Tampereen yliopisto

Toni Blåfield

DIFFERENT TYPES OF KEYLOGGERS

Mitigation and risk relevancy in modern society

Faculty of Information Technology and Communication Sciences
Bachelor's thesis
May 2020

ABSTRACT

Toni Blåfield: Different types of keyloggers – Mitigation and risk relevancy in modern society
Tampere University
Computing Sciences
Bachelor's thesis
May 2020

After computer networking through the Internet became popular, people decided to use mainly username-password combinations for their way to authenticate against different services. As a result, a significant tool was developed to attack this manner, a *keylogger*, to precisely collect this information and use it on behalf of the true owner.

There have not been too many researches and studies on this topic for a while, especially with same the research questions, therefore in this thesis the gap between the time from the previous researches up to date is researched. Some relatively new keylogger mitigation methods are introduced in this thesis.

The first purpose of this thesis is to give a detailed explanation of different types of keyloggers and their basic functionality, main targets, main use cases, and increase general awareness of the threat of a keylogger and the form of a keylogger in modern society.

After achieving a basic understanding of the previous aspects, the second purpose of this thesis is to introduce a few methods to mitigate these threats by their type, respectively. This thesis will also evaluate the relevancy of these threats in modern society. Finally, this thesis will conclude with a pervasive analysis of the relevance of these threats caused by keyloggers in the modern information security context.

In this thesis it is concluded that there are types of keyloggers that are not notably dangerous anymore, due to increased device mobility and smaller, handheld size of mobile devices. Consequently, physically visible keyloggers, like hardware keyloggers, are considered less as a threat. Despite that, there are new, modern, and remotely controlled mobile software keyloggers that can spread and collect keystroke input data without any level of visibility to the end user.

Keywords: keylogger, information security, hardware keylogger, software keylogger, mobile keylogger, authentication, multi-factor authentication

TABLE OF CONTENTS

ABBREVIATIONS

KL	Keylogger
OS	Operating System
SW	Software
HW	Hardware
MFA	Multi-Factor Authentication
2FA	Two-Factor Authentication
CPU	Central Processing Unit
RAM	Random Access Memory
USB	Universal Serial Bus
SMS	Short Message Service

1. INTRODUCTION

There has been a vast improvement in the user authentication processes over the past decade. Still, quite many authentication systems include text password inputs as the main way to authenticate into the system.

The problem in this specific authentication method is that using only a keyboard input for identifying a user has become a lot less secure since it is quite easy to break through this kind of authentication as an attacker.

There are many ways to accomplish this attack. For example, an attacker could just guess all the inputs, like username and password, and then get into the system, impersonating the user. The attacker could also try to convince the user to give them their secret information. Especially, the attacker could capture the user's input *during* the authentication process, to obtain the critical details required for the access. To achieve this, the attacker could use a *keylogger*.

This thesis aims to answer these questions: what is a keylogger? What types of keyloggers are there? How to recognise a keylogger? Are there any countermeasures to keyloggers and if there is, which are them and how do they work? Are keyloggers still a threat? What are modern keyloggers like and which forms do we encounter them in?

This thesis aims to properly introduce and explain the usage of countermeasures against keyloggers, and study if keyloggers are still a relevant threat to users today and in the future or have the countermeasures evolved to a level that makes keyloggers inefficient in malicious matter.

The first chapter will give a detailed introduction into keyloggers, different types of keyloggers, their typical appearances, and typical functioning. The second chapter will expand the understanding of the reader several ways keyloggers are deployed, monitored, and how the gathered data is collected and extracted. The third chapter focuses on mobile devices and keyloggers for different mobile platforms. The fourth chapter will discuss the countermeasures to keyloggers and their functionality. Finally, the fifth chapter concludes if keyloggers still persist a notable threat to users or not and justifies the deductions found.

2. INFORMATION ABOUT KEYLOGGERS

2.1 Types of keyloggers and their basic concepts

In general, a keylogger is a tool that tries to capture all the keystrokes of a device that it is installed to, providing the target device has a physical way to input textual data, for example, through a keyboard. Keyloggers are primarily used without the user's knowledge. Moreover, keyloggers are also used legitimately to track, for instance, children's actions on the device or to supervise the company's computers for misuse. [1]

Nonetheless, keyloggers are many times used, as stated in the introduction, criminally, to gain access to someone's private resources and then use them to the attacker's advantage instead of the initial legitimate and official purposes of a keylogger.

Keyloggers may appear in various forms. Keyloggers can be divided into two main categories: *hardware keyloggers* and *software keyloggers* [17]. There are other categories for keyloggers, but these are the two most important ones that make sense in the basic keylogger theory. This has also been the most common division in related studies on this topic.

Hardware keyloggers are keyloggers that operate on a separate hardware device. Usually hardware keylogger devices can be hidden or injected into the target machine in a way that it is hard or practically impossible to detect the physical existence of the keylogger. Hardware keyloggers are usually masked as hardware peripherals that share the common hardware interfaces like USB or PS/2, matching the input device interfaces [18] [19]. Typically, the hardware keylogger has an internal memory that it uses to store the keystrokes internally and independently [17].

Software keyloggers are software programs or scripts existing and executing inside the operating system of the target machine, specifically a computer in the following sections, and they are depending on the host OS of the computer. Similarly, the hardware keyloggers are created in such a way that they are very well hidden from the end user(s) of the target machine. Software keyloggers, unlike hardware keyloggers use the resources and memory of the host OS and are relying on the OS file system, for example.

