UNIVERZITET U NIŠU
ELEKTRONSKI FAKULTET

Forenzika aktivnosti na računaru

Seminarski rad
Studijski program: Računarstvo i informatika
Modul: Softversko inženjerstvo

Student:

Đorđe Petković, br. ind. 1614

Mentor:

prof. Bratislav Predić

Niš, Septembar 2024. godine

`

Sadržaj

`

Uvod

Računarska forenzika korišćenja je specijalizovana grana unutar šireg polja digitalne forenzike, koja se fokusira na analizu i istraživanje korisničkih aktivnosti na računarskim uređajima. Ova disciplina uključuje oporavak, ispitivanje i tumačenje podataka vezanih za interakcije korisnika sa različitim softverima i aplikacijama. Cilj je razumeti kako je računar korišćen, ko ga je koristio i u koje svrhe. U mnogim slučajevima, ove informacije mogu pomoći u određivanju radnji koje je korisnik preduzeo, praćenju obrazaca ponašanja ili uspostavljanju vremenske linije događaja [1]. Bilo da se radi o istraživanju kršenja korporativnih politika, kriminalnim aktivnostima ili neovlašćenom pristupu sistemima, forenzika korišćenja računara pruža ključne uvide koji mogu biti od presudnog značaja u pravnim slučajevima i incidentima kibernetičke bezbednosti.

Aplikacija razvijena kao deo ovog projekta osmišljena je da prikuplja detaljne podatke o interakciji korisnika sa različitim aplikacijama na njihovom računaru. Praćenjem koje aplikacije se koriste, koliko dugo i kada, aplikacija pruža granularan pregled aktivnosti pojedinca. Pored toga, prati unos sa tastature i miša kako bi pružila dodatne uvide u ponašanje korisnika, kao što je aktivno angažovanje tokom određenih perioda. Ovi podaci, kada se prikupe i analiziraju, mogu biti korisni u scenarijima koji uključuju praćenje produktivnosti, otkrivanje neovlašćenog korišćenja softvera ili čak identifikaciju potencijalnih unutrašnjih pretnji. Aplikacija beleži obrasce korišćenja i pruža vizuelnu reprezentaciju ovih ponašanja, nudeći dragocen alat za digitalne forenzičke istrage usmerene na otkrivanje načina na koji je sistem korišćen tokom vremena.

Osnove digitalne forenzike

Digitalna forenzika je nauka o prikupljanju, čuvanju i analizi digitalnih dokaza sa elektronskih uređaja na način koji je pravno prihvatljiv na sudu. Obuhvata niz tehnika koje se koriste za istraživanje incidenata kibernetičkog kriminala, neovlašćenog pristupa podacima i drugih digitalnih nepravilnosti. U savremenom svetu, gde većina aktivnosti ostavlja digitalni trag, digitalna forenzika igra ključnu ulogu u rešavanju zločina, identifikaciji zlonamernih aktivnosti i zaštiti osetljivih podataka [1]. Uključuje sistematski proces identifikacije, oporavka i dokumentovanja elektronskih podataka sa računara, pametnih telefona, mreža i cloud okruženja. Bilo da je reč o pravnim postupcima, korporativnim istragama ili odgovoru na incidente u kibernetičkoj bezbednosti, ova disciplina je od suštinskog značaja za osiguravanje da se digitalni dokazi sačuvaju bez izmena i da se mogu koristiti za izgradnju ili odbranu slučajeva.

`

Značaj digitalne forenzike značajno je porastao u poslednjim decenijama zbog sve veće zavisnosti od digitalnih tehnologija u svakodnevnom životu. Istorijski gledano, digitalna forenzika se pojavila kao grana tradicionalne forenzike kako su računari postajali sve prisutniji krajem 20. veka. U svojim ranim fazama, fokusirala se uglavnom na oporavak obrisanih datoteka ili istraživanje zločina povezanih sa računarima. Sa porastom ličnih računara 1990-ih, digitalna forenzika se proširila da obuhvati istraživanje pojedinačnih sistema, a do ranih 2000-ih razvila se da pokrije mobilne uređaje i umrežena okruženja. Ekspanzija pametnih telefona, rast cloud računanja i pojava Interneta stvari (IoT) dodatno su proširili obim digitalne forenzike. Istražitelji se sada suočavaju sa složenim ekosistemom međusobno povezanih uređaja, što zahteva sofisticirane tehnike za oporavak podataka iz sve raznovrsnijih izvora.

Digitalna forenzika se deli na nekoliko grana, od kojih svaka obrađuje specifične vrste dokaza i sistema. Računarska forenzika fokusira se na desktop i laptop računare, analizirajući sve, od hard diskova do datotečnih sistema, i široko se koristi u kako u policijskim, tako i u korporativnim istragama. Mobilna forenzika je brzo napredovala sa pojavom pametnih telefona i tableta, fokusirajući se na vađenje podataka kao što su tekstualne poruke, logovi poziva i GPS lokacije sa mobilnih uređaja [2]. Mrežna forenzika se bavi istragom podataka razmenjenih preko mreža, što je ključno za identifikaciju kibernetičkih napada, povreda podataka i neovlašćenog pristupa sistemima. Na kraju, cloud forenzika se pojavila kako je cloud skladištenje i usluge postalo svuda prisutno, zahtevajući jedinstvene metode za pristup podacima smeštenim u virtualizovanim okruženjima. Ove grane, iako su različite, često se preklapaju u savremenim istragama, odražavajući složenu prirodu digitalnih ekosistema i potrebu za sveobuhvatnim pristupima u otkrivanju i analizi digitalnih dokaza.

Grane digitalne forenzike

Evo nekih od najvažnijih grana digitalne forenzike [2]:

Računarska forenzika	Fokusira se na identifikaciju, oporavak i analizu podataka sa računara (desktop i laptop). Ovo uključuje ispitivanje hard diskova, obrisanih datoteka i sistemskih logova kako bi se otkrio neovlašćen pristup ili prikupili dokazi o kriminalnim aktivnostima.
Forenzika mobilnih uređaja	Specijalizovana je za vađenje i analizu podataka sa mobilnih uređaja poput pametnih telefona i tableta. Istražitelji često pribavljaju logove poziva, SMS poruke, GPS podatke, fotografije i

`

	korišćenje aplikacija kako bi pratili kretanja i komunikaciju.
Mrežna forenzika	Bavi se praćenjem i analizom mrežnog saobraćaja kako bi se otkrili i reagovali na kibernetičke napade ili neovlašćene prenose podataka. Mrežna forenzika je ključna u kibernetičkoj bezbednosti za identifikaciju napadača i zaštitu kompromitovanih sistema.
Cloud forenzika	Uključuje istraživanje podataka smeštenih na udaljenim serverima putem cloud usluga kao što su Google Drive, AWS ili Microsoft Azure. Budući da su cloud podaci distribuirani preko virtualizovanih okruženja, ova grana zahteva specijalizovane tehnike za pristup i analizu.
Forenzika memorije	Fokusira se na analizu volatilanog memorijskog prostora (RAM) u aktivnim sistemima kako bi se identifikovao malver, aktivni procesi ili tragovi kibernetičkih napada koji možda nisu zabeleženi na hard diskovima.
IoT forenzika	Istražuje podatke sa uređaja Interneta stvari (IoT), kao što su pametni sistemi za dom, nosivi uređaji i industrijski senzori. Ovi uređaji mogu sadržati ključne dokaze u slučajevima koji uključuju nadzor, prevaru ili fizičke zločine.

