**Triskelion: A Python Based Software Keylogger for Vulnerability Assessment in the Cyberspace**

**Ochieng’ Arnold**

**150211**

**CNS**

**Supervisor Name**

**Mr. Humphrey Owuor**

**Submitted in Partial Fulfillment of the Requirements of the Bachelor of Science in Computer Networks and Cybersecurity as the Strathmore University**

**School of Computing and Engineering Sciences**

**Strathmore University**

**Nairobi, Kenya**

**May 2024**

Declaration and Approval

I declare that this work is authentic and purely the work of my hands. This research proposal is devoid of content present in already submitted and awarded material by Strathmore University or any other university except where proper references and citations have been made where due in the research proposal.

Student Name: Ochieng’ Arnold

Admission Number: 150211

Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

The proposal of **Ochieng’ Arnold** has been reviewed and approved by **Mr. Humphrey Owuor**

Supervisor Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Acknowledgement

I would officially like to thank and give recognition to the following parties for their unconditional support as I sat through this project. Had it not been for their presence, care and good wishes, the project would not have been a success. Special thanks to my supervisor Mr. Humphrey Owuor, my family and close kith and kin who took it in their stride to ensure I was well catered for and was as fit as a fiddle as I undertook this project.

Abstract

The rapid evolution of technology, driven by innovative solutions, research, and market demands, has significantly advanced communication, networking, hardware, and software sectors. However, these advancements have also facilitated the rise of cybercrimes, with keylogger attacks being a notable threat to data security and privacy. This study aims to address gaps in effective detection mechanisms for and the knowledge availed to the public about software keyloggers. The objective is to design and develop a Python-based software keylogger named Triskelion, which will be used to assess vulnerabilities in cyberspace through an offensive approach. Triskelion will capture keystrokes, clipboard data, and screenshots from a target machine, operating in an ethically controlled environment to ensure legitimate use. The collected data will be analyzed to gain insights into keylogger evasion techniques, functionalities, and threat scope. The expected outcome is to enhance the understanding of keylogger behaviour, which will inform the development of more robust and dynamic detection mechanisms. These findings will also aid in raising user awareness about the risks posed by keyloggers and the importance of protecting sensitive information. Ultimately, this research aims to contribute to the cybersecurity field by providing solutions that mitigate the impact of keylogger attacks and uphold data integrity and user privacy.

Table of Contents

[Declaration and Approval i](#_Toc167188249)

[Acknowledgement ii](#_Toc167188250)

[Abstract iii](#_Toc167188251)

[List of Figures vii](#_Toc167188252)

[List of Tables viii](#_Toc167188253)

[List of Abbreviations ix](#_Toc167188254)

[Chapter 1: Introduction 1](#_Toc167188255)

[1.1 Background Information 1](#_Toc167188256)

[1.2 Problem Statement 2](#_Toc167188257)

[1.3 Objective 3](#_Toc167188258)

[1.3.1 General objective 3](#_Toc167188259)

[1.3.2 Specific Objectives 4](#_Toc167188260)

[1.4 Research Questions 4](#_Toc167188261)

[1.5 Justification 4](#_Toc167188262)

[1.6 Scope, Delimitations and Limitations of the project 5](#_Toc167188263)

[1.6.1 Scope of the project 5](#_Toc167188264)

[1.6.2 Delimitations of the project 5](#_Toc167188265)

[1.6.3 Limitations of the project 6](#_Toc167188266)

[Chapter 2: Literature review 7](#_Toc167188267)

[2.1 Introduction 7](#_Toc167188268)

[2.2 The design of keyloggers 7](#_Toc167188269)

[2.3 The development of software keyloggers 8](#_Toc167188270)

[2.3.1 Windows Keyboard Hook method 8](#_Toc167188271)

[2.3.2 Keyboard State Table method 9](#_Toc167188272)

[2.3.3 Kernel-Based Keyboard Filter Driver method 10](#_Toc167188273)

[2.4 The evaluation of keyloggers’ functionalities and threat spectrum 10](#_Toc167188274)

[2.5 The development of simple detection mechanisms for keyloggers 11](#_Toc167188275)

[2.5.1 Signature-Based Detectors 11](#_Toc167188276)

[2.5.2 Behavioural-Based Detectors 12](#_Toc167188277)

[2.6 Related works 12](#_Toc167188278)

[2.6.1 Case 1: Design, Analysis and Implementation of an Advanced Keylogger to Defend Cyber Threats (Bejo et al., 2023) 12](#_Toc167188279)

[2.6.2 Case 2: Enhancing Digital Security with the Advanced Keylogger Project (Kapare et al., 2024) 13](#_Toc167188280)

[2.6.3 Case 3: Developing Advanced Software Keylogger using Python and Creating Awareness of their Functionalities (Joy et al., 2023) 13](#_Toc167188281)

[2.7 Gaps in related works 14](#_Toc167188282)

[2.8 Conceptual Framework 14](#_Toc167188283)

[Chapter 3: Methodology 16](#_Toc167188284)

[3.1 Introduction 16](#_Toc167188285)

[3.2 Methodology 16](#_Toc167188286)

[3.3 Justification of the methodology 16](#_Toc167188287)

[3.4 Methodology diagram 17](#_Toc167188288)

[3.4.1 Planning 17](#_Toc167188289)

[3.4.2 Analysis 18](#_Toc167188290)

[3.4.3 Design 18](#_Toc167188291)

[3.4.4 System prototype 18](#_Toc167188292)

[3.4.5 Implementation 19](#_Toc167188293)

[3.4.6 System 20](#_Toc167188294)

[3.5 Deliverables 21](#_Toc167188295)

[3.5.1 Proposal 21](#_Toc167188296)

[3.5.2 Design diagrams 21](#_Toc167188297)

[3.5.3 Keylogger program executable 21](#_Toc167188298)

[3.5.4 Final report 21](#_Toc167188299)

[3.6 Tools and techniques 21](#_Toc167188300)

[3.6.1 Python 21](#_Toc167188301)

[3.6.2 Kali Linux Virtual Machine 21](#_Toc167188302)

[3.6.3 Oracle VirtualBox 21](#_Toc167188303)

[3.6.4 Git and GitHub 22](#_Toc167188304)

[3.6.5 Visual Studio Code 22](#_Toc167188305)

[References 23](#_Toc167188306)

[Appendices 26](#_Toc167188307)

[Appendix 1: Gannt Chart 26](#_Toc167188308)

[Appendix 2: Marking Guide 27](#_Toc167188309)