Both types have their advantages and disadvantages, both of which are processed later in this chapter. It is also possible to use both in combination of multiple sources of a keystroke input device or even multiple keystroke input devices separately.

2.2 Hardware keyloggers

Hardware keyloggers are hardware devices that are placed between the input interfaces of a computer machine and the input device, typically a keyboard, interfering with the electric signals originating from the input device, going towards the computer. [2]

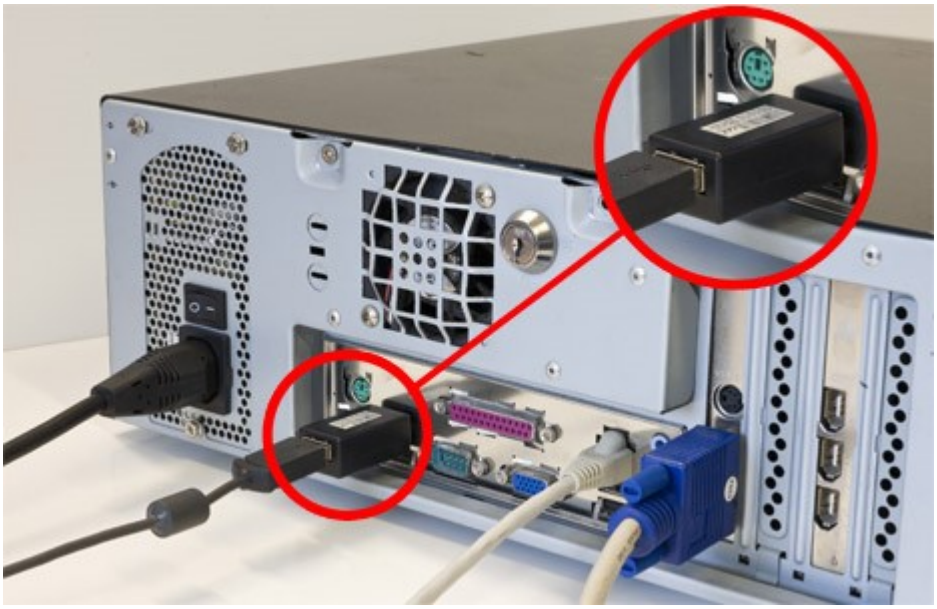

Figure 1: A typical example of a hardware keylogger is a USB hardware keylogger installed between the motherboard of the computer and the keyboard USB cable. For a regular computer user, it can be very hard to detect the keylogger just by a glance at the computer. Source: Wikimedia. [5]

Hardware keyloggers can be divided into *active* and *passive* ones, both of which able being either self-powered, containing its own power source and not depending on the host, or using the host/target as their power source. [2]

Active hardware keyloggers are devices that are connected in series between the computer and the input device, repeating the input signals. The active keylogger seems to be the most common commercial hardware keylogger type. A passive keylogger is, in contrast, connected in parallel between the computer and the input device, only observing the state of the line between those two without emitting any signals on its own. [2]

In addition, a division between *evasive* and *stealthy* hardware keyloggers can be done when analysing the detection of the keyloggers. A stealthy keylogger's main strategy on staying undetected is to keep as low profile as statically possible and once detected, it takes no further actions to maintain its disguise. An evasive keylogger, instead, can take extra measures against its detection. [2]

Hardware keyloggers can be harder to detect in general than software keyloggers [18]. Since they are in the hardware layer, hardware keyloggers are usually beyond the software detection algorithms and software like antivirus programs. Then again, hardware keyloggers are harder to deploy than software keyloggers, since they need both physical presence and an ability to physically install the keylogger hardware to the target system without being detected during the process.

One other thing that needs consideration with hardware keyloggers is the final delivery of the collected data, collected keystrokes, to the collector to be analysed. Since it is dedicated hardware, it would need a lot of additional hardware and software to be able to run the deliveries remotely.

If the result delivery would be done remotely, one way would be over a common communication method, like WLAN or Bluetooth, or over another type of a wireless connection [19]. This would require additional hardware to transfer the data. Transferring over the internet would be even more complicated, since the keylogger would also need a proper way to connect to a network and a reliable protocol to transfer the data. But in theory, it would be possible.

A more traditional way to do the transfer is to physically detach the keylogger after a reasonable time period and then read the contents from it. Considering these limitations, the most typical places where hardware keyloggers would be used, are places and computers with public access and without proper surveillance, for example public libraries or public offices or official departments and buildings like schools and universities [19].

2.3 Software Keyloggers

In recent years, the quality of software, hardware capabilities, and internet bandwidths have improved enormously. Consequently, software keyloggers have become more popular than hardware keyloggers for the sake of having better portability, usability, controllability, and disposability than hardware keyloggers in general. The most important reasons for choosing a software keylogger over a hardware keylogger are the easy deployment of the keylogger and automated collection of the results over the internet.

There are several types of computer malware that are, in general, considered harmful computer software, compared to typical software behaviour. For example, there are viruses, trojans, and spyware as subcategories of malware. Software keyloggers are typically considered *spyware*.