Korišćenje digitalne forenzike

Neki od slučajeva primene digitalne forenzike su [3]:

Kriminalne istrage	Digitalna forenzika pomaže policiji da pribavi podatke kao što su e-mailovi, fotografije i dokumenti koji se mogu koristiti kao dokazi u krivičnim slučajevima (npr. prevara, kibernetičko uznemiravanje, trgovina drogom).

`

Kibernetički napadi	Nakon provale u podatke, mrežna forenzika i forenzika memorije koriste se za praćenje staze napadača i utvrđivanje koji su podaci kompromitovani.
Krađa intelektualne svojine	Računarska forenzika može otkriti neovlašćeno kopiranje ili prenos osetljivih korporativnih informacija ili intelektualne svojine konkurentima.
Odgovor na incidente	Kompanije često koriste digitalnu forenziku da utvrde kako se cyber napad dogodio, ko su bili počinioci i kako sprečiti buduće provale.
Ponašanje zaposlenih	U korporativnim okruženjima, digitalna forenzika može se koristiti za istraživanje slučajeva zloupotrebe kompanijskih resursa, neovlašćenog pristupa ili curenja podataka od strane zaposlenih.
Građanske parnice	U slučajevima razvoda ili starateljstva, digitalna forenzika može pribaviti dokaze sa ličnih uređaja koji mogu uticati na ishod pravnih postupaka.

Digitalni dokazi u forenzici

Digitalni dokazi se odnose na sve podatke koji su sačuvani ili preneseni u digitalnom obliku i koji se mogu koristiti u pravnim istragama. Kako tehnologija postaje ključni deo svakodnevnog života, digitalni dokazi su dobili na značaju u oblastima kao što su policijske istrage, privatne istrage i kibernetička bezbednost. Ovo može da obuhvati jednostavne datoteke i e-mailove, kao i složenije logove, šifrovane podatke i obrisane informacije koje zahtevaju specijalizovane alate i tehnike za oporavak i analizu [1].

`

Tipovi digitalnih dokaza

E-mailovi i logovi komunikacije	Digitalna forenzika često uključuje pribavljanje e-mailova, logova čatova i komunikacije na društvenim mrežama. Ovi podaci su ključni u istraživanju zločina kao što su prevara, uznemiravanje i kibernetičko uznemiravanje.
Sistemski logovi i podaci o aktivnostima	Sistemski logovi, poput korisničkih aktivnosti ili informacija o prijavljivanju, mogu pružiti detaljne uvide u to kada i kako je sistem ili nalog bio pristupljen. Ovo može biti posebno korisno za praćenje neovlašćenog pristupa ili zlonamernih aktivnosti.
Datoteke i dokumenti	Važni digitalni dokumenti, fotografije ili medijski fajlovi mogu poslužiti kao ključni dokazi u krivičnim slučajevima, kao što su prevara, kršenje autorskih prava ili ilegalna distribucija.
Šifrovani podaci	Jedan od najvećih izazova u digitalnoj forenzici je suočavanje sa šifrovanim podacima. Kriminalci često koriste šifrovanje kako bi prikrili kompromitujuće informacije. Dešifrovanje ovih podataka zahteva specijalizovane tehnike.
Obrisani ili skrivene podatke	Forenzički stručnjaci često mogu da oporave obrisane ili skrivene podatke koristeći alate za oporavak ili duboku forenzičku analizu, što može biti ključno za otkrivanje pokušaja skrivanja informacija.

U kontekstu ove aplikacije, koja prati aktivne aplikacije, beleži klikove mišem i nadgleda pritiske na tastaturu, podaci koje generiše mogli bi poslužiti kao dokazi iz sistemskih logova—slično tome kako forenzički stručnjaci ispituju korisničke aktivnosti da bi pratili obrasce ponašanja. Na primer, mogli bi se koristiti u slučajevima kada je potrebno analizirati radno vreme korisnika ili digitalne navike. Ova vrsta dokaza takođe može biti korisna u identifikaciji unutrašnjih pretnji ili pružanju uvida u istrage koje se oslanjaju na vreme, kao što su periodi kada su se zlonamerne aktivnosti dogodile. Iako nije forenzička u smislu oporavka podataka, ova aplikacija doprinosi analizi ponašanja u digitalnoj forenzici.

`

Čuvanje ponašanja korisnika svodi se na svakodnevno kreiranje pdf dokumenta sa analizom upotrebe računara u određenom vremenskom periodu (na primer u toku radnog vremena), vizualizacijom tih podataka pomoću pie grafa za procentualno korišćenje aplikacije, kao i grafa koji pokazuje odnos aktivnosti na računaru (klikovi miša i tastature) u određenim vremenskim intervalima.

Ova analiza korišćena računara bi se na kraju svakog intervala korišćenja kao attachment osobi koja je zadužena za analizu ponašanja korisnika (pr. Radnika jedne kompanije). U ovom rešenju, šalje se email na određenu email adresu, koja se može dinamički podesiti pomoću **env** fajla.

Izazovi u prikupljanju digitalnih dokaza

Prikupljanje digitalnih dokaza predstavlja nekoliko tehničkih i pravnih izazova [2]:

Šifrovanje	Mnogi uređaji i sistemi za skladištenje podataka su zaštićeni šifrovanjem, što otežava istražiteljima pristup sadržaju. U kontekstu ovog aplikacije, ako bi podaci o korišćenju bili šifrovani, to bi odražavalo stvarne scenarije u kojima istražitelji treba da pristupe šifrovanim logovima ili datotekama.
Brisanje podataka	Korisnici često pokušavaju da obrišu digitalne tragove. Forenzički stručnjaci koriste napredne alate za oporavak da bi pribavili obrisane podatke. Slično tome, aplikacija bi mogla biti proširena da prati kada je aplikacija zatvorena ili kada su njeni logovi korišćenja obrisani, što bi služilo kao forenzički dokaz o pokušajima skrivanja aktivnosti.
Problemi sa jurisdikcijom	Sa podacima koji su smešteni na različitim lokacijama (posebno u cloud uslugama), identifikacija koje zakone primeniti može biti teška. Dok se ova aplikacija fokusira na lokalno praćenje korišćenja, korišćenje aplikacija u cloud-u postavljalo bi dodatna pitanja jurisdikcije za forenzičku analizu.

