. (2010)