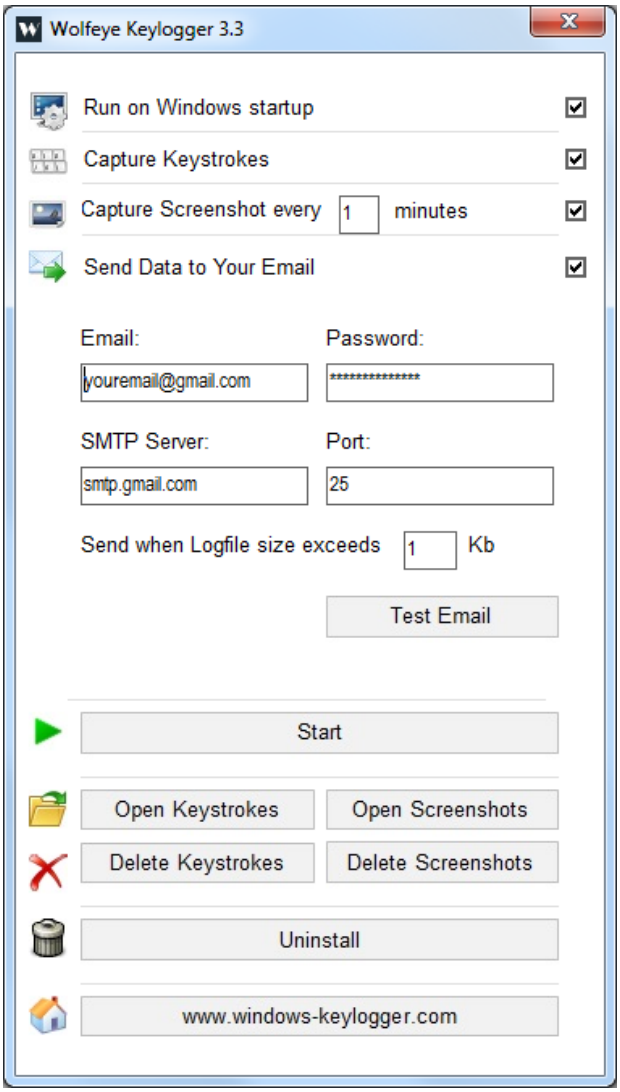

Figure 2: A screenshot of the graphical user interface of a very typical software keylogger. This specific screenshot is from Wolfeye Keylogger. The functionalities of a SW keylogger can vary a lot, depending on the product. Source: Wikimedia. [6]

There are multiple ways to distribute software keyloggers. The most usual way nowadays is to inject the keylogger inside another, a legitimate program or another virus as a carrier program, and then wait it for to be downloaded and installed by the victim and then wait for the keylogger to activate, to begin further actions. [3] This way, the distribution of the keylogger can be automated and the distribution scale becomes huge. These kinds of keyloggers, once detected, are identified, without an exception, as pure malware.

Another way is to have either physical or remote access to the target machine and then manually install the keylogger in a way that the keylogger won't be detected by any automated detectors or anti-keylogger programs, leaving the detection totally for the end user(s).

The easiness of receiving the collected data is based on the fact that software keyloggers lay on the software layer of the machine, inside the OS. With enough privileges and clever algorithms, the keylogger can, after enough collection or periodically, send its payload through an internet connection to the attacker, without any level of detection.

The availability of the tools and mechanisms in the software keylogger depends on the specific keylogger. In Figure 2, the example keylogger interface provides the following options: launch the keylogger at operating system start, capturing keystrokes, capturing screenshots, sending the collected data to an email address, using specific email server details to send the email, including a testing functionality. There are also separate buttons for viewing and deleting the collected inputs and screenshots manually.

The option to run on OS boot is useful when the keylogger software is used for a longer period and is not monitored by the person who is managing the keylogger. This way, it is ensured that the keylogger can be always active, no matter if the OS was rebooted in between the lifetime of the keystroke collection process.

This specific keylogger in Figure 2 can also be set to collect either plain keystrokes, or screenshots of the monitor screen, or even both. The screenshot capture feature can be useful to be able to determine the context of the keystrokes. For example, in terms of the authentication process, sometimes the service that is being accessed, is not available only by looking at the keystroke input. Thus, capturing pictures of the screen in the moment of inserting authentication credentials can help to determine the purpose of those credentials. In addition, there might occur additional useful information about the authentication process that is visible only by looking at the current screen.

In the specific keylogger, there is a method to deliver the payload from the keylogger remotely, in this case, by email. The frequency of the data collection can be adjusted by the size of the collected information. Presumably, the logs reset when the delivery is made, and the collection process loops from the beginning. The email delivery service requires some additional configuration. Besides the receiving address and password for the email, the mail server details must be known to the keylogger to be able to send the collected data.

3. KEYLOGGERS ON MOBILE DEVICES

3.1 About mobile devices and keyloggers in general

In this section, *mobile devices* are considered as modern, handheld electronic devices, specifically computers, sharing the basic capabilities, software and hardware with traditional personal computers. For example, they include a mainboard, CPU, RAM, and some persistent storage. The operation of the device is provided by an OS running on the device. [7] The typical input of a mobile device is touchscreen and all the user input are passed to the device by touching the screen. The main types of mobile devices that are in interest for keyloggers by their advanced functionality are either *smartphones* or *tablets*. For example, services like banking services or social media services are available for use on mobile devices.

These specific modern mobile devices in wider use have existed only a relatively short time compared for example to personal computers. Additionally, mobile device development has been very rapid when compared to personal computers. Therefore, keylogging on mobile devices is quite a new concept. Nonetheless, the menace of keyloggers has already reached also mobile devices. There are some studies on this field and a couple of real-life examples of cases that have occurred in keylogging on mobile devices [8] [9].