U kontekstu ove aplikacije, logovi vremena korišćenja aplikacija, klikovi mišem i pritisci na tastaturu mogli bi biti povezani sa procesima lanca čuvanja dokaza. Baš kao što forenzički dokazi treba da se čuvaju bezbedno od trenutka prikupljanja do sudskih postupaka, podaci iz ove aplikacije treba da budu sigurno skladišteni i dostupni, obezbeđujući njihovu integritet za

`

potencijalnu pravnu upotrebu. Lanac čuvanja podrazumeva dokumentovanje ko je pristupio podacima i kada—ova aplikacija bi mogla biti deo toga timetamp-ove u logovima, pomažući pri autentifikaciji obrazaca korišćenja.

Prikupljanje podataka

U ovom poglavlju biće predstavljeni načini prikupljanja podataka na MacOs i na Windows sistemu.

Prikupljanje podataka na MacOs

U macOS, praćenje korišćenja aplikacija funkcioniše tako što nadgleda sistemske procese i aktivne prozore aplikacija. Operativni sistem održava detaljne informacije o svakoj pokrenutoj aplikaciji putem upravljanja procesima i WindowServer-a, koji je odgovoran za prikazivanje i upravljanje prozorima na ekranu. Kada se prebacuje fokus između aplikacija, macOS prepoznaje ovu promenu i dovodi prozor nove aplikacije u fokus, što sistem beleži kao prozor koji je najaktuelniji ili aktivan.

Napravljena aplikacija koristi sposobnost macOS-a da upravlja ovim procesima i prikuplja informacije o trenutno aktivnoj aplikaciji. Koristeći AppleScript ili slične sistemske komande, možete upitati macOS koja aplikacija ima fokus u bilo kom trenutku. Na primer, komanda kao što je

```
osascript -e 'tell application "System Events" to get name of first application process whose frontmost is true'
```

vraća ime najaktuelnijeg procesa aplikacije. Ova komanda komunicira sa procesom System Events u macOS-u, koji prati fokus aplikacija i druge sistemske događaje vezane za interakcije korisnika.

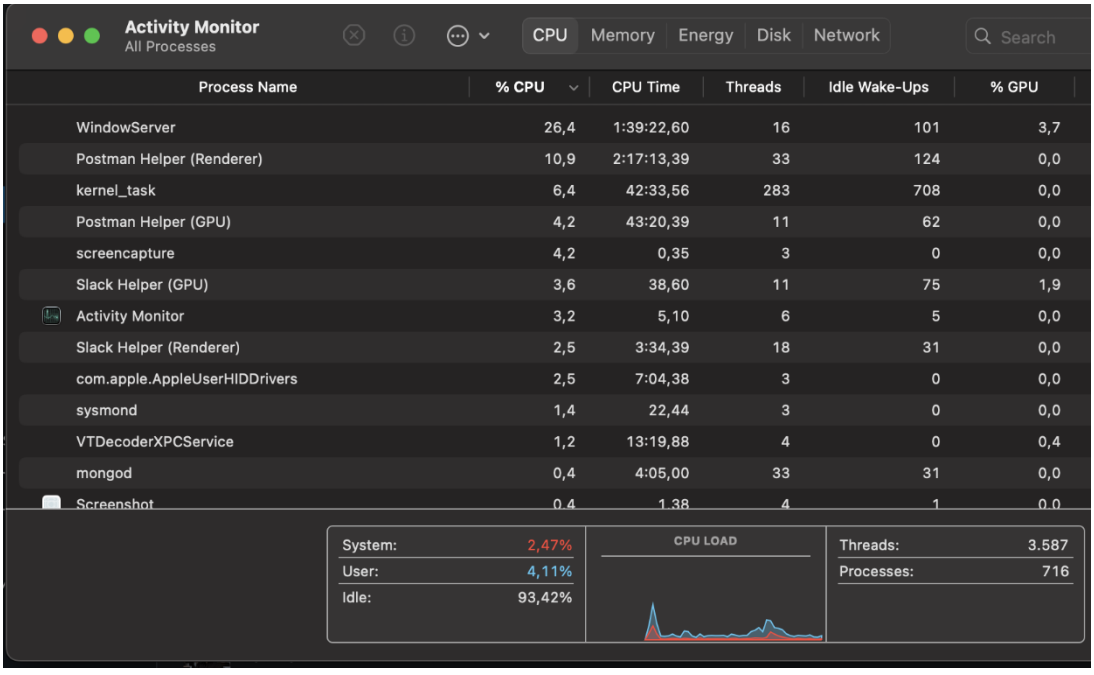
Ključ za praćenje korišćenja aplikacija je redovno ispitivanje sistema kako bi se proverile promene u aktivnom prozoru. Radi efikasnijeg pozadinskog rada aplikacije, koristi se određeni vremenski interval nakon koga se izvršava ova AppleScript-a (na primer, svakih 10 sekundi) da bi se proverilo koja aplikacija je trenutno aktivna. Ovo pruža log korisničkog ponašanja, beležeći koliko dugo je svaka aplikacija u fokusu, dajući uvid u obrasce korišćenja aplikacija. Takvo praćenje je relevantno u digitalnoj forenzici, jer se može koristiti za

`

nadgledanje aktivnosti korisnika, otkrivanje potencijalne zloupotrebe sistema ili pružanje podataka za procene produktivnosti.

Na primer, ako korisnik kodira u Visual Studio Code i često prebacuje na Google Chrome, ova aplikacija beleži svako prebacivanje i prati koliko dugo svaka aplikacija ostaje u fokusu. Akumulacijom ovih podataka tokom određenog vremenskog perioda, mogu se analizirati obrasci da bi bolje razumeo kako je sistem korišćen. Ova analiza može biti korisna prilikom rekonstrukcije digitalnog traga korisnika tokom analize performansi radnika, ili prilikom istraživanja nekog nelegalnog/nedoličnog ponašanja.

U macOS-u, svaka aplikacija je predstavljena procesom, a aplikacija Activity Monitor (prikazana ispod) prikazuje sve pokrenute procese u sistemu. Pristupanjem ovim procesima, ova aplikacija pribavlja relevantne informacije kao što su ime aktivne aplikacije i prati njen vremenski period korišćenja.



Slika 1.0 activity monitor

Ovo praćenje zasnovano na procesima osigurava da aplikacija može tačno zabeležiti korisničke aktivnosti. U scenarijima gde korisnik može pokušati da prikrije svoje tragove ili ako se postavlja pitanje korišćenja neovlašćenog softvera, podaci o korišćenju aplikacija mogu biti preuzeti i analizirani. Sposobnost praćenja korišćenja aplikacija, zajedno sa

`

aktivnostima tastature i miša, čini ovu aplikaciju potencijalnim alatom za razumevanje obrazaca korišćenja sistema.

Prikupljanje podataka na Windows-u

U Windows-u, prikupljanje podataka o korišćenju aplikacija prvenstveno se oslanja na interakciju sa sistemskim procesima i praćenje aktivnog prozora kako bi se odredilo koja aplikacija je u fokusu. Operativni sistem pruža nekoliko alata i API-ja koji olakšavaju ovaj proces, omogućavajući pristup informacijama na sistemskom nivou o pokrenutim aplikacijama i njihovim povezanim procesima.

