Preventing capture of keystrokes in Windows OS.

A dissertation submitted in partial fulfilment of the requirements for the degree of Bachelor of Science (Honours) in Software Engineering

by A. Yousaf

Department of Computing & Information Systems
Cardiff School of Management
Cardiff Metropolitan University
April 2017
Declaration

I hereby declare that this dissertation entitled preventing capture of keystrokes by improving current algorithms is entirely my own work, and it has never been submitted nor is it currently being submitted for any other degree.

Candidate: Azeem Yousaf

Signature:

Date:

Supervisor: Dr Tom Crick

Signature:

Date:
Abstract

This paper focuses on keylogger capturing keystrokes. The system will prevent keyloggers from capturing keystrokes of the users, allowing the user to be assured any login credentials or private data will not be captured. This will make their personal data secure even if their antivirus fails to detect the malicious software.

Acknowledgement
I would like to thank my supervisor Dr Tom Crick who has guided me during my dissertation write up. I would also like to thank all my friends and family who offered me support, helping and ensuring I complete this dissertation on time.
Table of Contents

Abstract .......................................................................................................................... 3
Acknowledgement .......................................................................................................... 3
0. Introduction ............................................................................................................... 5
1.0 Literature review ..................................................................................................... 7
2.0 Methodology ........................................................................................................... 9
3.0 Research .................................................................................................................. 12
  3.1 What are keyloggers? ............................................................................................... 12
  3.2 difference between offline and online keyloggers .................................................. 13
  3.4 Problem ................................................................................................................ 13
  3.5 How antivirus detects malicious software ............................................................. 17
  3.6 Existing solutions .................................................................................................. 18
4.0 Requirements .......................................................................................................... 19
  4.1 Prioritising functional and non-functional ............................................................ 20
5.0 Designing process ................................................................................................... 22
  5.1 Scope .................................................................................................................... 22
  5.2 UML use case ....................................................................................................... 24
  5.3 Storyboard ........................................................................................................... 28
  5.4 Class diagram ...................................................................................................... 29
    5.4.1 SAFEMODE-GUI .......................................................................................... 29
    5.4.2 SAFEMODE ................................................................................................. 30
    5.4.3 Cleaning ........................................................................................................ 31
6.0 GUI ........................................................................................................................ 32
  6.1 Program usage ....................................................................................................... 34
  6.2 Testing .................................................................................................................. 35
  Multiple key logger test ............................................................................................. 36
  6.3 Pros and cons ....................................................................................................... Error! Bookmark not defined.
7.0 Review and conclusion ............................................................................................. 41
  7.1 Evaluation of the dissertation ................................................................................. 41
  7.2 future work ......................................................................................................... 42
  7.3 Summary .............................................................................................................. 42
  7.4 Conclusion .......................................................................................................... 44
Appendix ......................................................................................................................... 45
Keywords ....................................................................................................................... 45
Ethic number ................................................................................................................ 45
Ethic form ..................................................................................................................... 45
Code ............................................................................................................................... 49
  SafeMode_GUI ........................................................................................................ 49
  SafeMode ................................................................................................................ 59
  Cleaning ................................................................................................................... 82
  Cleaning-CL ............................................................................................................ 90
Reference: ................................................................................................................... 92
0. Introduction

The aim of this project is to write a computer program that will be useful for businesses or average users. The software can be used by anyone, from military to organisations and users for personal use. Users will be able to download this open source software to help them increase security on their computer; this software creates a secure desktop mode allowing you to write confidential documentation or to entering their credentials securely. This helps the user to be assured that all hackers’ malicious software is not able to record keystrokes offline and online. This can be very useful for businesses as this can provide an extra layer of security, protecting their customer’s personal data which is enforced by the Data Protection Act 1998.

The Data Protection Act 1998 ensure companies secure their customers’ data and to be stored for only the required period. If companies breach this act they will be prosecuted which will cost them thousands of pounds; this has happened to many companies which will be discussed later in the report. The Data Protection Act 1998 has the following rules; data needs to be used lawfully and fairly, only kept for the intended purpose. Companies cannot keep customer’s data longer then needed and must handle accordingly to the peoples’ data protection rights. Due to the United Kingdom having many people from different ethnic backgrounds, sexuality, criminal record etc. There are stronger laws about this information as it can be used to discriminate against people (Data Protection Act 1998. 2017).

This project was decided to be developed as research shows there was keyloggers stealing credentials of organisations and average users. There were more reports that Windows 10 have keyloggers implemented into the new operating system that was released over a year ago (back in 2016) (Dion Dassanayake. 2017). Security is an extensive field as there are many attack vectors allowing hackers to gain data from organisations or users. This report will not stop the major concerns in security as this tackle a certain area that causes threats to users, this is known as SCOPE. This allows the paper to address within a certain area of the security field attacking the issues related to keylogging.

This paper will be covering literature review in the same field, allowing readers to come up to speed with the existing solutions and research that were carried out regarding keyloggers. It is important to review the literature as it can give us more valuable information about this field. These academic papers are carried out by other researchers. This will improve the readers understanding regarding keylogging and the techniques which were used. The literature review can show us the potential solutions for the same issue that the paper will be tackling today.