There are very few physical input interfaces in a mobile device easily reachable by the user, typically a single USB charging and data connection port and optionally an audio jack port for wired input/output audio devices. These interfaces are constantly visible to the device user. As we know about the visibility and portability issues of hardware keyloggers (Section 2.2), it is practically impossible to use pure hardware keyloggers on a mobile device to successfully gather enough user input from the mobile device to be able to commit any useful actions. Therefore, only software keyloggers are considered a notable threat on mobile devices, thus the only category being analysed in this chapter.

Then again, on the software side, there are numerous techniques and exploits for keylogging. Since there are a lot of different mobile device manufacturers with a lot of different models that have their customised OSes and customised security features, there are multiple possibilities for attackers to try to get their hands on.

3.2 Mobile operating systems

Android is the most popular mobile device operating system on the market with its market share on mobile phones over 70 percent in January 2020 [10]. Android uses a customised version of the open-source *Linux* kernel, being a variation of a Linux operating system, that is a *Unix*-like operating system. It has been controversial if Android is truly a Linux operating system or not [23]. There are many OS distributions for desktop and laptop computers that use Linux kernel or its variant [24]. Android is highly customisable and freely distributed; therefore, it is highly used by many mobile device manufacturers, thus increasing the amount of the total usage of the OS.

The second most popular operating system with a 24 percent market share is *iOS* (iPhone Operating System), created by Apple [10]. IOS is, in contrast to Android, fully proprietary, designed to run only on Apple's mobile products. There exist many other minor OSes but since their combined market share is less than one percent, they will not be discussed in this study [10]. In summary, there are two major OSes available for mobile devices, one relies on openness and customisation, the other on closeness and strict operating policies.

Since Android is used in the majority of mobile devices and uses source code available to anyone for reading, Android possesses a greater risk to especially zero-day vulnerability attacks than for example iOS. A zero-day vulnerability means a defect in the software that is already discovered but note yet been fixed or patched, in consequence, allowing attackers to exploit that vulnerability [11]. Android allows the user to easily bypass many security features, thus making it less secure from the user perspective. IOS again has strict rules on both the development side and the user side. IOS users can download external applications only from the official application store and many security features cannot be bypassed without special knowledge or special privileges on the system. In the next subchapters, it is discussed more in detail about how these weaknesses allow exploiting keyloggers.

3.3 Keyloggers on third-party keyboards

As mentioned in the earlier chapter about Android customisability, it can truly enhance the user experience of the system if it is possible to change the appearance of the on-screen keyboard. However, this has its downsides too. In a related study, it was discovered how easy it is to inject keyloggers to third-party keyboards [12]. For iOS devices, it would be a lot harder to apply this scenario, since the limitations of the system prevent

the user from changing the keyboard in an insecure manner and as mentioned in Chapter 3.2, installing third-party applications from untrustworthy sources is not directly possible.

The keylogger application in the study was systematically collecting user input from anywhere there was form-based input, e.g. from web browser websites. In addition, the keyboard with keylogger was published to the Android official application store, Google Play Store, and passed all the automated tests and finally got successfully published to the application platform even though it contained a keylogger in it and was masked as a keyboard application. None of the virus scanners was even able to detect the keylogger in the keyboard application. [12]

The keylogger keyboard application created in the study used only *internet permission* of the Android system [12]. In Android, there are defined a set of different permissions for applications that they can require from the user for allowance. For example, a map application could use location permission, or camera application would use the camera permission. The typical Android application permissions are divided into basic permissions and special, dangerous permissions. [13] User can explicitly choose which special permissions are enabled for an application, but the user cannot explicitly disable basic permissions, like internet connection or access to the user input, but instead, these permissions are required in the application installation process for the application to be installed in the first place.

The other problem that truly made this exploit efficient, was the inefficiency of the automated tests in the validation process for publishing on the official application store [12]. Many times, people blindly trust the applications that are published in the official app store. This is understandable since it is, for Android and iOS, the only official way to install third-party applications and these stores are in most cases pre-installed on the device containing either of these OSes. Thus, the acceptance and publication process for new apps should only accept highly qualified applications that would not contain any malicious code. This was also mentioned in the related study [12]. Nonetheless, it is impossible to completely ensure this only by automated verification procedures.

3.4 Touch-based keylogging

Considering the fact that the majority of user input on mobile devices, especially confidential input like passwords, key codes, etc. are provided by touching the screen and almost every modern mobile device has a touchscreen, it should be considered as the primary channel of interest for keyloggers. Though it is simpler to collect input data di-

rectly using the keyboard software (Sect. 3.3), a more general and unified way of keylogging would be through the collection of touch input and providing the input keystrokes by analysing the touch input with various techniques. The problem in this approach is the availability of the touch information as it might not be available for every program and every user level. The more general term for this approach is *touchlogging* [14].

Touchlogging could also be used for legitimate and helpful purposes, for example learning the user's touch habits, thus preventing unauthorised access by unique touch behaviour. However, the significant threat with touchloggers would be the utilisation of them as keyloggers. One approach, that was also presented in the related study, for receiving keystroke data by touchlogging would be by using the knowledge on the typical position of soft keyboards, which is usually at the bottom of the touchscreen. [14] The positions of the soft keyboard can vary, depending on the keyboard software, the size of the keyboard keys used, and some other user-based customisation. However, this issue could be mitigated by creating a set of most common keyboard configurations and using all of them against the received touch data, filtering out the meaningful results that make sense in the keystroke data, and that way also figuring out the current device-specific configuration.