List of Figures

[Figure 2.1 Block Diagram of Hook Mechanism (Abukar et al., 2014) 8](#_Toc167188310)

[Figure 2.2 Keyboard State Table Method (Abukar et al., 2014) 9](#_Toc167188311)

[Figure 2.3 Proposed Taxonomy for Keyloggers (Bhardwaj & Goundar, 2020) 11](#_Toc167188312)

[Figure 2.4 Emails Received Containing Logs (Bejo et al., 2023) 12](#_Toc167188313)

[Figure 2.5 Flow Diagram (Joy et al., 2023) 13](#_Toc167188314)

[Figure 2.6 Conceptual Framework 15](#_Toc167188315)

[Figure 3.1 Prototyping Methodology (Aman et al., 2018) 17](#_Toc167188316)

[Figure 3.2 Evolutionary Prototype Model (Shao & Dida, 2020) 20](#_Toc167188317)

List of Tables

[Table 3.1 Test Case Template Example 19](#_Toc167188318)

List of Abbreviations

API: Application Programming Interface

ID: Identity/Identification

IDE: Integrated Development Environment

MHz: Megahertz

OS: Operating System

PC: Personal Computer

RF: Radio Frequency

SMTP: Simple Mail Transfer Protocol

USB: Universal Serial Bus

# Introduction

## Background Information

The evolution of technology can be defined as the sustained growth and advancement of technologies over time facilitated by innovative solutions, research and demands of the market. It is wholesome as it entails developments in communication, networking, hardware and software technologies which brazes the trail for metamorphic changes in our society today (Nafish & Rengarajan, 2024). Drastic changes and advancements in technology brought about the growth and development of the internet and the cyberspace to interconnect users and break geographical boundaries between them to facilitate communication, medical care, business and education among other sectors of daily life. However, with these notable advancements, cybercrimes and attacks have become more prevalent (Nagathota et al., 2023). Data breaches otherwise known as data leaks refer to security violations whereby unauthorized parties gain access to guarded data caused by several causes, one of them being hacker’s activities by use of malware such as phishing emails and keyloggers (Ifinedo et al., 2024). The scope of this study is limited to software keyloggers with functionalities to steal passwords, obtain clipboard data and screenshot user’s activity techniques such as server architectures for storing and reporting stolen credentials in stealth mode on user devices.

According to Kapare et al. 2024, the risk factor associated with sensitive information on the internet is exceedingly high considering the plethora of information on the internet. Keyloggers which were originally adopted for licit uses such as troubleshooting and forensic analysis have since been employed in malicious activities such as identity theft, financial fraud and espionage in certain cases. Their clandestine nature makes them invisible to users who fail to know of their presence on their devices which further propagates the risk of data breaches and exploitation. Despite the rapid advancements in keylogger based attacks over the recent years, there have not been many proposals of efficacious and effective solutions to address the problem of keyloggers especially software keyloggers as the signature and behaviour based detection techniques try to overcome keyloggers but fail to provide assured detection putting in consideration their sophistication (Santoki, 2014). Furthermore, by logging keystrokes, hackers can obtain and access sensitive information with the log files providing sufficient information about keyboard events and applications visited by users which offer proof to reveal users’ activity thus infringing on their privacy and most importantly, without their consent or knowledge as many antivirus programs lack the capabilities to identify the presence of keylogger activity on their devices (Bejo et al., 2023). The insidious nature of software keyloggers deters their detection and this poses a great threat to data safety and privacy.

The significance of addressing these gaps with regards to keyloggers lies in the significance of protection of sensitive information of users in the cyberspace. The internet has become an imperative part of our world today as it has increased efficiency and enhanced convenience of previously tedious tasks. Information shared on the internet or accessed by internet users needs to be well protected against breaches and unwanted access to uphold the privacy of users and integrity of the data. This emphasizes the need for robust and dynamic detection mechanisms for keyloggers to provide alerts and detect keyloggers to make users aware of keylogging activities. This initiates prompt responses in curbing these attacks and denying access of users’ sensitive information to hackers with the intention of financial gain or propagating other cybercrimes using the keylogger as a backdoor. Nonetheless, it aids in providing information on notable signs of keylogger attacks on systems and offers solutions in deterring their success.

## Problem Statement

According to Mbanaso & Dandaura, 2015, the advent of the Internet and the growing utilization of information technologies have resulted in remarkable transformations in human existence. It is revolutionizing the economic growth of numerous nations, removing trade obstacles, and enabling global communication, cooperation, and idea sharing despite the conventional boundaries of time, place, and class. The convergence of people, information systems, and the internet commonly referred to as "cyberspace" has produced a global virtual environment for competitive advantage. Cyberspace technologies are being used by governments, enterprises, organizations, and individuals globally to increase productivity and profitability. It is, in fact, changing security postures, socioeconomic activity, and opening doors for success and creativity. Additionally, it has increased the tools available to enhance global welfare and universal governance. The internet has brought about more opportunities for research, development and innovations, which ultimately is leading to exceptional economic growth and prosperity, as well as enabling informed societies worldwide at an amazing speed (Mbanaso & Dandaura, 2015).

However, vulnerabilities, hazards, and risks are ever-present in cyberspace, creating opportunities for exploitation, collusion, and conflict. Because of the inherent vulnerability that varies due to multiple factors, the increasing level of interdependence between physical and virtual components, people, and processes is relentlessly opening up unpredictable vulnerabilities, threats, and risks (Mbanaso & Dandaura, 2015). A lot of information is shared on the internet via our devices as our lives become more digital and this might be compromised by keylogging techniques which capture keystrokes entered into a system obliviously to the user (Bejo et al., 2023). Malware designed to do this are keyloggers, surveillance types of software to capture keystrokes and avail them to unwanted parties. They were initially hardware based and were utilized for legal purposes such as debugging and systems repair but with the evolution of technology, they became more sophisticated and dangerous. They have become a hazard to individuals and businesses as they are utilized by attackers to steal sensitive information such as passwords and login credentials, bank credentials and personal communications. This has eventually led to identity theft and financial fraud among other cybercrimes on the cyberspace (Joy et al., 2023). This thus compromises the safety of users’ information online and hinders effective use of the internet and its innovations to enhance convenience and efficiency.

The challenges therefore present the need for an intervention with regards to keyloggers and their propagation. This emphasizes the need for this study, to come up with a software keylogger to comprehend their mechanisms of operation and evasion as well as assess the vulnerabilities coupled with its use. The implementation and deployment of the keylogger will aid in collection of data for analysis to provide insights on the development of a method of detection for the keylogger. The information gathered at the end of the study will be significant in dissemination of knowledge regarding keyloggers and advance the fields of cybersecurity and malware assault research.