After the literature review, the report will be discussing what methodologies will be used for secondary research to the designing methodology. It is very crucial to have a methodology when you are developing a software from the research that will be carried out. The methodology allows us to have a clear idea on what needs to be done to accomplish the research paper goals and objectives.
This report will address the purpose of this dissertation, the real-world problem. The idea of the research paper is finding solutions to current world problems and explaining how their solution solves the problem they stated in their research paper. This is an important part of the dissertation as the reader needs to understand what issues the paper is trying to address. When the paper has stated the problem that is trying to be solved then the paper will move onto the requirements chapter. This section will state what is a MUST functionality and what is a desirable feature. The requirement section will also talk about the MoSCoW method, which will allow the report to prioritise the important features.

The paper will contain the designing process; this section will be about designing the software that is being developed as a solution to the problem. It will contain diagrams and theoretical side of the software. These diagrams will explain the software functionality on paper making it clear to readers on how the software operates. The diagrams included in the paper will be UML class diagrams, UML use case diagrams and storyboards, since it was decided that the paper would use scrum over waterfall methodology for designing and developing purposes.

After the software has been implemented from the design chapter it is very important to carry out testing. This report will include the following “black box testing”, this allows the paper to test the program, finding any possible bugs with the implemented software. The final section of the project the paper will discuss its findings which were based on research and experimentation with code and testing. The paper will also discuss whether the project was successful in meeting the objectives and goals which were set out. The paper will also justify why some choices were made over certain decisions and how they benefit the outcome of this project.

The paper will also discuss whether the solution that will be attempted has solved the problem that is stated in this report. This section is very crucial as evaluation is required to determine if enough critical analysis is carried out. This report will also include future work that can be done by others and how the potential improvements can be made.
1.0 Literature review

Keyloggers have different techniques to capture your keystrokes, these techniques are hooking, kernel-based and API based. “Hooking” is a broad term which covers a broad range of techniques used to alter the behaviour of operating systems or applications. It intercepts notification that a key has been pressed. Many rootkit technologies implement hooking by faking the output of application calls that would otherwise reveal their existence”. The diagram below shows how the hooking architecture works (Howard, Adam; Hu, Yi, 2012).

(Figure 1: https://www.codeproject.com/KB/winsdk/HookSys/Figure_1.jpg)

The second technique is kernel-based keyloggers, these “malicious programs can use technology (also used by rootkits) to subvert the operating system and gain access to hardware. For example, a keylogger could act as a keyboard device driver and intercept data on its way from the keyboard to the operating system” (Howard, Adam; Hu, Yi, 2012).

The last technique is API keylogging, this technique is implemented by hooking to keyboard API, getting the state of the keyboard to provide data. Although these are easy to write it is also easy to detect this malicious software based on the CPU usage but this can be very challenging since a single machine could have many processes running at one time (Howard, Adam; Hu, Yi, 2012).

The current line of defence is signature based, API calls and behaviour based. Signature based is defences like antivirus scanning binaries files for sequence known to identify malicious software, these are known as MD5 hashes or any other security hashes e.g. SHA1. The only issue with this method is antivirus misclassifying a non-malicious binary file as malicious and can lead to deletion of the file (Ortolani, S., 2013). Behaviour based is another
form of detection technique used to identify keyloggers. This type of approach is most successful compared to signature-based, "In layman terms, what is deemed malicious is no longer the sequence of bytes forming a binary, but the set of system interactions caused by it”. However, this approach can be a problem because you need to define what is malicious behaviour, behaviour detection process uses methods like monitoring API calls, memory and data flow to determine malicious behaviour (Ortolani, S., 2013).

Attackers need to introduce the keylogger to the victim computer, for hardware keyloggers this require physical access to hardware. Introducing keyloggers (software) can be done by viruses, mostly known as Trojan horses, the hacker will wait for the user to type login credentials before trying to attempt the login process themselves. The image below shows the process of the attacker, however, this can be time-consuming for the hacker as it entirely depends on the user computer usage.

(Figure 2: http://www.sciencedirect.com.ezproxy.cardiffmet.ac.uk/science/article/pii/S0167404807000569)
2.0 Methodology

This project required a lot of thought, how will this project be approached and how it will accomplish its goals and objectives. By establishing the aims and goals of this paper will give a precise indication of how the paper will be laid out including the development process. For this paper, it was decided that secondary research from journals, books and website were required as primary data search would not satisfy the projects aims within time frame. For this paper, a test will be shown from the software that was developed to tackle this problem, these tests will show some form of proof that it is possible to prevent capture of keystrokes.

Secondary research will consist of information about keyloggers, what it is and how it manages to capture keystrokes. It will also contain important information about events that occurred over the past 5-10 years, these events have caused major problems for organisations and average users. The research will also discuss and test antivirus on their detection rate on keyloggers, show why it was important to conduct this result and development. Secondary research is important to show information and data that has already been researched into, allowing the paper to find references to back up previous works that exists e.g. journals, websites and books.