As it was demonstrated in the related study, successful implementation of a touchlogger for an iOS mobile device requires a few additional security bypasses to be fully functional: an administrative (root) access to the device and override internal system-level methods and the touchlogger software is required to constantly run on the background. To gain root access on an iOS device in the first place, the exploitation of an existing vulnerability in the OS must be done. This process is called *jailbreaking*. [14] When these requirements are compared to the requirements in third-party keyboard keylogging on Android systems (Sect. 3.3) it is notable how much more effort and additional indirect requirements these requirements generate. For example, most of the practically useful jailbreak exploits require physical access to the device [15] [16]. Considering the target devices are mobile devices, it is quite a heavy requirement for the attacker.

4. COUNTERMEASURES

4.1 About countermeasures

In their research, Gerdes and Mallick noted four main method categories of countermeasures on keyloggers, being *avoidance, detection, exhaustion,* and *obfuscation* [2]. These are the high-level prevention methods for any type of keyloggers and should be considered in their general meaning as the basis for developing any countermeasures.

Avoidance means simply the usage of alternative methods of text inputs in a way that the keylogger eventually collects no meaningful data to the user. For example, using on-screen keyboards instead of the physical one is one way to avoid a possible keylogger.

The detection means that the user takes additional actions in terms of trying to constantly detect if there is any kind of keylogger being used during the operation of the machine to detect possible keyloggers and mitigate them. This is very practical in case there are hardware keyloggers used in the system.

Exhaustion is a method to fill or overwrite the memory of the keylogger with garbage, destroying any valuable information on the keylogger. This, however, might not work on some modern keyloggers containing program code that can take this countermeasure action into account.

Obfuscation means that the true input is tried to be hidden using e.g. flood of random values or some generated noise around the real data. By using this method, the keylogger might be left, again, without any valuable information, or at least the real information is harder to be analysed from the data. This method works on the inline keyloggers of hardware keyloggers that repeat the signals. [2] Therefore, this method is not easily applied to keyloggers which are purely software-based, and who read the clear final values of the inputs completely digitally.

In table 1 is summarised some general countermeasures and analysed their effects on different types of keyloggers. These results are not completely valid in every possible case but they try to give a direction of which to follow when encountering a type of keylogger and as a checklist to measure which countermeasures should generally be considered when creating a mitigation plan for the specific keylogger scenario. In table 1, it is assumed that mobile keyloggers are purely software keyloggers, based on the previous analysis results (Sect. 3.1).

Table 1: Analysis of countermeasures on different keylogger types.

Keylogger type **Countermeasure action**	**Hardware keyloggers**	**Software keyloggers (on Personal Computers)**	**Mobile keyloggers**
avoid mistrustful file sources		X	X
download external applications only from the official application sources		(X) if both application and official source is available	X
(regular) checks on software components and filesystem		X	X
(regular) checks on hardware components for anything suspicious	X		
avoid shared or public devices	X	X	(X) if mobile device is shared with a potential attacker, then works, but this is rare
avoid inserting mistrustful devices into own device	X if the device itself is a hardware keylogger	X if the device is a data storage	(X) if the device is data storage and automatic access is enabled

use multi-factor authentication	X	X	X if there is an external device used, for total security

There are a few combinations in table 1 that are not explicit, but they require some constraint to be secured. For example, personal computers usually do not have a unified way to install applications from, controversial to mobile devices. Thus, it is common that even though containing an official application source, it is very easy to download applications from third party sources. Some applications are not available on the official source, forcing the user to acquire those applications from potentially dangerous sources. Also, multi-factor authentication assumes the user has some external device for securing the mobile device. If the single mobile device was solely used for the authentication process, for example by using one application to authenticate and some other application to commit the second factor within the device OS, it would possibly endanger the authentication target.

4.2 General countermeasures on hardware keyloggers

The best weapon against hardware keyloggers is general awareness and vigilance. Users should constantly check their hardware for any modifications or changes and see if there is any keylogger installed. Usage of public and physically unprotected computers that are not under decent surveillance is always a risk that should be minimised by avoiding the usage of these computers.

The typical places for a hardware keylogger in a computer are on the backside of the computer device. However, the true location depends on where the connection to the textual input device is located [5]. The best place to start looking for them is to simply check where the cable from the input device is connected to the computer hull. After figuring out the cabling, it is advisable to remove the cable connection and carefully inspect the interfaces in the cable end. If the cable contains any removable components except for the cable connection to the computer itself, that are not required for the cable to be inserted to the ports of the computer motherboard, for example USB ports, these external components might happen to be hardware keyloggers.

The physical access to the computers the user is using to access important services, need to be secured with strong enough access control. In other words, it should be taken

care of who can physically access the computers. The amount of those people should initially be as low as possible and only trustworthy, well-known people to the owner or user of the computer.

In modern society, this threat is slowly moving away by itself, since the devices we use for sensitive services are constantly moving to more and more mobilised devices, thus making it significantly harder for hardware keyloggers to have a foothold on. Also, the number of hardware peripherals is decreasing as cloud computing is expanding and evolving, and different types of hardware connections are becoming more and more wireless. Moreover, we are slowly moving off the trend of having physical external devices but instead using and controlling them wirelessly and remotely.