## Objective

### General objective

To design and develop Triskelion, a Python based software keylogger to exploit vulnerabilities in an ethically controlled environment to grasp the principles behind keyloggers and use the data collected to develop a simple mechanism for keylogger detection.

### Specific Objectives

The specific objectives of this research entail:

1. To design Triskelion as a software keylogger to run on a target device.
2. To develop the Triskelion keylogger using Python that can capture keystrokes, screenshots and copied clipboard data on a target device.
3. To analyze the functionalities and threat spectrum of keyloggers.
4. To develop a mechanism of detection to detect the Triskelion keylogger based on the data collected and analysis of the keylogger.

## Research Questions

Some of the research questions include:

1. What are some of the underlying principles that make software keyloggers undetected on devices?
2. How can users be more wary of keyloggers running in the background of their devices and what measures are to be taken in the event of the detection of a keylogger?
3. What are the legal uses of keyloggers and what defines and guides their legal use?
4. What is the system architecture for keyloggers in terms of the connections between the attacker and the keylogger software itself to send the keystrokes generated by the user?
5. How can detection methods be designed to detect keyloggers running on users’ devices to provide early detection before attack propagation?

## Justification

Based on the gaps of keylogger detection presented by Santoki, 2014 and the insidious nature of keyloggers highlighted by Bejo et al., 2023, they validate the need for conducting this study. The research provides some insight into the internal workings and guiding principles of keyloggers, in addition to the vulnerabilities they attack. This highlights the need to understand not only what keyloggers do but also how they operate, including how they get past security measures and hide in the background while carrying out their nefarious schemes. Additionally, this data helps the project incorporate a Triskelion keylogger detection mechanism that seeks to explore dynamic detection schemes that will suffice in notifying users of keylogger activity. This makes it possible to make users more aware of the risks associated with keyloggers and offers more information than what they already know and further aid in more research on the topic.

## Scope, Delimitations and Limitations of the project

### Scope of the project

This proposed study is to focus on the design and development of Triskelion, a software keylogger with three main functionalities: capturing keyboard strokes, screenshots and copied clipboard data on a target machine. The keylogger will then be executed in an ethically controlled environment and the captured data will be sent over to a server which is accessible by the attacker machine. The success of the keylogger and the data collected will then offer information on the threat spectrum of keyloggers. Additionally, this will be instrumental in the development of a detection system for the keylogger which will offer detection and provide an alert on the presence of the keylogger. The information gathered will be used to assess vulnerabilities associated with keyloggers.

### Delimitations of the project

The following have been excluded from this study:

1. Hardware keyloggers – although they are also as lethal as software keyloggers, hardware keyloggers are relatively easy to notice since they exist as unaccounted for peripheral devices on users’ systems. They are also hard to deploy since the attacker must have physical access to the users’ devices to insert the hardware keyloggers and might be noticed via surveillance cameras. These factors make hardware keyloggers uncommon and thus exclusion in this study.
2. Other forms of malware – there is a multitude of malware in the cyberspace each with their different propagation methods and threat spectrums. This study chooses to focus on keyloggers and explores it in entirety since researching on each one requires time and intricacy that is not catered for in the project timeline.
3. The development of a prevention mechanism – the study propagates the keylogger and offers the design of a detection mechanism based on the success of the attack. The limited time for the project does not allow the design of a prevention mechanism which however can be explored in future as an extension of the current study.

### Limitations of the project

First, the Triskelion will be designed to capture keystrokes, copied keyboard data and the screenshots on devices running Windows OS. This thus excludes other operating systems such as Linux and MacOS which are also susceptible to keylogger attacks. Furthermore, availing the keylogger program as an open-source tool on GitHub deters the ethical use case of Triskelion. The project is to be conducted in an ethically controlled environment to help assess vulnerabilities, but nothing stops another developer with malicious intent from using Triskelion to propagate a cyberattack. Finally, exploring legal use cases of keyloggers might prove to be difficult since it breaks user privacy of the target, whether for domestic use or by orders of state intervention. User privacy is among the requirements for any secure system and should always be upheld with no compromise.

# Literature review

## Introduction

The goal of this project is to create a Python-based software keylogger that can be used to find vulnerabilities in an ethically controlled environment, gain insight into how keyloggers operate, and improve network security for people online. The design and implementation of Triskelion will be instrumental in understanding keylogger functionalities and coming up with a mechanism of discerning keylogger activities on user devices. This chapter intends to delve into certified journals, articles and other forms of literature to eke information on keylogger design principles, case studies and keyloggers developed in the recent past and possible gaps in research revolving around keyloggers.

## The design of keyloggers

As per the views of Shirke et al., (2023), “The design and implementation of a keylogger depends on factors such as the transmission medium, the target machine type, the keylogger’s intended duration of operation, and the desired level of discretion and traceability while active.” Commonly employed methods include remote introduction of software keyloggers into an operating system or using hardware keyloggers which are introduces to systems physically. They should further capture keystrokes in real time and additional data such as mouse clicks and even capture screenshots all of which can be monitored discreetly by the owner of the keylogger. The captured data can either be saved on a folder hidden on the user’s system or send them to the owner’s email address and most importantly, the keylogger has to remain hidden from the victim while capturing the keystrokes. (Shirke et al., 2023)

As per the views expressed by Abukar et al., 2014, there are various designs in the making of keyloggers that bring about the various types of keyloggers. First, hardware keyloggers involve use of physical devices either connected between the keyboard and computer directly such as the USB keylogger, or those that require the installation of the keylogger circuit into the keyboard standard. Next is acoustic keyloggers which capture sounds of single keystrokes using special keystrokes to listen to a user typing. They make use of parabolic microphones to record the sounds of the keyboard a hundred feet away (Olzak, 2008). Abukar et al., 2014 further discusses wireless keyloggers that intercept sent payloads from wireless keyboards that use 27 MHz RF connection of encoded RF transported keystroke characters. It transfers the logs up to 100 meters. Finally, software keyloggers are designed to intercept data as it traverses along the keyboard and the OS. They can be sent as links that entice users to click them or as phishing emails. The captured data is either stored remotely and sent to the attacker, saved on a hidden file locally or sent to the attacker’s email address.

## The development of software keyloggers

According to Abukar et al., (2014), there are three main methods in the development of software keyloggers.