When the secondary research is done, this project will have developed a program, the program primary function will be preventing capture of keystrokes. Testing will also be done to see the type of effects it has and if it completes its primary objective. Designing process for this report is important as it can help make clear choices on each stage of the development stage. For this report, it was decided that scrum will be the best approach for development. To complete this part of the dissertation an appropriate programming language was required that would be able to complete the tasks. C# was decided to allow the paper to deliver the correct piece of software, there are other languages that could possibly be used. Java is the most popular programming language in 2016 (TIOBE, 2017), although this language is very useful and capable of most development it does not allow the developer to have direct communication with hardware. Java runs all its code in JVM; it is an abstract computer machine that allows computers to run java programs (Wikipedia. 2017). C# allows the task to be completed more easily even though one dependency is being used to help with completing the task. Software development tools are required to write code and compile; it was decided to use Visual Studio 2015 community version. This software is free to download from Microsoft allowing developers to write code up to 4+ languages e.g. C#, C++, Visual basic and F#.
This decision was made due to scrum having a few benefits over other models such as the waterfall model. Scrum model grants early release of the software, showing how the software of the program is progressing, this is done by storyboards and sprints. The waterfall model is a driven model that has phases which does not allow any early release of the program, costing time to fix any errors or changes that need to be changed or fixed (Open View Labs. 2017).

The final stage of this project will be to test the software to see if it delivers its intended purpose, this will be done on a virtual machine as the report will be required to download multiple keyloggers to test the software capabilities. This is the best decision that was decided as malicious software should never be installed on your PC as it can replicate itself and run every time your machine starts up by adding a registry key or duplicating itself into the start-up folder. Some malware is very good at hiding itself as it can potentially hide as a system file.
The software will also have to pass the black box test; the table below has outlined the test which it needs to be passed to be successful.

Table 1

<table>
<thead>
<tr>
<th>Test ID</th>
<th>Test description</th>
<th>User input (if required)</th>
<th>Expected result</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The application can run successfully on Windows OS without any errors occurring.</td>
<td>Not required</td>
<td>Application to start</td>
<td>Application has successfully started without producing errors</td>
</tr>
<tr>
<td>2</td>
<td>Application can load text file</td>
<td>Location of text file</td>
<td>Default text software should run in “SafeMode”</td>
<td>NotePad starts in “SafeMode”</td>
</tr>
<tr>
<td>3</td>
<td>Any web browser starts in secure mode</td>
<td>Location of web browser .exe file</td>
<td>Browser to run</td>
<td>Browser started in secure mode</td>
</tr>
<tr>
<td>4</td>
<td>Prevent keystrokes from being captured</td>
<td>Press keys on keyboard in secure mode</td>
<td>No capture of keystrokes</td>
<td>Keyloggers fail to capture keystrokes</td>
</tr>
<tr>
<td>5</td>
<td>Able to access web browser and text documents</td>
<td>Not required</td>
<td>Web browser and text documents to appear in secure mode</td>
<td>Unable to select both applications</td>
</tr>
<tr>
<td>6</td>
<td>Application throws error if required files are missing</td>
<td>Not required</td>
<td>Shows error</td>
<td>Showed errors, naming the missing files</td>
</tr>
<tr>
<td>7</td>
<td>Application closes when user exit the desired app</td>
<td>User closes web browser or document</td>
<td>Program closes</td>
<td>Program closed successfully</td>
</tr>
<tr>
<td>8</td>
<td>Prevent the user from running the secure mode if a file has not been selected</td>
<td>Not required</td>
<td>If file not selected the run button will not appear</td>
<td>Program prevents user from running secure mode without file selection</td>
</tr>
<tr>
<td>9</td>
<td>User can run task manager in “SafeMode”</td>
<td>User uses hotkeys...</td>
<td>Task manager should appear</td>
<td>User was successful in to getting the task manager running</td>
</tr>
<tr>
<td>10</td>
<td>The software prevents keystroke capture from multiple keyloggers</td>
<td>User needs to press random keys on keyboard</td>
<td>Keystrokes are not captured.</td>
<td>Keyloggers failed to capture any keystrokes</td>
</tr>
<tr>
<td>11</td>
<td>Software runs on more than 1 version of Windows operating system</td>
<td>Not required</td>
<td>Runs on multiple windows operating system</td>
<td>Software has successfully run on windows 7 and Windows 10 OS</td>
</tr>
</tbody>
</table>
After the black box testing, a practical test will be carried out on a virtual machine, testing multiple keyloggers against the developed solution. More than one malicious software will be required to be downloaded, it would be ideal if it is a Trojan horse and an independent keylogger. Both malicious software will be tested at the same time seeing if the program prevents the malicious software from capturing the keystrokes. One of the virtual machine requirements will be Windows operating system (any version) that has net framework 2.0 or above installed, storage and ram will be decided on the day depending on the host available ram and storage.

This will be the final section of the report. There will not be any mythology required for the evaluation of the dissertation. This section will only require us to evaluate the results and test that were conducted. Critically analysing the system and intended purpose of this paper, from there a conclusion and evolution will be written stating the positive and negative, justification and future work as mentioned in the introduction. This section will also contain a summary, summarising what was carried out, for example, testing and the type of data gathered. Summarising will allow the paper to remind the reader of the research layout, research including the problem it is trying to solve and the solution that was developed to solve this problem. A conclusion will also be within this section talking about the findings and how the dissertation managed to solve the problem or not based on the recent test and research that will be carried out.