4.3 General countermeasures on software keyloggers

As mentioned in the previous chapters, software keyloggers might be harder to detect physically, since they exist only in the software and might be very well hidden. Therefore, a good way to detect and avoid software keyloggers, is using another software for that task. There is plenty of antivirus software that is capable of detecting maliciously received keyloggers and prevent their functioning. [4] The problem in antiviruses is that they are not able to detect keyloggers that were installed in a non-detectable way (Sect. 2.3). Using only this method, these types of keyloggers are left undetected.

In addition to the previous method, general awareness, like in the hardware keylogger case (Sect. 4.2.) is a great aid in protecting oneself from software keyloggers. Using only trusted, verified and widely accepted sources for any file downloads, not opening email attachments from untrustworthy, or even at least suspicious sources are ways to improve one’s security.

The threat of software keyloggers is focusing more on the mobile development side (Sect. 3, Sect. 4.4). Thus, it is important to focus on mobile devices and their specific operating system and software behaviour more than traditional personal computer technology. There are numerous ways to fight against mobile device keyloggers, and these methods are discussed more thoroughly in Section 4.4).

4.4 Countermeasures on mobile device keyloggers

To mitigate the threat of mobile device keyloggers, especially those mentioned in chapter 3, necessary actions are required from both mobile device users and developers. Both perspectives are discussed in this section. Mobile device and mobile software developers need to focus on secure architectures and implementations while end users need to

ensure they are using the devices securely and maintain general security awareness (Sect. 4.2, 4.3).

One way to protect users from keyloggers relying on remote machine payload delivery is to develop software that tracks and detects such malicious transfer of sensitive input to third party servers, warning the user and/or the OS about these malicious events. These types of applications are using information flow tracking to detect such behaviour [12]. Consequently, the user or the system can take further actions to mitigate the keylogger in action. It must be considered, though, that any automated and sophisticated detection applications will require more resources from the underlying system [12].

To prevent keylogging based on third-party keyboards, it is advisable to carefully consider if it is truly necessary to acquire third-party keyboards in the first place. In such cases, avoiding the use of third-party applications in situations where native or preshipped applications are available and mostly fulfil the user requirements, it would be preferable to avoid third-party keyboards, especially when processing sensitive input.

However, if it is unavoidable, only trusted, widely used and highly rated third-party keyboards that have been in public distribution for a longer time should be acquired [12]. Also, only official third-party application providers, like Google Play Store for Android or App Store for iOS should be used for acquiring third-party keyboards. When choosing the third-party keyboard, it is also important to check a couple of parameters: who has published the keyboard application? has the publisher published other applications that are relatively popular? when is the keyboard published? how many downloads the application has? what reception has the application received? To prevent malicious third-party applications ending up on official application stores, the application validation processes must be improved and enhanced [12]. In addition, the post-processing of published application is required, to detect afterwards if applications have slipped through the pre-validation and review processes.

As discussed the threat of touchlogging (Sect 3.4), it is relatively hard to deploy in practice and is quite well protected by the development side. However, by the user side, it is still possible to fall under the victim of this type of attack if the security of the mobile device is not taken seriously enough. The best mitigation for this kind of attack or elevated attacks, in general, is to simply maintain enough security. For example, using passwords and passcodes to restrict access to the device and its resources, allowing both physical and logical access only to trustworthy parties, with sufficient monitoring by the administrator of the device, would protect the user from touchlogger attacks.

4.5 Multi-Factor Authentication

One of the most promising tools so far, published and taken into wide use in online services quite recently, not only against keyloggers but generally improving the security of authentication, is *Multi-Factor Authentication* (MFA), specifically *Two-Factor Authentication* (2FA). The definition of authentication factors is: “something that you know”, “something that you are” and “something that you have”. Multi-factor authentication means using more than one of these three concepts, in practice. The typical combination that is necessary to be analysed in the following cases that are relevant to the keyloggers, is “something that you know”, which means, for example, the combination of username and password. The other factor then is typically combination or exclusively “something that you have”, meaning a second device and “something that you are”, meaning usage of a biometrical function in the authentication process. [20]

Two-factor authentication is a special case of multi-factor authentication, where there are only two factors used, the main factor which is the main device used for the login and the second factor which is a second independent device providing additional security, in the case of having “something that you have”. The basic concept of multi-factor authentication is that there is another device used as part of the authentication process and partially as an external randomness source.

When registering to a system, the user can specify a personal device, typically a mobile device i.e. personal smartphone, tablet, or another handheld device that is frequently carried with the user. There is an authenticator application involved, which is first downloaded and installed to the device, and then the specific system or service is activated in the application, using a secure token. [21]

Consequently, the next time the service requires authentication from the user, it asks for a random-generated sequence, usually consisting of numbers, that is typically short enough to be memorised but long enough not to be guessed or brute-forced and it is usable only once. As a final rule, it is invalidated immediately after use or after some timeout period, if not used.

If the implementation of the MFA fills all the requirements, it may be considered as an extremely secure authentication manner. Providing that the keylogger has collected the basic authentication information, such as the username and password for the specific service, it would still be unable to access the service without having full access to the separate device for the very short timeline the one-time password is valid, which can be for example 30 seconds [21].

Even if a keylogger would reach all the devices that were used in parts of the authentication process, multiple factors would stop the attack. The invalidity of old authentication codes, the uniqueness, and randomness provided, the entropy used for the randomness being secure enough and the very short lifespan of a single-use authentication code are factors that provide the absolute security in mobile 2FA. No one would be fast enough to receive, organise, and extract the key information for the authentication until it would be too late.