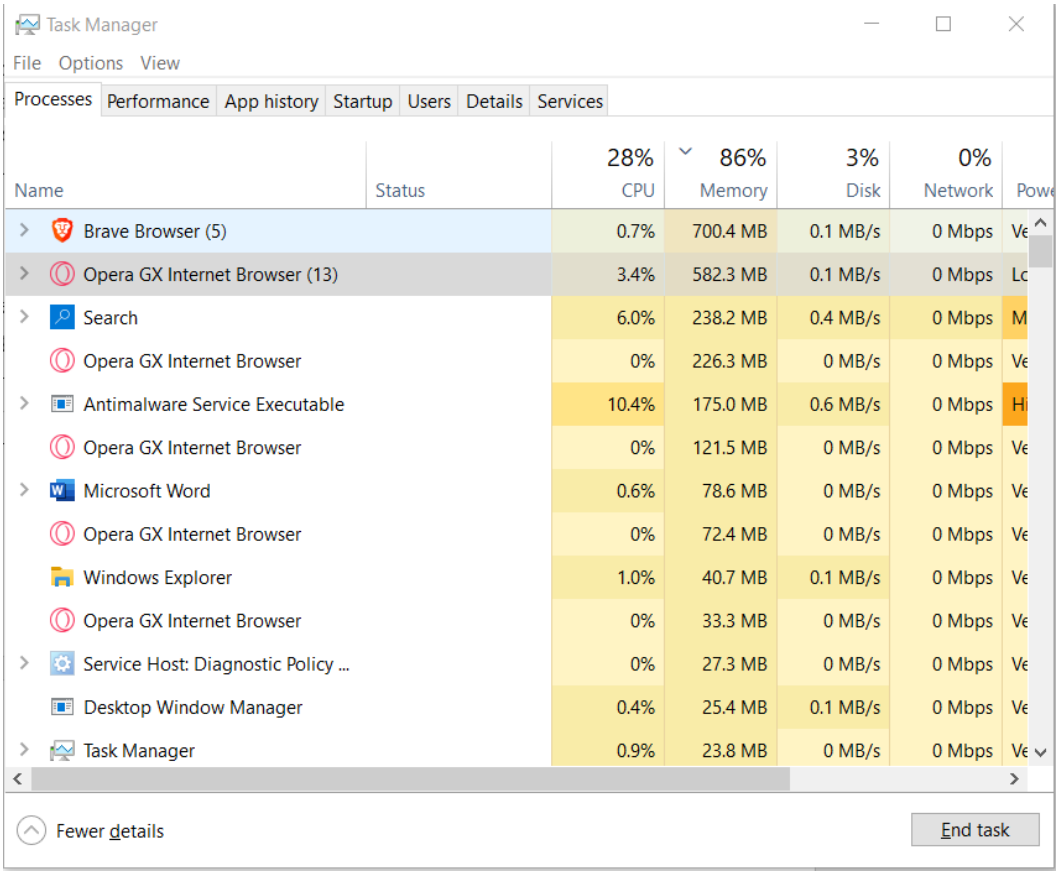
Da bi prikupila podatke o tome koje aplikacije se trenutno koriste, ova aplikacija se povezuje sa Windows API-jem, koji uključuje funkcije kao što su **GetForegroundWindow()** i **GetWindowText()**. Funkcija **GetForegroundWindow()** vraća identifikator (handle) prozora koji je trenutno u fokusu, dok **GetWindowText()** preuzima naslov aktivnog prozora. Ove funkcije omogućavaju aplikaciji da identifikuje koja aplikacija se trenutno koristi i beleži relevantne detalje kao što su ime aplikacije, naslov prozora i ID procesa.

Na primer, ako korisnik aktivno radi u web pregledaču poput Google Chrome-a, pozivajući GetForegroundWindow() dobija se identifikator prozora Chrome-a, a GetWindowText() će pružiti naslov prozora, kao što je naziv trenutno otvorene kartice. Periodičnim ispitivanjem ovih informacija, aplikacija može pratiti koliko dugo korisnik provodi u svakoj aplikaciji, prebacujući se između aplikacija po potrebi.

Aplikacija takođe može koristiti Windows Management Instrumentation (WMI) servis da bi postavila upite o detaljima sistema vezanim za pokrenute procese. WMI pruža dublji nivo pristupa informacijama povezanim sa procesima, kao što su ID procesa (PID), korišćenje memorije i CPU. Ove informacije su od vitalnog značaja za identifikaciju koji procesi odgovaraju kojim aplikacijama.

Windows strukturira svoje pokrenute procese na način da svaka aplikacija ili servis ima odgovarajući proces ili skup procesa. Ovo je vidljivo u Task Manager-u, gde je svaka pokrenuta aplikacija navedena zajedno sa povezanim sistemskim resursima kao što su korišćenje CPU-a i memorije. Korišćenjem sličnih sistemskih poziva, aplikacija može programatski prikupiti ove podatke, pružajući detaljan log korišćenja aplikacija tokom vremena.

`



Slika 1.1 task manager

Evo pojednostavljenog primera kako funkcioniše prikupljanje podataka:

Praćenje aktivnog prozora: Svakih 10 sekundi, aplikacija poziva GetForegroundWindow() da proveri koja aplikacija je u fokusu.

Prikupljanje naslov prozora: Aplikacija koristi GetWindowText() da dobije naslov prozora, koji često uključuje ime aplikacije i relevantne detalje (kao što je naziv datoteke ili naslov web stranice).

Ova metoda praćenja aktivnog prozora i beleženja informacija o procesima je veoma efikasna, osiguravajući da aplikacija može neprekidno da beleži korisničke aktivnosti u smislu korišćenja aplikacija. Ona se besprekorno uklapa u Windows-ov sistem upravljanja procesima, omogućavajući detaljno praćenje aplikacija bez značajnog opterećenja na performanse sistema.

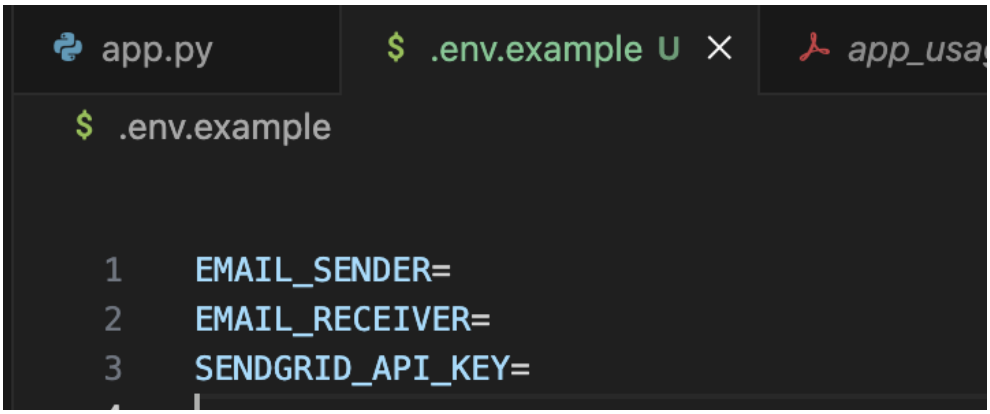
`

Aplikacija

U ovom poglavlju biće predstavljen rad aplikacije, sa svim njenim delovima.