3.0 Research

Research will be carried out for this project; this will give a detailed background of keyloggers, type of techniques used by hackers and how keystrokes are captured e.g. hooking. The report will also go through the problems it has caused over the past 10 years, naming some major security concerns. This allows the reader to have a better understanding of how this malicious software is dangerous and why organisations are trying hard to prevent their system from being compromised. The report will also state why antivirus is failing to keep keyloggers off our machines due to the detection methods that are being used by antivirus.

3.1 What are keyloggers?

Keylogger is malicious software written by hackers to capture keystrokes from your keyboard, allowing the hacker to have access to any keystrokes stored on your PC or key logs which will get emailed to the hacker. Most keyloggers are known as malware and comes with other features/viruses; the main example would be Trojan horse virus. This malicious code is used for login credentials and identity theft; there are two types of keyloggers, hardware and software. Software keyloggers are more common than hardware; this is because it is easier to install onto the victims’ devices, this can be done by binding the malware to other programs or files. The functionality of the keylogger “runs hidden in the background, making a note of each keystroke you type. The software could scan through the file for certain types of text — for example, it could look for sequences of numbers that
look like credit card numbers and upload them to a malicious server so they can be abused” (Keyloggers Explained: What You Need to Know. 2017).

As written in the literature review, keylogging has three techniques to capture keystrokes, modern keyloggers use hooking or API based. Knowing this makes the process of prevention much easier because we understand the process of how keyloggers can capture keystrokes. Due to this type of information, we can consider how the paper can prevent modern keyloggers from capturing key strokes, the software development will also become much easier.

3.2 Difference between offline and online keyloggers

For hackers, there are online keyloggers and offline keyloggers. They both have the same purpose but there is a slight difference and that is how the hackers access your keystrokes. For an offline keylogger, a text file is created on your storage and normally it is encrypted and gets sent to the hackers’ email/FTP over specified time or intervals, this email can be sent every 24hrs. Online keyloggers are different, these allow hackers to see your keystrokes in real time as you type the hackers can see it in real time. Real-time keyloggers are normally a feature of Trojan horse virus, this virus also comes with other features allowing the hacker to have a backdoor access to your computer, Trojan horses can also include offline keyloggers (Sreenivas, R Sreeram; Anitha, R).

3.4 Problem

Keyloggers steal users’ confidential data for illegal purposes for example credit card, identity thief and login credentials. Keyloggers are also commonly used by hackers and jealous partners who think you are cheating on them only because it is easy to download from the internet for any average person to use, some keyloggers are also known to have keyloggers bind into their executable file or any other file types. This can be done because of buffer overflow. Malicious software called keyloggers has been a major issue for everyone, over the past few years keyloggers have stolen many important data from businesses and average users.
The table below shows some security breaches that was led by keyloggers over the past 10 years.

<table>
<thead>
<tr>
<th>Year</th>
<th>Loss</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>2009</td>
<td>$415,000</td>
<td>Using a key logger, criminals from Ukraine managed to steal a county treasurer’s login credentials. The fraudsters initiated a list of wire transfers each below the $10,000 limit that would have triggered closer inspection by the local authorities</td>
</tr>
<tr>
<td>2006</td>
<td>£580,000</td>
<td>The criminals sent a Trojan via mail to some clients of the Nordea online bank. Clients were deceived to download and execute a “spam fighting” application which happened to install a key logger</td>
</tr>
<tr>
<td>2006</td>
<td>1 million euro</td>
<td>A privacy-breaching malware was used to collect credentials and bank codes of several personal bank accounts in France. The money was transferred to accounts of third parties who were taking the risk of being identified in return for a payment</td>
</tr>
</tbody>
</table>

(Ortolani, S., 2013)

As you can see from the table above, keyloggers are a major concern for businesses and organisations. Therefore, many solutions and techniques were deployed over the years but did not manage to solve the issues with keyloggers. There have been many other cases where keyloggers can be downloaded without the user knowing; there was a report by Websense Security Labs stating attackers are using fake BBC news to lure IE browser to install keyloggers, Trojan horse (which can also contain keyloggers) and worms (Fox News, 2017). Many websites have also adopted two authentication logins to tackle attack vectors like keyloggers, phishing page and sniffing. However, if the second authentication generated number is weak than it is possible for hackers to gain access. Most website use code authentication, most website code length are \( n = (4 \leq n \leq 6) \), 6-digit code length are acceptable but are useless if no restrictions are applied to prevent other attacks like brutal force, however this can take time since there can be countless possibilities, consider the code length is 6, making the largest number in a 6-digit code 999999. The possible combination would be \( 1,388,859,722,465,276,756,946,699,997,550,001 \). Although most companies use a 4-digit code length, making the combination probability \( 416,250,145,812,501 \).
To work out the total amount of combination we use the formula below:

\[ \binom{n}{r} = \frac{n!}{r!(n-r)!} \]

Due to 4-digit passcode being weak, hackers are able to brute force this method if no restriction is in place. At a given time, Facebook were vulnerable to this type of attack, allowing hackers to reset their users’ passwords, granting them full access to their social media account (Graham Cluley. 2017). There is a saying, its best to prevent the cause rather than find a remedy. This is very true, if keyloggers can be prevented from capturing credentials then the second authentication is less likely needed but still is required due to other attack vectors like phishing and sniffing.