Using the authentication correctly, meaning that the second factor is another independent physical device, and the code is read from another device optimally at the same time as being written in the first device, it would be a practically impossible task for a single keylogger to record and send the one-time-usable code as the input device used to input the singe-use code would be attached to a different machine than the first location of the generated code. Simply put, the attacker would need to have two, separate keyloggers inserted into both devices at the same time, listening to both keyloggers in real-time, which can practically be considered extremely implausible.

Challenges in multi-factor authentication are the additional requirements from users, which themselves require additional time and money [20]. To successfully configure an MFA system, one would be required to have another separate smart device that contains its operating system, cryptographic devices, and a way to connect with the service that is being used to authenticate into. These devices cost a lot of money. To avoid using expensive devices, simpler methods, like *Short Message Service* (SMS) messages could be used as the second factor. However, SMS is proven to be a very insecure method for delivering secrets and using SMS as the only other factor could endanger the whole authentication process [22].

The second requirement from the user is to allocate enough time to correctly setup and configure all the steps for the MFA system, which include installation of an authentication application to the other factor, connecting the authenticator application with the target service, to be able to provide the authentication through the application.

Even when all of this configuration steps are completed, there will be additional time needed in every authentication attempt, since the authentication process requires both the physical proximity of the other factor, available to the user through the authentication process and the user navigating to the authentication application to receive the single-use token.

For some people, these requirements might be too exhaustive, to be used for daily-basis or even more frequent authentication. The existence of MFA might reduce the usability

and accessibility and the authentication speed, thus raising the threshold to use or start using the MFA as part of the authentication.

In larger scaled parties, like companies, government sectors, and other non-individual parties, the usage of MFA might be enforced by the information security policies to ensure everyone handling the company information or accessing critical services in the company, are protected by MFA. This doesn't include individuals, but some very critical services might require using MFA to be able to access the service at all. This might be problematic to people who are not somehow able to frequently access or even afford some other factor besides the knowledge factor, for example.

To this day, MFA, with at least two factors being used, has proven to be an ultimate solution against keyloggers, as there is simply no practically effective way to break it by only capturing user input. In modern society, it is expected that the majority of people using information-sensitive and financial online services possesses external additional devices or other methods to provide multiple factors relatively easily, thus gaining much higher security with simple steps. However, there are still numerous services that do not, at least correctly, support this technology even though it would be possible, potentially jeopardising their users' privacy and assets.

5. CONCLUSION

Hardware keyloggers have traditionally very low deployability, transportability and usability since physical actions are required by the keylogger deployer. The attacker needs to have physical access to the target device when injecting the keylogger, but possibly also when the attacker wants to acquire the collected keystroke data. Technology is constantly moving into more mobilised form. Consequently, using hardware keyloggers becomes more and more unpractical in modern society. However, hardware keyloggers can still be used in very targeted scenarios where the victim is well known to the attacker, and the attacker has physical access to the target device. In these cases, the latest hardware keyloggers can be so compact that these keyloggers can operate in total obscurity. But for the greater audience, hardware keyloggers seem to have lost most of the significance, compared to software keyloggers.

Meanwhile, software keyloggers have evolved from the traditional personal computer keyloggers to a mobile-friendly form, into mobile keyloggers. Thus, it can be concluded that the software keyloggers are an increasing threat, in a new format. Software keyloggers, in general, have much better portability and distributability than hardware keyloggers, for example. Software keyloggers can widely be distributed over the internet and other digital media. They can be deployed in the form of a propagating virus that helps the keylogger to spread even further. Mobile keyloggers can take advantage of the official application markets to be recognized as legitimate applications, therefore being highly distributable, highly undetectable, and extremely efficient. All in all, software keyloggers keep evolving further and their potential capabilities extend constantly, in modern society.

It is clear that without any additional protection or countermeasures, keyloggers provide a significant threat to the user, since they might steal almost any personal and secret information that is, by any means, produced with any physical or logical input device, for example, with a keyboard or mobile device touchscreen. Not only just authentication credentials they can steal, but any other critical personal information, like social security numbers, home addresses, phone numbers, theoretically anything that is provided by keystroke inputs.

There are a lot of possibilities in the countermeasures against keyloggers. Efficient protection from keyloggers is not simple and consists of multiple areas. It requires constant proactivity by the user, especially in the form of awareness and both physical and logical

level actions. Most of the given solutions are not solely enough to protect oneself from keyloggers. Hence, it is required to combine them to enhance the protection. For example, any countermeasure that was designed against software keyloggers, does not, in any way, be able to even detect a physical keylogger, let alone neutralise it.

However, one of the countermeasures introduced was clearly above any other method in terms of protecting user authentication: multi-factor authentication. This method, which is quite recently taken into popular use, seems to be superior over any other countermeasure since once properly implemented, it can protect from any type of keylogger, whilst having its additional requirements and actions needed.

Considered all the previous measurements, it may be noted that, if all these countermeasures, especially the lattermost one, are properly used with each other and general awareness is acknowledged, keyloggers do not pose a threat to the user anymore, expecting that keylogger was the only tool that was used in the attack.

When it comes to modern society, the keyloggers on mobile devices seem to be a possibly progressive threat as they are becoming more popular and the techniques these keyloggers use are constantly expanding, meanwhile the traditional keyloggers made for personal computers have lost their mainstream in the society.