Podešavanje okruženja (env)

Da bi aplikacija bila funkcionalna, potrebno je imati određene env promenjive. U fajlu **.env.example** moguće je videti potrebne promenjive



Slika 2.0 env example

Nakon toga, u aplikaciji se učitavaju vrednosti za **env**, što se kasnije koristi za slanje mailova zaduženom licu (pr Hr u kompaniji, ili istražiteljima)

```
load_dotenv()

EMAIL_USERNAME = os.getenv("EMAIL_SENDER")
EMAIL_RECEIVER = os.getenv("EMAIL_RECEIVER")
SENDGRID_API_KEY = os.getenv('SENDGRID_API_KEY')
```

Slika 2.1 učitavanje env promenjivih

`

Prikupljanje podataka o korišćenju aplikacija

Prikupljanje podataka o korišćenju aplikacija implementirano je na Windows i MacOs sistemima. Prikupljanje podataka o aplikacijama je različito, te se u aplikaciji proverava koji je sistem, i na osnovu toga se prikupljaju podaci.

MacOs

Zbog kompleksnosti MacOs, i zbog nedovoljno konkretnih podataka dobijenih pomoću biblioteka koje su implementirane u Python-u, u aplikaciji se koriste AppleScript-e za dobijanje potrebnih aplikacija.

Sama AppleScript-a nam daje output u obliku naziva aplikacije, i naslova prozora, koji se dalje obrađuje da bi se sačuvao u nasim strukturama za čuvanje podataka o sesiji. Takođe, neke aplikacije kao što je Visual Studio Code, imaju drugačije nazive u samom MacOS sistemu, što od nas zahteva dodatno mapiranje.

Postojao je način da se konkretno iz podataka sa MacOs-a izvadi pravo ime aplikacije, ali zbog kompleksnosti i zbog davanja ne uvek konzistentnih informacija, nije implementirano. Takođe, mapiranje bi trebalo biti odrađeno za neke dodatne aplikacije, nakon što se nepoklapanje pronađe.

`

```
def get_active_window():
    current_os = platform.system()

    if current_os == "Darwin":
        script = '''
        tell application "System Events"
            set frontApp to first application process whose frontmost is true
            set appName to name of frontApp
            set windowTitle to ""
            if appName is "Electron" then
                try
                    set windowTitle to name of first window of frontApp
                end try
            end if
            return {appName, windowTitle}
        end tell
        '''
        try:
            result = subprocess.run(['osascript', '-e', script], capture_output=True, text=True)
            output = result.stdout.strip()
            if output:
                values = output.split(", ")
                app_name = values[0].replace(",", "")
                if app_name.__contains__("Electron"):
                    return "Visual Studio Code"
                return app_name
            else:
                return "No active application found"
        except subprocess.CalledProcessError as e:
            return f"Error: {e}"
```

Slika 2.2 prikupljanje podataka na MacOs

Windows

Kod Windows sistema, postoje biblioteke koje nam pomažu u radu sa Task manager-om, što dosta pojednostavljuje proces prikupljanja podataka o samom sistemu.

Kao što je objašnjeno u nekom od prethodnih poglavlja, koristi se **GetActiveWindow()** funkcija, koja nam daje sve potrebne informacije o samoj aplikaciji.

`

```
elif current_os == "Windows":
    import pygetwindow as gw
    window = gw.getActiveWindow()
    if window is not None:
        return window.title
    return None

else:
    return "Unsupported OS"
```

Slika 2.3 prikupljanje podataka na Windows

Prikupljanje podataka o korišćenju ulaznih uređaja

Za prikupljanje podataka o aktivnosti perifernih uređaja (miša i tastature) koristi se biblioteka **pynput**, koja nam pruža mogućnost korišćenja funkcija **on_click** i **on_press** što nam daje informacije o klikovima na tastaturi i mišu.

U daljem razvoju aplikacije, ovaj deo bi mogao da se unapredi, da nam da dodatne informacije o samoj aktivnosti na ulaznim uređajima.

```
def on_click(x, y, button, pressed):
    if pressed:
        current_hour = get_current_hour()
        mouse_clicks, keyboard_presses = activity_per_hour[current_hour]
        activity_per_hour[current_hour] = (mouse_clicks + 1, keyboard_presses)

def on_press(key):
    current_hour = get_current_hour()
    mouse_clicks, keyboard_presses = activity_per_hour[current_hour]
    activity_per_hour[current_hour] = (mouse_clicks, keyboard_presses + 1)
```

Slika 2.4 prikupljanje podataka u korišćenju ulaznih uređaja

`

Vizualizacija podataka

Vizualizacija podataka o korišćenju aplikacija

Za vizualizaciju informacija o korišćenju aplikacija pravi se PieChart, pomoću biblioteke **plt**. Na njemu je prikazano procentualano vreme korišćenja svake aplikacije.

```
def generate_pie_chart(usage_data):
    labels = list(usage_data.keys())
    times = [info['total_time'] for info in usage_data.values()]

    plt.figure(figsize=(6, 6))
    plt.pie(times, labels=labels, autopct='%1.1f%%', startangle=140)
    plt.title('App Usage Breakdown')

    # Save as an image
    pie_chart_path = './REPORTS/app_usage_pie_chart.png'
    plt.savefig(pie_chart_path)
    plt.close()
    return pie_chart_path
```

Slika 2.5 kreiranje pie grafa

Vizualizacija podataka o ulaznim uređajima

Za vizualizaciju ovih podataka, korisi se graf, koji na jednoj osi ima sati (svaki sat u toku intervala), dok na drugoj ima broj klika mišem i broj unosa na tastaturi. Takođe se koristi biblioteka **plt**.

`

```python
def generate_bar_graph(activity_data):
    hours = list(activity_data.keys())

    mouse_clicks = [activity_data[hour][0] for hour in hours]
    keyboard_presses = [activity_data[hour][1] for hour in hours]

    plt.figure(figsize=(8, 6))
    width = 0.35
    plt.bar(hours, mouse_clicks, width, label='Mouse Clicks', color='blue')
    plt.bar([h + width for h in hours], keyboard_presses, width, label='Keyboard Presses', color='orange')

    plt.xlabel('Hour of the Day')
    plt.ylabel('Number of Actions')
    plt.title('Mouse Clicks and Keyboard Presses per Hour')      You, last week • fix: update app with graphs
    plt.legend()

    bar_graph_path = './Reports/activity_bar_graph.png'
    plt.savefig(bar_graph_path)
    plt.close()
    return bar_graph_path
```