Even though we have antiviruses they do fail to detect many keyloggers, to prove a point this report conducted tests against most antiviruses. These keyloggers are public and can easily be downloaded from the internet, these keyloggers are not advanced but is suitable for this test. The paper has used two different keyloggers, both keyloggers have different results and both results were shocking as the highest detection rate is 4 out of 40. From the figures below you can see two sets of results, Figure 5 was only detected by 1 antivirus. Figure 6 was also tested to see how many antiviruses would detect the keylogger, it was tested against 40 antiviruses and only 4 manage to detect this keylogger. Figure 6 test was still running during the screenshot but the detection rate remained the same when the process was finished.

Keyloggers can only be a threat once they are installed onto your devices, hackers use a different type of techniques, causing you to install the keylogger without acknowledging it. For example, malicious emails are being sent out to financial companies containing tools that will install the keylogger onto the victims’ computer. The user is greeted with a fake silver light update claiming it is a security update which tricks the user into installing this security update running the malicious code onto the target device (Targeted Threat Leads to Keylogger via Fake Silverlight Update | Proofpoint. 2017). This method of social engineering is very common for attackers as humans are known to be the weakest link in the security chain. Below are some key points about this method of attack:

- The attack was very narrow in scope - a small number of malicious emails appear to have been sent to users in a single organisation
- The emails included a Microsoft Word attachment that used an embedded object rather than macros to avoid detection; the embedded object was also highly obfuscated
- The payload was an unidentified keylogger hard coded to send logs from infected computers to two Gmail addresses.

Even though this type of attack was small, they have still tried to attack major financial institution; hackers target financial companies for obvious reasons, that area of businesses has the highest guarantee for stealing money.
From these results, you can clearly see that antiviruses as the last line of defence is not reliable and does not guarantee our data safety. There has been a Trojan horse called “Mask” that has not been detected for 7 years, allowing hackers to target an estimated 31 countries, having backdoor access to over hundreds of devices which run Windows, Linux and Mac operating system (Kaspersky. 2014). There is another line of defence that can help, firewalls are important as they prevent unauthorised/undesired incoming and outgoing connections. There are two types of firewalls, stateful and stateless. Firewall is the most important line of defence in your network, to block unwanted incoming or outgoing traffic. There are two different type of firewalls, stateful packet filter and just packet filter. Stateful firewall spends time in examining the layer 4 of the packet and layers below, this allows the firewall to see if the packets match any rules that have been set and if it does it allows it to go through (Stateful Firewalls | How a Stateful Firewall Works | InformIT. 2017). Stateful firewalls are very useful, but they can become expensive to buy including a reasonable amount of knowledge required as those firewalls needs sets of rules. Therefore, this is not the best solution for average users who can lack the knowledge of computers, making this solution becoming useless even if it can be effective. Due to this problem, many people
have tried to find solutions for this issue; the next section talks about the available potential solutions an organisation or average user could use.

3.5 How antivirus detects malicious software

Antiviruses has many ways of detecting malicious software as dangerous; their main detection technique is security hashes. This is done by generating a security hash (for example md5 hash) and checking its database if that hash exists. If the hash exists the antivirus will automatically remove the software from the computer. However, newly written malicious software is not known to antivirus companies, therefore no security hash exists in their database (How Antivirus Software Works. 2017).

Second section technique is heuristics. “Antivirus programs also employ heuristics. Heuristics allow an antivirus program to identify new or modified types of malware, even without virus definition files. For example, if an antivirus program notices that a program running on your system is trying to open every EXE file on your system, infecting it by writing a copy of the original program into it, the antivirus program can detect this program as a new, unknown type of virus. No antivirus program is perfect. Heuristics cannot be too aggressive, or they will flag legitimate software as viruses” (How Antivirus Software Works. 2017). Below is the graph showing detection rate between 19 antiviruses, lower the missing sample percentage represents a better detection rate. The graph below shows missing samples; this is referring to detection by using definition (security hashes e.g. md5 hash). Even though most antivirus shows 0,1 as their result, this does not mean antivirus will detect freshly written keyloggers. This is because antiviruses require viruses’ security hashes to identify the virus and remove the threats from the user device. When a new virus has been developed, the antivirus database does not contain security hashes of the malicious software and antivirus is unable to detect the malicious files as a threat (How Antivirus Software Works. 2017).
3.6 Existing solutions

There are other solutions that have attempted to tackle this situation. Anti-keyloggers have been developed over the years, below are some examples of anti-keyloggers which have all kinds of different features, the table below also states the issues with each anti-keylogger:

<table>
<thead>
<tr>
<th>Name</th>
<th>Price</th>
<th>Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>SpyShelter</td>
<td>35 euros (1 off payment)</td>
<td>“When first installed, it uses an extremely heavy hand – flagging many safe operations as potentially dangerous. You have to dig deep into the program's settings to correct this problem.”</td>
</tr>
<tr>
<td>Zemana</td>
<td>£87 every 3 years (another payment option is available)</td>
<td>“While you can create rules for how the program works, you cannot adjust how aggressively it treats potential threats.”</td>
</tr>
<tr>
<td>Key logger detector</td>
<td>£18 for 1 licence</td>
<td>“It is missing keystroke encryption to hide what you type from potential undiscovered keyloggers on your computer.”</td>
</tr>
<tr>
<td>guardedID</td>
<td>£20 a year</td>
<td>“It has no ability to scan for or remove keyloggers from your computer.”</td>
</tr>
</tbody>
</table>