### Windows Keyboard Hook method

This method gives the keylogger functions to monitor the keyboard. Keypresses are accounted for by the OS and the application is registered. Afterwards, the application must approve any message employing this mechanism before the message is sent to the original target to receive the message. Windows messages are affiliated with two unique hooks namely Global hook and Local hook. As the names suggest, the Global hook checks system wide message and Local hook checks messages specific to applications (Abukar et al., 2014). Figure 2.1 shows this method and its working mechanism.

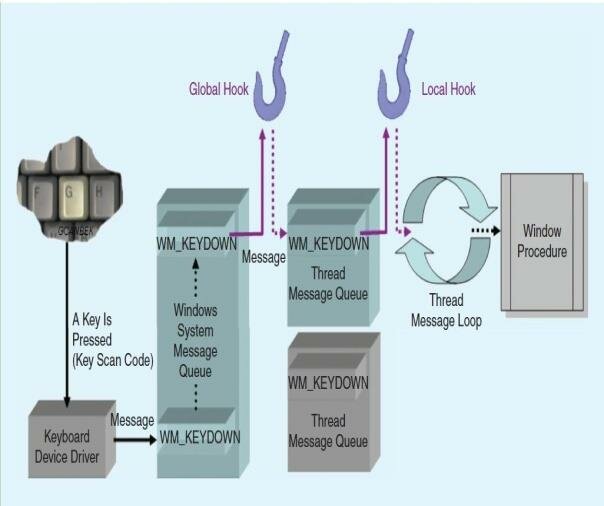


Figure 2.1 Block Diagram of Hook Mechanism (Abukar et al., 2014)

The keyboard hook is capable of:

1. Accessing all keyboard messages and transferring them to the consequent hook operation in a chain.
2. Changing the actual message and transferring it to the next hook procedure.
3. Disrupt message flow by failing to pass it to the next hook procedure.

### Keyboard State Table method

This contains a table which contains the state information of 256 virtual keys. Applications using window interfaces make references to the table to determine whether keys are up or down. An example provided by Abukar et al. (2014) is “ when key is pressed with Control or Shift key, keylogger can utilize the GetKeyboardState API functions to disclose or reveal the keystroke information, by adding its thread to the top-level of thread message loop of window using Attach-ThreadInput API.” Figure 2.2 indicates the state table and an example of its operation upon a key press.

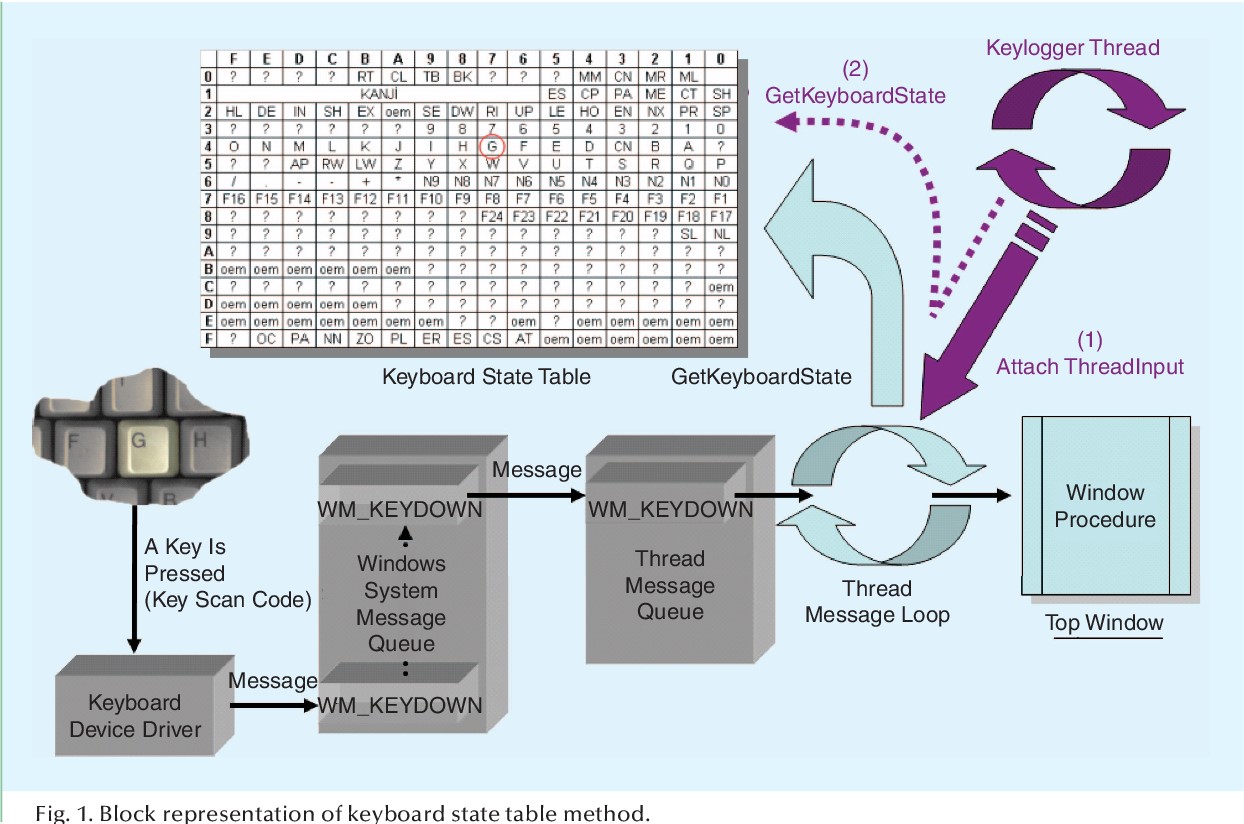


Figure 2.2 Keyboard State Table Method (Abukar et al., 2014)

### Kernel-Based Keyboard Filter Driver method

As suggested by Abukar et al. (2014), this method allows keyloggers’ filter driver to capture the keystrokes before they even reach the operating system. They are typically hard to detect but require administrative privileges to be installed on target devices.