Slika 2.6 kreiranje regularnog grafa

Generisanje PDF-a

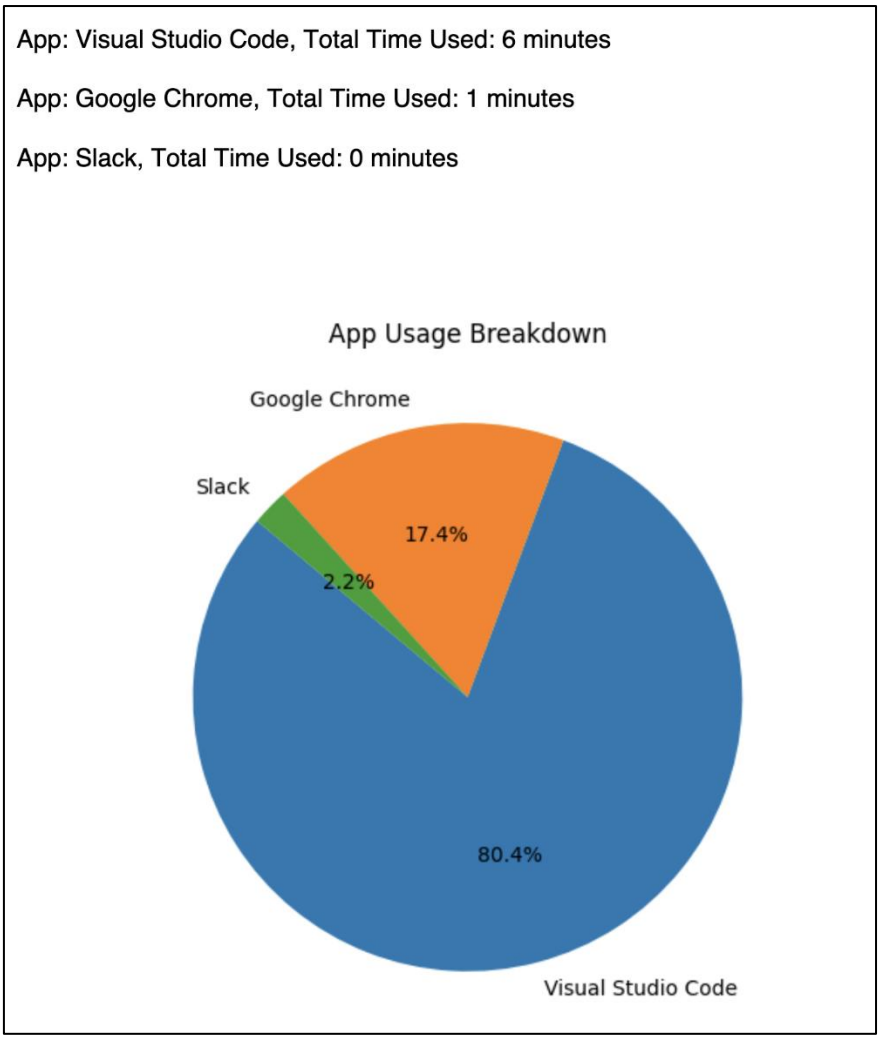
U prvom delu u generisanom PDF-u nalaze se informacije o korišćenju aplikacija.

```python
def generate_pdf_report():
    pdf = FPDF()
    pdf.set_auto_page_break(auto=True, margin=15)
    pdf.add_page()

    pdf.set_font("Arial", size=12)
    pdf.cell(200, 10, txt="Application Usage Report", ln=True, align='C')
    pdf.ln(10)

    # Write the log details to the PDF
    for app, info in usage_log.items():
        total_seconds = info['total_time']
        total_minutes = total_seconds // 60
        hours = total_minutes // 60
        minutes = total_minutes % 60

        if hours > 0:
          output = f"App: {app}, Total Time Used: {hours} hours and {minutes} minutes"
        else:
          output = f"App: {app}, Total Time Used: {minutes} minutes"      You, last week • add application
        pdf.cell(200, 10, txt=output, ln=True)

    pie_chart_path = generate_pie_chart(usage_log)
    pdf.ln(10)
    pdf.image(pie_chart_path, x=10, y=None, w=150)
```

Slika 2.7 generisanje pdf-a. Deo za korišćenje aplikacija

`

Nakon čuvanja PDF-a, taj deo izgleda ovako.



Slika 2.8 izgled pdf-a. Deo za korišćenje aplikacija

U drugom delu, nalaze se informacije o klikovima na perfiernim uređajima.

```
pdf.ln(10)
pdf.cell(200, 10, txt="Mouse Clicks and Keyboard Presses by Hour:", ln=True)
for hour, (mouse_clicks, keyboard_presses) in activity_per_hour.items():
    if mouse_clicks > 0 or keyboard_presses > 0:
        pdf.cell(200, 10, txt=f"{hour}:00 - {hour + 1}:00 -> Mouse Clicks: {mouse_clicks}, Keyboard Presses: {keyboard_presses}", ln=True)

bar_graph_path = generate_bar_graph(activity_per_hour)
pdf.add_page()
pdf.image(bar_graph_path, x=10, y=None, w=180)

timestamp = datetime.now().strftime("%Y-%m-%d")
pdf_output = f"app_usage_report_{timestamp}.pdf"
pdf_file_path = os.path.join('Reports', pdf_output)
pdf.output(pdf_file_path)
print(f"PDF report saved as {pdf_output}")
```

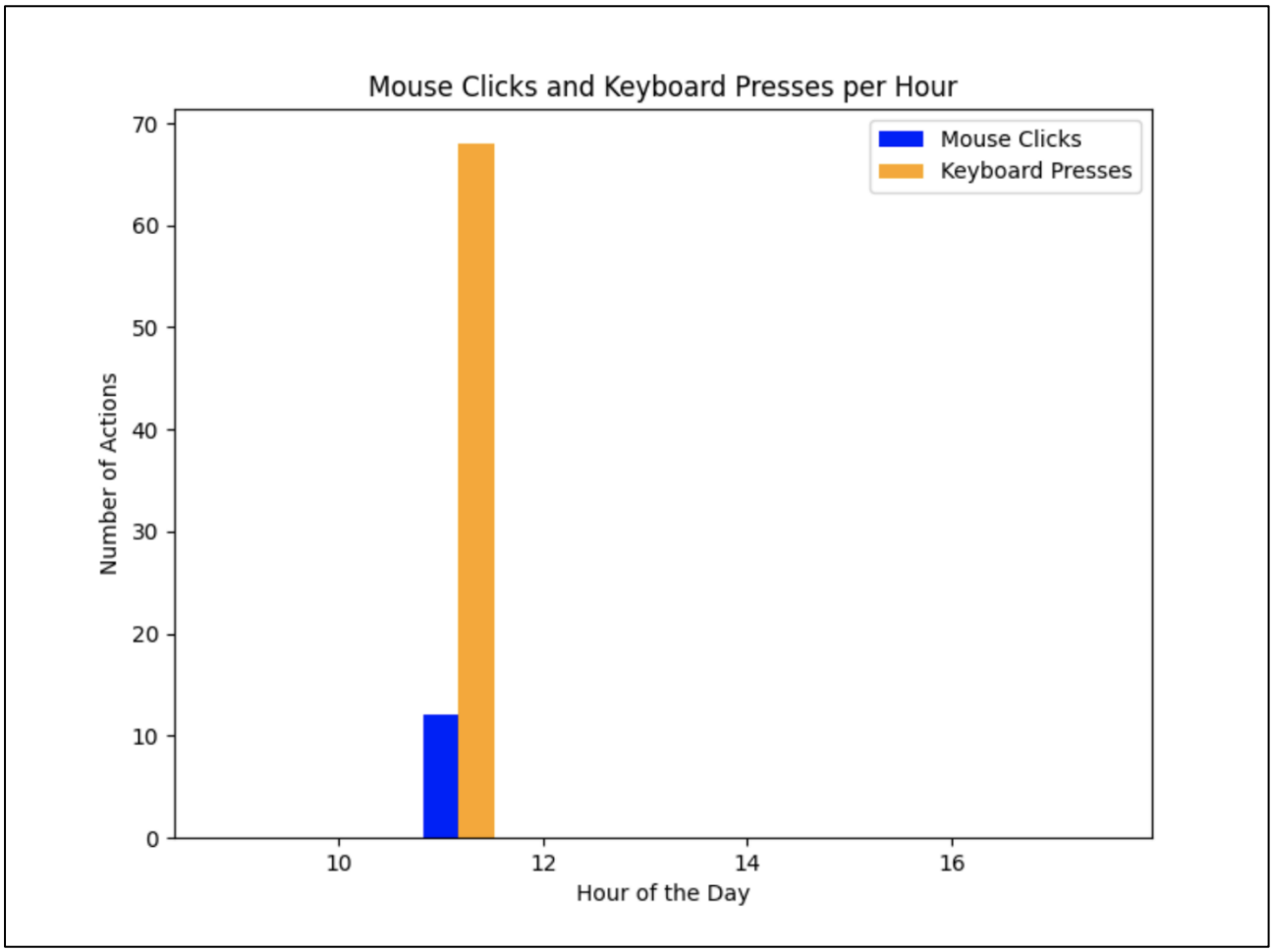
Slika 2.9 generisanje pdf-a. Deo za korišćenje ulaznih uređaja