The report above has named four commercial anti-keyloggers, these keyloggers except keylogger detector encrypt keystrokes to prevent the attacker understanding what was captured. Above there is a column with issues of each keyloggers and as you can see all of them have individual and unique issues. A lot of them have something in common, factor between these anti-keyloggers are heavy resource usage and no control over what should be allowed to capture keystrokes. This paper is different from these solutions because it does not use heavy resources, allowing your computer to run with speed and instead of encrypting keystrokes it unhooks the keyloggers, preventing them from capturing any form of data e.g. keystrokes. Further down this paper, there will be a testing section; testing to show no keystroke can be captured and it does not require a lot of resources e.g. Ram and hard disk space.
4.0 Requirements

To ensure this project runs smoothly, the paper needs to set out goals; these goals will state the functionalities and the requirements of the project during the development stage. Stating the goals and requirements early will allow the project to proceed with a clear structure, helping to reduce time and error during the development process. Below the paper has mentioned some important goals that are required to make the development intentions clear and precise.

**Goal 1 User interface**

The project should have a user interface, making the software more user-friendly, making the software more attractive to potential users. If the user interface is difficult to use, the users would not want to use the software because they want to feel in control of the system instead of the system overwhelming them.

**Goal 2 Speed**

The speed of the program is important, keeping much of the computer resources free as possible, this is mainly important for slow computers and heavy resource users. Users also intend not to use software that requires heavy processing power, and they want speed, this also has played a role in the graphical user interface design. Users wants to be able to navigate around the program without any issues. Therefore, the design needs to be simple and every operation can be done with a click of a button.

**Goal 3 Reliability**

The program needs to be reliable, allowing the users to rely on preventing the users keystrokes from being captured. This is the most important goal as it is the purpose of this paper. If the software fails to deliver this goal, it will make the software purposeless and therefore the dissertation will not be solving a problem that was stated earlier.

**Goal 4 Validation**

This goal is important, although much validation is not required the project still needs some validation in place. The project needs to validate any users input that might be required such as selecting files or ensuring files exist. Validation is important because it reduces the chances of errors or any crashing occurring, making the program more reliable and user-friendly.
4.1 Prioritising functional and non-functional features

When developing software, you need to ensure the main functionality of the software is completed before considering adding extra features. Not taking into consideration of essentials functions would lead to scope creep. Scope creep “refers to a project that has seen its original goals expand while it is in progress. As the term suggests, scope creep is a subtle process that starts with small adjustments and ends up resulting in projects that take far longer to complete or even fail before they are finished. Even if the project is completed, scope creep can result in final deliverables that look nothing like what was originally envisioned.” (Techopedia. 2017). The MoSCoW method helps to priorities the requirements that the paper stated earlier. MoSCoW means:

M - This must be included to meet software needs.
S - This should be included if possible but is not required.
C – This could have the requirements and does not affect the projects.
W – I would like this requirement but cannot be delivered in the time given.

- “The O’s in Moscow is added for acronym purpose”.

<table>
<thead>
<tr>
<th>ID</th>
<th>Requirements</th>
<th>Priority MoSCoW</th>
<th>Description</th>
<th>Benefits</th>
<th>Potential problems/solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Preventing capture of keystrokes</td>
<td>MUST</td>
<td>This functionality prevents the hackers from capturing key strokes of the user</td>
<td>Users can type confidential information or credentials without having the concern of attackers stealing their data</td>
<td>If this functionality does not work hackers would be able to capture key strokes of the users, containing information that can be confidential data or login credentials</td>
</tr>
<tr>
<td>2</td>
<td>Multi-Tasking</td>
<td>Would</td>
<td>This functionality would allow the user to multi-task. This is something that could be beneficial, allowing them to</td>
<td>This is useful for confidential research</td>
<td>User can only use one application, preventing them to multi-task which can make the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>use the program with more ease</td>
<td>experience unpleasant.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>-------------------------------</td>
<td>------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The solutions can be implemented in the near future</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>User should be allowed to use the software at their convenient</td>
<td>MUST User should be allowed to decide when they want the software to enter safemode</td>
<td>Allowing user to have full control, making it more user friendly</td>
<td>Users may forget to enter the secure mode and keyloggers can be captured.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The solution is to implement multi-tasking as this is the cause of allowing the user to decide when it should enter secure mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Validation across the entire project</td>
<td>MUST Validation should be put in place where the user input is required</td>
<td>If there is any missing files the user will be aware, the user will only be allowed to run the program when all inputs meet the requirements</td>
<td>User might try to run application that might not be supported.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>All software could be implemented in the future if not supported.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The table above states what it is prioritising what is required to achieve within the time frame given and what would be implemented if more time was given for this project. The table states the requirement, priority, description of the requirements. It also contains important and potentially useful information like benefits, allowing the report to state why
it is beneficial to have these requirements in the software. The table also has a potential problem and solution column, stating what could possibly go wrong for the requirement and what solutions can be put in place to prevent the issues from occurring. It is important to acknowledge all potential issues that can occur from the desired requirements. It is important because we can have a clear understanding of what is required for developing the solutions and the solutions for potential errors that can be caused. This can allow some consideration of potential solution to an error that can be caused; this can be validation before the program has starts.