## The evaluation of keyloggers’ functionalities and threat spectrum

As per the article by Bhardwaj & Goundar 2020, keyloggers might as well masquerade as legitimate programs similar to trojans and bypass anti-virus programs. They also operate at higher privilege levels in comparison to other forms of malware. They are not illegal per se, but most times are used for more harm than good. The article groups the functionalities of keyloggers into five groups. First is the security functionality which is related to how the keyloggers are invisible and are unable to be detected and camouflage from the Task Manager and conceal their activity. This is also related to encryption of the logged files locally and sending them to the attacker’s email address to public SMTP servers undetected. Second is the monitoring options portrayed in the keylogger. This entails interception of information such as keystrokes and logon credentials, as well as copying information from the memory or the clipboard is an advanced feature coupled with the ability to execute and terminate programs such as web cams and even shut down entire systems. Some keyloggers record on-mouse clicks as well as webcam and microphone recordings. Furthermore, they can monitor the online activities of users. This is inclusive of URLs and web pages accessed in web browsers, generating lists of emails both inbound and outbound via browsers and email applications and capture details of the messenger chats of users on applications such as Skype, Facebook and other social media applications. Nonetheless, they are also equipped with the ability to report and filter the logs sent to the attacker. These logs and reports document activities, the duration for predefined programs and a summary based on particular keywords. Finally, they can react and give alerts based on specified words as well as the ability to start and stop logging or only log keystrokes from specific websites with some being able to provide real time monitoring or even viewing on mobile phones. Figure 2.3 shows a proposed taxonomy for keyloggers based on their execution and functionality.

A diagram of software components

Description automatically generated

Figure 2.3 Proposed Taxonomy for Keyloggers (Bhardwaj & Goundar, 2020)

Keyloggers mostly threaten highly sensitive information such as passwords, user IDs, social security numbers as well as credentials pertaining to social media, email and gaming. They can record any text sequence and can also be utilized for espionage depending on the target machine (Navarro et al., 2012). They also threaten copied text or images, capture PC screens at random intervals, log activities executed on the target machines, Control Text Capture and online actions such executing program queries (Bejo et al., 2023).

## The development of simple detection mechanisms for keyloggers

Elelegwu et al., 2024 endorse that keylogger detectors, also known as anti-keyloggers work unitedly or as part of security software. They are tailored to locate keylogging activities offering some form of protection. They primarily function to protect legitimate users from keylogging activities and hinder the capture of their keystrokes. This guards their sensitive details from attackers. The types included include:

### Signature-Based Detectors

These detectors compare and cross-index system files with databases of well-known keylogger signatures.

### Behavioural-Based Detectors

These monitor the behaviour of the system and detect peculiarities and patterns associated with keylogger activities. These types are mostly in use nowadays.

Signature based methods have been bypassed using code encryption techniques and construct several variants of malware to evade such static detection techniques which led to the development of dynamic techniques such as virtual machine inspection, function call monitoring , dynamic binary instrumentation binary instrumentation (Das et al., 2016).

## Related works

Keyloggers have evolved over the years with the growth of the internet. Some of the related works about keyloggers include:

### Case 1: Design, Analysis and Implementation of an Advanced Keylogger to Defend Cyber Threats (Bejo et al., 2023)

The study involved the design, analysis and creation of an advanced keylogger in Python with the purpose to defend against cyberthreats. The keylogger code was executed and ran in the background, a log file was randomly created at random either in “Pictures” or “Downloads” directories with a random variable name but containing “I” in the second alphabet. It then would begin logging files for a duration of time specified in its code and after the time elapses, an email is sent with the log file attached to it and the log file is deleted from the system and this continues in a loop fashion. Figure 2.4 shows the received emails on the developer’s end containing the stored logs of captured data from the target machine.

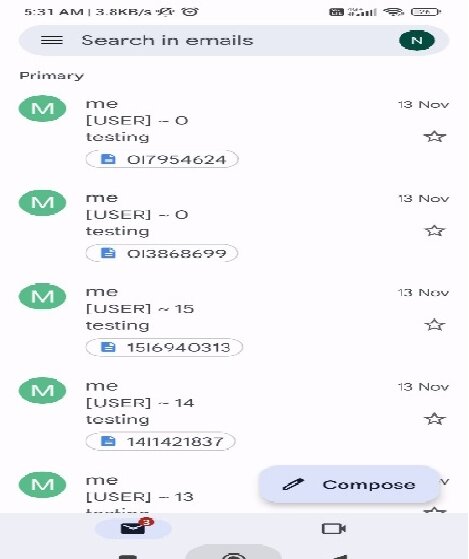


Figure 2.4 Emails Received Containing Logs (Bejo et al., 2023)

### Case 2: Enhancing Digital Security with the Advanced Keylogger Project (Kapare et al., 2024)

The project also developed the keylogger using Python, designed it to operate undetected in the background, evade detection mechanisms and covert collection of data specifically keystrokes, clipboard contents, audio recordings and screenshots. It also sent the logged data via email by the use of the `smtplib` library to establish a connection with the SMTP server and send the logged data. The captured data was then encrypted prior to transmission using the Fernet symmetric encryption algorithm and captured data was stored on the user’s system temporarily before encryption and transmission guided by preset thresholds for data accumulation. The project was then tested extensively to assess the keylogger’s performance while considering the ethicality of the experiments conducted. The results and conclusions were conducted thereafter.

### Case 3: Developing Advanced Software Keylogger using Python and Creating Awareness of their Functionalities (Joy et al., 2023)

The proposed system was designed with functionalities that entailed the capture of keystrokes, clipboard content, system information, clipboard images, screenshots and audio inputs via microphone and ensured undetected operation. After collection of this data, the data would be transmitted via email and the keylogger was designed to consume fewer resources and use a threaded approach to avoid detection. Figure 2.5 is the flow diagram for the project undertaken.

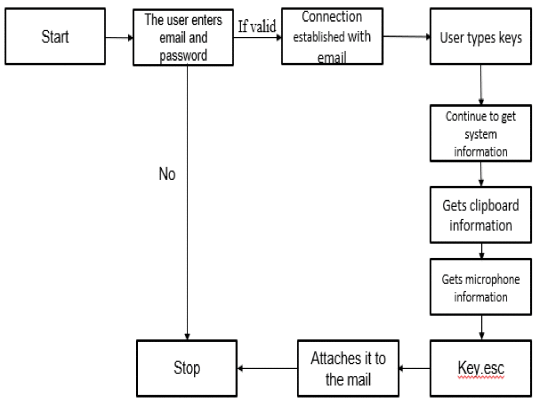


Figure 2.5 Flow Diagram (Joy et al., 2023)

The project highlighted the difference between traditional keyloggers which primarily track keystrokes and advanced variants with more and greater functionalities which increases their potential threat and effectiveness.