Nonetheless, not using these countermeasures properly or being negligent, may still predispose the user to the risks of keyloggers. All in all, it is heavily dependent on the actions of the user and the service provider, what are the levels of the risks encountered.

REFERENCES

[1] Sbai H., Goldsmith M., Meftali S., Happa J., A Survey of Keylogger and Screenlogger Attacks in the Banking Sector and Countermeasures to Them, Castiglione A., Pop F., Ficco M., Palmieri F., Cyberspace Safety and Security, CSS, 2018, p. 1 (cited 14.11.2019)

[2] Gerdes R.M., Mallick S., Physical-Layer Detection of Hardware Keyloggers, Bos H., Monrose F., Blanc G., Research in Attacks, Intrusions, and Defenses, RAID, vol 9404, 2015 (cited 14.11.2019)

[3] What is a keylogger: A brief on a dangerous and malicious tool, Comodo, Available: https://enterprise.comodo.com/what-is-a-keylogger.php (cited 14.11.2019)

[4] Mateiu M, Keyloggers: What They Are, Where They Come From, and How to Remove Them, AVG Website, 2018, available: https://www.avg.com/en/signal/keyloggers-what-they-are-where-they-come-from-and-how-to-remove-them (cited 14.11.2019)

[5] Hardware Keylogger, Wikipedia, licensed under CC, available: https://en.wikipedia.org/wiki/File:Usb-logger.jpg (cited 20.01.2020)

[6] Keylogger Software, as File, Wikipedia, licensed under CC, available: https://en.wikipedia.org/wiki/File:Keylogger_Typo_Software.png (cited 21.01.2020)

[7] Mobile device, Techopedia website, available: https://www.techopedia.com/definition/23586/mobile-device (cited 20.02.2020)

[8] Keylogger Attacks on Banking Apps Increase, Promon, security news, 2018, available: https://promon.co/security-news/keylogger-banking-apps/ (cited 20.02.2020)

[9] Thomson, I., 'Invisible Man' malware runs keylogger on your banking apps, The Register website, 2017, available: https://www.theregister.co.uk/2017/08/02/banking_android_malware_in_uk/ (cited 20.02.2020)

[10] Mobile Operating System Market Share Worldwide – January 2020, Statcounter, Global stats website, available: https://gs.statcounter.com/os-market-share/mobile/worldwide (cited 20.02.2020)

[11] Zero-Day Vulnerability, Trend Micro website, available: https://www.trendmicro.com/vinfo/us/security/definition/zero-day-vulnerability (cited 24.2.2020)

[12] Cho J., Cho G., Kim H., "Keyboard or keylogger?: A security analysis of third-party keyboards on Android," 2015 13th Annual Conference on Privacy, Security and Trust (PST), Izmir, 2015, pp. 173–176 (cited 24.02.2020)

[13] Android Developer Guide, Android Developers, 2019, available: https://developer.android.com/guide/topics/permissions/overview (cited 24.02.2020)

[14] Damopoulos, D., Kambourakis, G., Gritzalis, S., From keyloggers to touchloggers: Take the rough with the smooth, 2013, vol. 32, pp. 102–114 (cited 28.02.2020)

[15] Goodin, D., Developer of Checkm8 explains why iDevice jailbreak exploit is a game changer, Arstechnica, 2019, available: https://arstechnica.com/information-technology/2019/09/developer-of-checkm8-explains-why-idevice-jailbreak-exploit-is-a-game-changer/ (cited 01.03.2020)

[16] Newman, L.H., Unfixable iOS Device Exploit Is the Latest Apple Security Upheaval, Wired, 2019, available: https://www.wired.com/story/ios-exploit-jailbreak-iphone-ipad/ (cited 01.03.2020)

[17] What's the Difference Between Hardware Keylogger and Keylogger Software, EaseMon, available : https://www.easemon.com/whats-the-differences-between-hardware-keylogger-and-keylogger-software.html (cited 08.03.2020)

[18] Mihailowitsch, F., Detecting Hardware Keyloggers, DeepSec, 2010, available: https://deepsec.net/docs/Slides/2010/DeepSec_2010_Detecting_Hardware_Keylogger.pdf (cited 08.03.2020)

[19] Jablokow, A., Was That Always There? A Hardware Keylogger Threat, Ipswich, 2017, available: https://blog.ipswitch.com/was-that-always-there-a-hardware-keylogger-threat (cited 09.03.2020)

[20] Back to basics: Multi-factor authentication (MFA), Trusted Identities Group, Information Technology Laboratory, Applied Cybersecurity Division, National Institute of Standards and Technology, 2016, available: https://www.nist.gov/itl/applied-cybersecurity/tig/back-basics-multi-factor-authentication (cited 09.03.2020)

[21] What Is Two-Factor Authentication (2FA)?, Authy website, available: https://authy.com/what-is-2fa/ (cited 10.03.2020)

[22] Holmes, D., 6 Ways Attackers Are Still Bypassing SMS 2-Factor Authentication, Identity & Access, SecurityWeek, 2019, available: https://www.securityweek.com/6-ways-attackers-are-still-bypassing-sms-2-factor-authentication (cited 10.03.2020)

[23] Thornsby, J., Is Android really just Linux?, 2018, Android Authority, available https://www.androidauthority.com/android-linux-784964/ (cited 30.03.2020)

[24] List of Linux distributions (with a timeline), Wikipedia, available: https://en.wikipedia.org/wiki/List_of_Linux_distributions (cited 30.03.2020)