5.0 Designing process

The designing process is the first part of developing the solution to the problem, in this section, the report will be showing UML use case, Storyboards, UML class diagram and many more diagrams to support the development and understanding of the project. UML diagrams are an important aspect of development; it uses graphic notations showing a visual model of the software that is being developed.

5.1 Scope

“Project scope is the part of project planning that involves determining and documenting a list of specific project goals, deliverables, tasks, costs and deadlines”. Allowing the project to have a clear deadline and objectives to complete, below the report will set the project goals and scope.

- **System Boundaries:** This will define the area of the organisation under investigation and may also specify the limit of any new system implemented because of this project.
  - The analyst will be the application that was developed to tackle the paper problem that was stated earlier in the report.
  - The System includes security feature to prevent keylogging securing all the key strokes that could leak information like credentials and stopping techniques like hooking.

- **Constraints:** Factors, including budget, timescale and technology, which may restrict the study, or the solution, in some way. These constraints will be considered in more detail later in this chapter.
  - Technology: source available, no technology constraints.
  - Environment: this only supports for Windows operating system as the research title works scope is within the Windows OS.
  - Time: Limited time for development and research that could cause issues for this paper to find the best solution bug-free software.
  - Budget: no budget as this is a dissertation project.
Objectives: An unambiguous statement of the expectations of those in the client’s organisation who have initiated the project. These may be broken down by function or department. Well-defined objectives are clear and measurable.

- help this paper to show the functionality of this software and its intended purpose.
- Hand dissertation in on deadline day or before, completing software within a few months.

Permission: This will indicate who in the client’s organisation is responsible for the supervision of the project and if permission needs to be granted - for example to extend the scope of the analysis - who has the authority to do so. Points of contact and the appropriate reporting structure may also be defined.

- No permission is required as this project is being developed without needing other assistants or personal data which may concern legal and ethical issues.

End-Product: A description of the deliverable or end products of the investigation. This will usually take the form of a written report and a supporting presentation to managers of the client organisation.

- Dissertation solves a real-world problem by developing software to handle malicious software, allowing average users and companies to have extra security on their devices and preventing hackers from stealing keystrokes.

(SCOPE Reference: Donald Yeates, 2004)
5.2 UML use case

“UML Use Case Diagrams. Use case diagrams are usually referred to as behaviour diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors)” (Kirill Fakhroutdinov. 2017).

(Figure 8: UML diagram of the developed program)

1. User enter desired application
2. Programs enter “SafeMode”
3. User keystrokes will not be captured.

The diagram above shows the user interaction with the program, showing how the process has occurred and what stages and functionality the users are at.
**Functionality Description:** this function allows the user to select the desired file or application to run. This will update the user interface showing what file has been selected by the user, enabling the run button.

**Goal:** Store file to memory  
**Scope:** “SafeMode”  
**Level:**  
**Primary actor:** user/organisation  
**Stakeholder:**  
  - **Programmer:** Wishes there is no bugs/software error in the system.  
  - **Client:** wants to load file without any error  
**Preconditions:** Client must be downloaded  
**Trigger Event:** file is selected  
**Success Guarantees:** File successfully selected  
**Main Success Scenario:**

1. User runs application  
2. Selects file that he desired to run  
3. Enters SafeMode  

**Extensions:**

1a. Application not downloaded  
1a1. User downloads application  
1b. Application is missing files  
1b1. User re-downloads the application  
2a. application corrupt
2a1. Reinstall application
2b. Cannot load file due to privilege issues.
2b1. Runs SafeMode as administrator
2b2. Contact administrator if user does not have admin rights

(Figure 10: UML Use Case Diagram)

**Functionality Description:** This functionality starts when the user clicks the run button, allowing the user to enter a secure desktop mode. This mode allows the user to write confidential emails, documentation and enter login credentials. Without the hacker stealing any keystrokes via hooking or API based.

**Goal:** Enter SafeMode  
**Scope:** “SafeMode”  
**Level:**  
**Primary actor:** user/organisation  
**Stakeholder:**  
- **Programmer:** Wishes there is no bugs/software error in the system.  
- **Client:** wants to enter SafeMode  

**Preconditions:** Client must be downloaded and file selected  
**Trigger Event:** file is selected, and user enters “SafeMode”  
**Success Guarantees:** user in “SafeMode”  
**Main Success Scenario:**

1. User clicks “Run”  
2. User enters “SafeMode”

**Extensions:**
1a. User run button is not clickable
1a1. User selects file
2a. SafeMode does not run
2a1. File needs to be word or an application
2a2. Contact programmer and alert him of issue

Above are UML use case diagrams to show how the user interacts with the software, these UML diagrams also come with happy days’ scenarios.
5.3 Storyboard

Below are the sprints of the storyboard, showing what features are required for the software, even though it only has two storyboards these are all the features we require to develop the software. “SafeMode” is a secure desktop created by the program where the user can enter credentials or write confidential information (on email) without the keylogging capturing keystrokes.