## Gaps in related works

1. The keylogger was victim to detection by antivirus and still needed stealth development to avoid being discovered by advanced antivirus and firewalls and also was very CPU intensive (Bejo et al., 2023).
2. The keylogger's operation and performance were dependent on a number of technical aspects, such as system setups, program environments' and operating systems' compatibility, and the dependability of third-party libraries which were not fully explored (Kapare et al., 2024).
3. Some literature lacked information regarding whether the designed keyloggers were detected by antivirus programs or not as well as statistics on their detection.
4. There is lack of sufficient information on the duality of keyloggers, that is their legal use and the abuse of their functionalities to compromise data online.
5. Detection mechanisms that solely employed signature-based detection fall short since users are only secure against keyloggers whose signatures are in the database.

## Conceptual Framework

Figure 2.6 indicates the data flow in the system. The attacker sends out an anonymous link to a legitimate user. The user clicks on the link and unknowingly launches the keylogger program on his/her machine. The keylogger captures the keystrokes, copied clipboard data and takes screenshots. The captured information is then sent to a remote sever which both the keylogger program and the attacker have access to. The server stores this information received from the keylogger and the attacker can access this information from the remote server. Upon detection of the keylogger on the system, the user receives an alert informing him/her of its presence on their device. The attacker can access the captured information on the server for analysis.

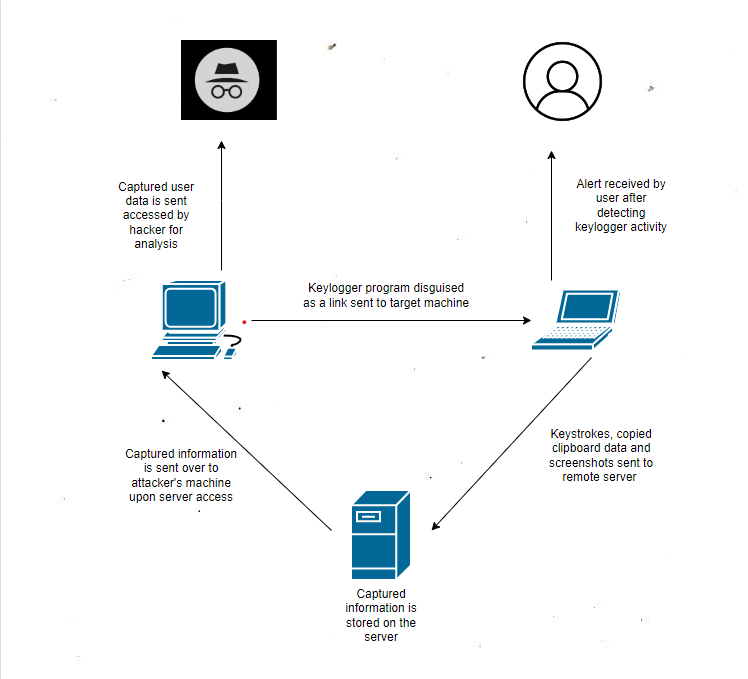


Figure 2.6 Conceptual Framework

# Methodology

## Introduction

This chapter outlines the methodology to be used for the project. It contains information regarding the approach that will be employed in project design and methods that will be utilized for completion of the project. The chapter has the methodology and justification for its choice, the methodology diagram and stages in the methodology with relevance to the development of the project, the deliverables realized during the project and their importance and finally tools and techniques to be used for the project as well.

## Methodology

The methodology of choice for the Triskelion keylogger is Prototyping. According to Zant (2005), prototype based methodologies are focused on development of an evolutionary prototype with a set of high precedence attributes and is further expanded and augmented with the onset of newer requirements. It is an iterative method with prompt planning where feedback is present which caters for repetition and refinement of the initial software until it meets what is required. The aim of prototyping is to develop the inceptive software model into the final system (Susanto & Meiryani, 2019). In this methodology, evolutionary prototyping is chosen since it involves the reuse and refinement of pieces of previously designed prototypes in making the final system (Laborde et al., 2020).

## Justification of the methodology

First, prototyping will be adopted since it effectively decreases the time spent in production since by using operational models of the end product early in the project, it does away with tedious and protracted revisions later and with the execution of task related designs completed concurrently instead of in a sequence throughout the project (Jones & Richey, 2000). Furthermore, it caters for the adaptability of change early enough before the final system design and development (IxDF, 2019). Prototypes also act as milestones as they indicate progress and advancements in the project as well as bring out features as well as their integration to make the project whole and functional (Berglund & Grimheden, 2011). Finally, prototyping allows the generation of ideas from what is implied and makes them tangible aiding in the processes of conceptualization, exploring designs and evaluation as well as construction (Berglund & Leifer, 2013). All these are vital to the project and the accomplishment of its objectives thus the reason for choosing prototyping methodology. Specifically, evolutionary methodology is optimal since it supports reuse of previous prototypes thus saves on time in the iteration process, allows refinement throughout the phase of development and allows focus on progressive development of less complicated to more complicated features of the project than developing the final system as a whole (Nordin et al., 2018).

## Methodology diagram

Figure 3.1 is the methodology diagram of the Prototyping methodology. It indicates the various stages of the methodology which include planning, analysis, design, prototype and implementation and the final system. Three of these phases are iterated for refinement of the prototype and testing to come up with the complete system.

A diagram of a system prototype

Description automatically generated

Figure 3.1 Prototyping Methodology (Aman et al., 2018)

### Planning

The developer critically calculates and charts out the project by clearly outlining the objectives of the project and the end goal to be achieved. The activities at this stage include:

Preparing the project proposal for Triskelion which outlines the project’s objectives and its justification, the review of previous literature and their gaps as well as related works and the methodology that will be used for the project as well as deliverables and tools and techniques to be used.

Identifying a project schedule for the keylogger whereby the developer schedules individual project tasks into and sets relevant timelines for conducting the project as well as well as the activities to be carried out and results.

The results at this stage are the project proposal document and a Gantt chart to show the work plan till the conclusion of the project.

### Analysis

This step involves gathering information regarding the desired features and functionalities of the system. Requirements are specified at this stage and once they are defined, the developer comes up with a roadmap of how the system will be developed and how to begin the prototype development. Triskelion’s three functionalities are defined here, that is capturing keystrokes, copied clipboard data as well as screenshots. Python is identified as the programming language for the development of the keylogger software itself. The configurations for the remote server where the keystrokes will be sent, stored and accessed will also be analyzed and accounted for in the analysis stage.

### Design

The developer comes up with a quick design of the keylogger to provide a brief idea of the final system. This is necessary for creating prototypes and is not the final design as it will require refinement and modifications according to the feedback obtained after testing and the developer’s review. As such, this step will be iterated until the design covers all the requirements identified and conceptualizes the final system. The research paradigm chosen for the project is Structured Systems Analysis and Design Method. The reason for its choice is that it breaks the project into stages, modules, steps and tasks which suffices for the design of Triskelion.