`

U generisanom pdf-u, to izgleda ovako.

Mouse Clicks and Keyboard Presses by Hour:

11:00 - 12:00 -> Mouse Clicks: 12, Keyboard Presses: 68



Slika 2.10 izgled pdf-a. Deo za korišćenje ulaznih uređaja

Slanje email-a

Za slanje mail-ova koristi se Sendgrid [5], servis za slanje mail-ova. U mailu, pored naslova i kratke poruke, dodaje se generisani pdf za današnji dan, i šalje korisniku koji je postavljen u **env** okruženju.

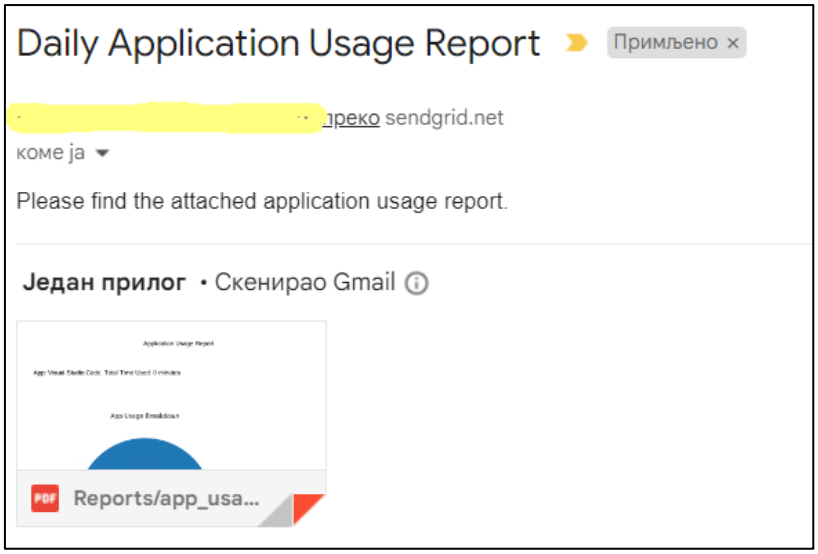
`

```python
def send_email(pdf_filename):
    print("""Sending the PDF report via email using SendGrid.""")

    with open(pdf_filename, 'rb') as f:
        pdf_data = f.read()
        encoded_pdf = base64.b64encode(pdf_data).decode()  # Encode as base64 string

    message = Mail(
        from_email=EMAIL_USERNAME,
        to_emails=EMAIL_RECEIVER,
        subject="Daily Application Usage Report",
        plain_text_content="Please find the attached application usage report."
    )

    attachment = Attachment(
        FileContent(encoded_pdf),
        FileName(pdf_filename),
        FileType('application/pdf'),
        Disposition('attachment')
    )

    message.attachment = attachment

    try:
        ssl_context = ssl.create_default_context()
        ssl_context.check_hostname = False
        ssl_context.verify_mode = ssl.CERT_NONE

        urllib.request.urlopen = urllib.request.build_opener(
            urllib.request.HTTPSHandler(context=ssl_context)
        ).open

        sg = SendGridAPIClient(os.environ.get('SENDGRID_API_KEY'))
        response = sg.send(message)
        print(f"Email sent successfully! Status code: {response.status_code}")
    except Exception as e:
        print(f"Failed to send email: {e}")
```

Slika 2.11 slanje mail-a

Dobijeni mail izgleda ovako.

`



Slika 2.12 izgled mail-a

Pokretanje aplikacije

Pri pokretanju aplikacije, startuju se listeneri za miša i tastaturu, i proverava se da li je trenutno vreme u opsegu radnog vremena (može se promeniti). Ukoliko jeste, pokreće se funkcija za praćenje aktivnosti korisnika, koja se automatski zaustavlja na kraju vremenskog intervala (u ovom slučaju radnog vremena)

Ukoliko se aplikacija manualno prekine, svakako ce generisati pdf i poslati email sa tim attachment-om. Nakon završetka aplikacije, gase se listeneri za praćenje miša i tastature.

```
if __name__ == "__main__":
    # Setting up listeners for mouse and keyboard tracking
    mouse_listener = mouse.Listener(on_click=on_click)
    keyboard_listener = keyboard.Listener(on_press=on_press)

    mouse_listener.start()
    keyboard_listener.start()

    try:
        while is_within_work_hours():
            log_application_usage()
    except KeyboardInterrupt:
        # if stopped manually, still generate Pdf report
        generate_pdf_report()

    # Stopping listeners
    mouse_listener.stop()
    keyboard_listener.stop()
```

Slika 2.13 pokretanje aplikacije

`

Problemi u korišćenju forenzike korišćenja računara

Postoji nekoliko mogućih dodataka koji bi mogli poboljšati korisnost aplikacije i otkloniti trenutna ograničenja. Jedno značajno poboljšanje bilo bi dodavanje prilagodljivih podešavanja privatnosti koja bi korisnicima omogućila da odrede koju vrstu podataka žele da prikupljaju i kada se praćenje vrši. Ovaj nivo kontrole bi mogao ublažiti neke probleme u vezi sa privatnošću, dok bi funkcionalnost aplikacije ostala nepromenjena. Takođe, proširivanje aplikacije da preciznije i opširnije prati korisnikovu aktivnost, kao što je vreme provedeno na specifičnim zadacima ili sajtovima unutar aplikacija, moglo bi pružiti korisnicima koji žele sveobuhvatniju analizu detaljnije uvide. Međutim, ovo bi trebalo raditi uz punu transparentnost prema korisnicima, kako bi se održalo poverenje i etička upotreba.