### System prototype

In this stage, the developer creates a small working model of the system. From the requirements and information gathered, the prototype will be created. This model will be improved upon with subsequent iterations of the process. Triskelion will be prototyped using evolutionary prototyping; thus, the prototypes will be developed incrementally starting with less complicated to more complicated functionalities at each iteration and reusing prototypes as each iteration to save on time. The tools and development techniques to be used will be Python which is the programming language which the keylogger program will be written, Kali Linux virtual machine which will be configured to act as the server for the storage and access of the captured data of the keylogger, Oracle VirtualBox will the hypervisor for the Kali Linux virtual machine, Visual Studio Code will be the IDE for writing and editing the code for the keylogger program and Git and GitHub for version control and collaboration on the project.

### Implementation

The prototype at each iteration is put into practice and tested. Testing is necessary to check whether it meets the requirements and accomplishes the functionalities stated in the analysis stage. The prototypes of the keylogger are tested using experiments and white box testing to determine the accuracy since the design and implementation of Triskelion is known to the developer. The testing types to be used are system integration to ensure the subsequent prototypes developed work with increasing functionalities of each prototype and can be integrated finally to develop the final system and the other testing type is functional testing to verify the keylogger works as expected, that is capturing the keystrokes, copied clipboard data and screenshots. Table 3.1 is a test case template example to be drawn for the test cases for Triskelion.

Table 3.1 Test Case Template Example

|  |  |
| --- | --- |
| Fields | Description |
| Test Case ID | Unique ID for the test case. |
| Test Case Description | Description for the test case to give information on what the test case is all about. |
| Pre-Conditions | Conditions that need to be met before conducting the test case. |
| Test Steps | Detailed steps to be carried out during the testing. |
| Test Data | Input for the test case. |
| Expected Result | The result that is implied or expected to be observed after conducting the test case. |
| Post Condition | Conditions that need to be met after conducting the test case. |
| Actual Result | The true results observed after conducting the test case. |
| Status | Either “PASS” or “FAIL” depending on the comparison of the actual result and that which was expected. |
| Project Name | Name of the project to which the test case belongs. |
| Module Name | Name of the module to which the test case belongs. |
| Created By | Name of the creator of the test cases. |
| Date of Creation | Date which the test case was created. |
| Reviewed By | Name of the reviewer of the test case. |
| Date of Review | Date of the review of the test case. |
| Executed By | Name of the tester who executed the test case. |
| Date of Execution | Date which the test case was executed. |
| Comments | Relevant comments to aid in the comprehension of the test case. |

Implementation of the prototypes brings them into focus and action allows the developer to observe the functionality of the keylogger. Feedback is obtained from the results and necessary adjustments are made where fit during each iteration of the previous stages and then further tested and implemented.

### System

This is the final system. The developer conducts the final implementation and testing of Triskelion and after it has been observed to meet all requirements, the system is finally delivered for the project.

Figure 3.2 is a diagram as well for the evolutionary prototyping which involves the reuse of the previous prototypes to refine and adjust the final system.

A diagram of a system

Description automatically generated

Figure 3.2 Evolutionary Prototype Model (Shao & Dida, 2020)

## Deliverables

The deliverables of the project include:

### Proposal

Detailed document containing relevant information about the project such as the scope of the project, objectives and methodology to be employed in the project.

### Design diagrams

These are the diagrams used for the design of the project which will be delivered as well for scrutiny and review of how they aided in bringing the project to life.

### Keylogger program executable

This is the actual keylogger software program. It will be submitted and used for the final demonstration at the end of the project.

### Final report

This is a document containing the project in its entirety. It contains the project proposal as well as information regarding the system implementation and design and testing to be employed for the system.

## Tools and techniques

The following tools and techniques will be used in the project design and implementation.

### Python

Programming language in which the keylogger program will be written. It leverages its flexibility, ease of use, and extensive library support.

### Kali Linux Virtual Machine

This will be configured to act as server to which keystrokes are sent and stored.

### Oracle VirtualBox

Hypervisor for running the Kali Linux virtual machine. It has multiple features and offers relatively great performance for running guest operating systems. It is also the only professional solution freely available as open-source software under the terms of the General Public License version 3.

### Git and GitHub

Git is a version control system used to track code changes and collaboration on projects. It performs project management via repositories, allows control and tracks changes, branching and merging to allow development for different parts and versions of projects as well as pushing local updates to the project. GitHub creates tools that utilize Git and is used for hosting source code.

### Visual Studio Code

IDE for software writing, editing and testing. It contains tools that make the development process less tedious such as IntelliSense for autocompletion, debugging capabilities within the editor, built in Git commands and has many extensions to support various programming languages.

References

Abukar, Y., Maarof, M., Hassan, F., & Mohamed, A. (2014). Survey of Keylogger Technologies. *I Nternational J Ournal of C Omputer S Cience and T Elecommunications*, *5*, 25–31.

Aman, N., Miftah, R., Murli, N., Mustapha, A., & Zainuddin, M. (2018). Convoy Marshal: Group-Based Navigation Mobile Application. *Advanced Science Letters*, *24*, 1660–1665. https://doi.org/10.1166/asl.2018.11132

Bejo, S., Kumar, B., Banerjee, P., Jha, P., Singh, A., & Dehury, M. (2023). *Design, Analysis and Implementation of an Advanced Keylogger to Defend Cyber Threats* (p. 2274). https://doi.org/10.1109/ICACCS57279.2023.10112977

Berglund, A., & Grimheden, M. (2011). *The Importance of Prototyping for Education in Product Innovation Engineering*.