Poboljšanje bezbednosti podataka je još jedno područje koje treba razmotriti. Osiguranje da svi prikupljeni podaci budu šifrovani, kako u tranzitu tako i u skladištu, pomoći će u zaštiti osetljivih informacija od neovlašćenog pristupa. Implementacija jasnih i transparentnih korisničkih sporazuma, kao i pružanje opcija za anonimizaciju podataka, moglo bi dodatno da otkloni probleme sa privatnošću. Štaviše, integracija sistema za sigurno i verifikovano logovanje dodatno bi povećala pouzdanost prikupljenih podataka, osiguravajući da oni budu validni i upotrebljivi u forenzičkim istragama.

Zabrinutosti vezane za privatnost	Jedna od glavnih zabrinutosti u forenzici upotrebe računara, posebno sa aplikacijama kao što je ova za praćenje aktivnosti korisnika, jeste potencijalno narušavanje privatnosti. Aplikacija prikuplja detaljne podatke o tome kako korisnici koriste različite softvere, što može otkriti osetljive informacije o njihovim navikama, interesovanjima, pa čak i o ličnosti. Ovo je posebno relevantno u radnom ili deljenom okruženju, gde bi praćenje moglo preći etičke granice ako nije sprovedeno na transparentan način. Ključno je osigurati da su korisnici svesni koje se informacije prikupljaju i zašto. Pružanje jasnih politika privatnosti i omogućavanje korisnicima da se prijave ili odjave iz specifičnih tipova praćenja može pomoći u ublažavanju ovih zabrinutosti. Takođe, anonimizacija podataka gde god je to moguće može pomoći u zaštiti ličnih informacija, a i dalje omogućiti korisne uvide.
Bezbednost prikupljenih podataka	Druga važna zabrinutost jeste bezbednost podataka koje aplikacija prikuplja. Budući da aplikacija prikuplja osetljive informacije o aktivnostima korisnika, postaje potencijalna meta za sajber napade. Šifrovanje podataka, kako tokom prenosa tako i tokom skladištenja, je neophodno kako bi se sprečio

`

	neovlašćen pristup. Dodatno, sigurne prakse skladištenja i robusne mere autentifikacije treba da budu na snazi kako bi se osiguralo da samo ovlašćeni korisnici mogu da pristupe ili modifikuju podatke. Proboj ovih podataka može imati ozbiljne posledice, naročito ako uključuje lične ili poverljive informacije.
Etika i pravne implikacije	Takođe, postoje etičke i pravne implikacije u vezi sa korišćenjem aplikacija za praćenje. U nekim regionima, strogi zakoni regulišu prikupljanje ličnih podataka, a kršenje ovih zakona može dovesti do pravnih posledica. Na primer, praćenje bez eksplicitnog pristanka može narušiti regulative privatnosti, kao što je Opšta uredba o zaštiti podataka (GDPR) u Evropskoj uniji. Kao rezultat, ključno je osigurati da su korisnici potpuno informisani i da daju pristanak pre nego što se bilo koji podaci prikupe. U organizacionom okruženju, neophodno je imati jasne politike koje definišu prihvatljive prakse praćenja, kako bi se osiguralo da se aplikacija koristi odgovorno i u skladu sa zakonskim okvirima.

`

Zaključak

Aplikacija razvijena za praćenje upotrebe računara nudi praktičan alat u oblasti digitalne forenzike, odnosno u forenzici upotrebe računara. Ova aplikacija pruža uvid u način na koji pojedinci koriste različite softverske aplikacije, prikupljajući podatke koji mogu pomoći u analizi produktivnosti, praćenju radnog mesta, ili čak u istragama nepravilnosti. Međutim, iako aplikacija efikasno ispunjava svoju svrhu, kao što je pomenuto u prethodnom poglavlju, postoje značajni izazovi u vezi sa privatnošću, bezbednošću i etikom, koje je potrebno rešiti kako bi se osigurala odgovorna i zakonska upotreba.

Budući da aplikacija prati i beleži specifične detalje o aktivnostima korisnika, ona ima potencijal da naruši privatnost ako se ne koristi uz odgovarajuću saglasnost i transparentnost. Da bi se to ublažilo, buduća unapređenja bi mogla biti usmerena na dodavanje prilagodljivih podešavanja privatnosti, omogućavajući korisnicima da kontrolišu koji podaci se prikupljaju i koliko dugo.

Bezbednost podataka je još jedno ključno područje za unapređenje. S obzirom na to da aplikacija prikuplja osetljive informacije, osiguranje šifrovanja podataka tokom prenosa i skladištenja je poželjna kako bi se zaštitili od neovlašćenog pristupa. Implementacija sigurnih praksi skladištenja i omogućavanje da samo ovlašćeni korisnici mogu pristupiti tim podacima takođe bi unapredila poverenje u aplikaciju.

Sa etičkog aspekta, ključno je osigurati da se aplikacija koristi odgovorno i u skladu sa relevantnim zakonskim okvirima. U okruženjima gde se sprovodi praćenje zaposlenih, korisnici moraju biti potpuno informisani o tome koji podaci se prikupljaju i imati opciju da pristanu na određene aktivnosti praćenja. Pravne regulative, kao što je GDPR, naglašavaju potrebu za transparentnim praksama prikupljanja podataka, a buduće verzije aplikacije mogle bi uključiti ugrađene funkcije za usklađenost sa ovim propisima.

Aplikacija je zamišljena da demonstrira potencijal forenzike upotrebe računara u široj oblasti digitalne forenzike, nudeći koristan alat za razumevanje digitalnog ponašanja. Rešavanjem pitanja privatnosti, bezbednosti i etike, i uz dalja unapređenja, ona može da se razvije u sveobuhvatnije i odgovornije rešenje za individualne korisnike i organizacije.

`

Reference

[1] Wikipedia – Digital forensics

https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiK1NeR-4CJAxVOgf0HHflfIrAQFnoECE0QAQ&url=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FDigital_forensics&usg=AOvVaw3_V-Cok0fs9mtqkmXJUXfV&opi=89978449

[2] What is computer forensics

https://www.techtarget.com/searchsecurity/definition/computer-forensics#:~:text=Computer%20forensics%20is%20the%20application,in%20a%20court%20of%20law.

[3] Computer forensics – subsequent references

https://www.cisa.gov/sites/default/files/publications/forensics.pdf

[4] Python - https://www.python.org

[5] Sendgrid documentation - https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwjYzdTT-4CJAxVIgv0HHWdMJ2sQFnoECBcQAQ&url=https%3A%2F%2Fwww.twilio.com%2Fdocs%2Fsendgrid&usg=AOvVaw31cEUuE2Ud6pJB5QfCzR1I&opi=89978449