Berglund, A., & Leifer, L. (2013). Why we Prototype! An International Comparison of the Linkage between Embedded Knowledge and Objective Learning. *Engineering Education*, *8*, 2–15. https://doi.org/10.11120/ened.2013.00004

Bhardwaj, A., & Goundar, S. (2020). Keyloggers: Silent cyber security weapons. *Network Security*, *2020*, 14–19. https://doi.org/10.1016/S1353-4858(20)30021-0

Das, S., Liu, Y., Zhang, W., & Chandramohan, M. (2016). Semantics-Based Online Malware Detection: Towards Efficient Real-Time Protection Against Malware. *IEEE Transactions on Information Forensics and Security*, *11*(2), 289–302. https://doi.org/10.1109/TIFS.2015.2491300

Elelegwu, D., Chen, L., Ji, Y., & Kim, J. (2024). A Novel Approach to Detecting and Mitigating Keyloggers. *SoutheastCon 2024*, 1583–1590. https://doi.org/10.1109/SoutheastCon52093.2024.10500122

Ifinedo, P., Vachon, F., & Ayanso, A. (2024). Reducing data privacy breaches: An empirical study of relevant antecedents and an outcome. *Information Technology and People*. https://doi.org/10.1108/ITP-07-2022-

IxDF, I. D. F.-. (2019). *What is Prototyping?* [Web Page]. https://www.interaction-design.org/literature/topics/prototyping

Jones, T. S., & Richey, R. C. (2000). Rapid prototyping methodology in action: A developmental study. *Educational Technology Research and Development*, *48*(2), 63–80. https://doi.org/10.1007/BF02313401

Joy, J., Rajaram, V., Aditya, A. R., & Pandimurugan, V. (2023). Developing Advanced Software Keylogger using Python and Creating Awareness of their Functionalities. *2023 2nd International Conference on Automation, Computing and Renewable Systems (ICACRS)*, 1551–1557. https://doi.org/10.1109/ICACRS58579.2023.10404996

Kapare, R., Gawade, A., Kamble, A., & Tembhurnikar, P. (2024). *Enhancing Digital Security With The Advanced Keylogger Project*. *12*(3), j617–j624.

Laborde, P., Costiou, S., Le Pors, É., & Plantec, A. (2020, December). *15 years of reuse experience in evolutionary prototyping for the defense industry*. International Conference on Software and Systems Reuse, Hammamet, Tunisia. ffhal-02966691f.

Mbanaso, U., & Dandaura, P. (2015). The Cyberspace: Redefining A New World. *Journal of Computer Engineering (IOSR-JCE*, *17*, 2278–2661. https://doi.org/10.9790/0661-17361724

Nafish, A., & Rengarajan, A. (2024). The Effects of Technology evolution on Cybercrime. *International Journal of Innovative Research in Computer and Communication Engineering*, *12*(02), 1060–1066. https://doi.org/10.15680/IJIRCCE.2024.1202056

Nagathota, J., Kethar, J., & Gochhayat, Ph.D., S. P. (2023). Effects of Technology and Cybercrimes on Business and Social Media. *Journal of Student Research*, *12*(4). https://doi.org/10.47611/jsr.v12i4.2284

Navarro, J., Naudon, E., & Oliveira, D. (2012). Bridging the Semantic Gap to Mitigate Kernel-Level Keyloggers. *2012 IEEE Symposium on Security and Privacy Workshops*, 97–103. https://doi.org/10.1109/SPW.2012.22

Nordin, A., Rusmi, A., Mutalib, M., Suhaizad, F., Burhanudin, R., & Khamis, N. (2018). Development of Requirements Pattern Repository: Towards Supporting Requirements Reuse. *Advanced Science Letters*, *24*, 1847–1851. https://doi.org/10.1166/asl.2018.11174

Olzak, T. (2008). *Keystroke logging (keylogging)*.

Santoki, H. P. (2014). *Design and Implementation of Detection of Key Logger*. *2*(2), 1999–2017.

Shao, P. E., & Dida, M. A. (2020). The Implementation of an Enhanced EFD System with an Embedded Tax Evasion Detection Features: A Case of Tanzania. *Journal of Information Systems Engineering and Management*, *5*(1). https://doi.org/10.29333/jisem/7824

Shirke, A., Pawar, R., Bivalkar, M., Waghela, H., & Shah, Z. (2023). Advance Keylogger – Capturing Keystrokes. *2023 6th International Conference on Advances in Science and Technology (ICAST)*, 250–255. https://doi.org/10.1109/ICAST59062.2023.10455057

Susanto, A., & Meiryani. (2019). System Development Method with The Prototype Method. *International Journal of Scientific & Technology Research*, *8*(07), 141–144.

Zant, R. (2005). Hands-on prototyping in system analysis and design. *Issues in Information Systems*, *6*.

Appendices

1. Gannt Chart



1. Marking Guide

Strathmore University

School of Computing and Engineering Sciences

Project Proposal Assessment Guide

|  |  |
| --- | --- |
| Student Number: | 150211 |
| Working Title: | Triskelion: A Python Based Software Keylogger for Vulnerability Assessment in the Cyberspace |

|  |  |  |  |
| --- | --- | --- | --- |
| Evaluation Areas | Weight | Score | Notes |
| Title page:  Informative, concise, and appropriate | 2 pts |  |  |
| Abstract  To have background, problem, solution, methodology (approach data and tools) outcomes and expectations | 2 pts |  |  |
| Introduction  Background (2)  *A clear illustration of issue, context and audience*  Problem Statement (2)  *Pain points, audience, who is affected and how solution comes in to fix the pain.*  Objectives (S.M.A.R.T and Linked to Problem Statement) (2)  Research questions (1)  *Alignment of questions with objectives*  Justification (2)  *Should be research supported.*  Scope of Project (2)  *Specify boundaries of people process, HW/SW, data etc.*  Limitations (1)  *Challenges Expected*  Delimitation (1)  *To do to counter anticipated challenges.* | (13 pts) |  |  |
| Literature Review/Related Work  Objectives mapping to Literature Review (2)  Critique of Theoretical framework and content  adequacy (2)  *Principles, parameters of consideration*  Discussion of technologies contextualization for the proposed work (2)  Citations of content and alignment to work (2)  Review of at least 3 systems comprehensively the working behind it (2)  Gaps identification, analysis relative to the  proposed solution (1)  Conceptual Framework clear to communicate how it works, data flows, processing, actors (3)  *Diagram that’s clear; discussion of diagram.*  *Describe input process output storage boundaries.* | (14 pts) |  |  |
| Methodology  Methodology and justification (2)  Correct Methodology Application (1),  Design and Development tools (2)  Deliverables and milestones (2)  Examinable bits from ideation  Proposal, design, test cases documentation doc  Proof of concept- modules  Gantt Chart that makes sense relative to the  project (1) | (8 pts) |  |  |
| Proposal Presentation  Table of Contents and List of Figures (2)  Are relevant references provided and formatted correctly? (2)  Is there a clear and proper use of language? (1)  Effective report structure (chapters and sections) and layout (2) | (6 pts) |  |  |
| Total Marks | 45 |  |  |

Verdict (Please tick) Accept Reject

Comments (Reasons for Reject/Accept)