1. **“Entering SafeMode”**
   - User can enter “SafeMode”

2. **“Prevent keylogging”**
   - Programs are not allowed to capture keystrokes

- +Can user enter this “SafeMode”?
- +User in “safeMode”
  +Hooked keyloggers cannot log
5.4 Class diagram

Class diagram is an important step in software designing; its purpose is to show the structure of the system. This is done by describing objects and classes the system requires (Fowler, 2013). The class diagram below will show the global variables every class has and the relationship between each class. The class will also show the methods within each class, allowing us to understand how the class methods operate. Below are three class diagrams of the three programs that are going to be developed.

5.4.1 SAFEMODE-GUI

(Figure 11: SafeMode_GUI class diagram)
5.4.2 SAFEMODE

(Figure 12: SafeMode class diagram)
5.4.3 Cleaning

As you can see, there are 3 different class diagrams; this is because the application requires 2 external executable files including a .dll dependency, this .dll file was created from a different software engineer called George Mamaladze, this .dll file helps me to perform actions which are required for preventing keystrokes. Each of the files is required as they serve a different purpose, the SafeMode_GUI class is the user interface that the user is greeted with at the start. Allowing the user to select the file he chooses to run in this secure mode. SafeMode does most of the heavy work, creating new taskbar, preventing key hooking, etc. showing you relations between all three-executable file.

(Figure 13: cleaning class diagram)
6.0 GUI

In this section, the paper will be showing the development process of the project; below you can see the GUI. The user interface was designed to be simple and easy to use for users; this was done to allow the user to have a joyful experience with the software. Users will not attempt to use software that is heavily populated with tabs, text fields and buttons on one interface.

(Figure 14: user interface when program first starts)

Image above is the first user interface the user will be greeted with, in this section the user will be allowed to select the file or application the users wants to launch. This is simply done by clicking “Select File”, a dialog will appear asking you to navigate and select the file you wish to load.

(Figure 15: GUI once chrome was selected)
This is the second stage; the user would have selected the application or documentation the user wants. The run button will become enabled allowing the user to start in “SafeMode”. If the user decides the user wants to exit the application the user can simply click on close on the X on the top right to close the application. As you can see from the figure above you can see the icon of the application, allowing the user to be aware of what file has been selected, along with a label saying “chrome.exe”.

(Figure 16: user in secure mode and no keyloggers can capture keystrokes)

Once the user has clicked the run button the application should take over the screen, running the file the user has selected, it has also prevented keyloggers from capturing keystrokes by unhooking them. Since most or nearly all modern keyloggers are API based techniques, this software becomes very effective by preventing your keystrokes from being captured. The application prevents the user from running software; it has also disabled the taskbar and other features. The user can access task manager but use the host key control + shift + escape. The user can also forcefully close the application by using the hot key control + alt + K; this is useful as it can allow the application to close without having to terminate any process since the application does it for you. This hotkey is useful because other software can crash or freeze, which can cause problems for the user. Allowing the user to force quit the application which having to worry about being stuck in this secure mode. There is another hotkey allowing the user to see what process is running inside this secure mode, so the user can be ensured no malicious software is capturing their keystrokes. The hotkey to access this is control + alt + V. Hotkeys were the best idea for the user to load upside function of the application, making it cleaner and easier to use. The only problem this causes is that user might be unaware of the hotkeys, and nothing is displayed to the user about hotkeys.
6.1 Program usage

Below are two figures of the computer resourcing being in used, this allows the paper to show the effect the software has on the computer. You can see fewer resources are required when the application is running. Due to the application taking over the user screen it requires fewer resources to run because icons, taskbar and other features have been replaced with this new security overlay. Unfortunately, due to computers having an extensive amount of memory and processing power this does not provide a desirable benefit but can be useful for future work which will be mentioned later in the report. It is very important to see how the application impacts the computer resources and its behaviour, if any negativity is being shown then it is a sign something is going wrong and more work is required.

Before

(Figure 17: computer resource usage before the project is running)

After

(Figure 18: computer resource usage after the project is running)
# 6.2 Testing

<table>
<thead>
<tr>
<th>Test ID</th>
<th>Test description</th>
<th>User input (if required)</th>
<th>Expected result</th>
<th>Result</th>
<th>Pass/Fail</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The application can run successfully on windows OS without any errors occurring.</td>
<td>Not required</td>
<td>Application to start</td>
<td>Application has successfully started without producing errors</td>
<td>Pass</td>
</tr>
<tr>
<td>2</td>
<td>Application can load text file</td>
<td>Location of text file</td>
<td>Default text software should run in “SafeMode”</td>
<td>NotePad starts in “SafeMode”</td>
<td>Pass</td>
</tr>
<tr>
<td>3</td>
<td>Any web browser starts in secure mode</td>
<td>Location of web browser .exe file</td>
<td>Browser to run</td>
<td>Browser started in secure mode</td>
<td>Pass</td>
</tr>
<tr>
<td>4</td>
<td>Prevent keystrokes from being captured</td>
<td>Press keys from keyboard in secure mode</td>
<td>No capture of keystrokes</td>
<td>Keyloggers fail to capture keystrokes</td>
<td>Pass</td>
</tr>
<tr>
<td>5</td>
<td>Able to access web browser and text documents</td>
<td>Not required</td>
<td>Web browser and text documents to appear in secure mode</td>
<td>Unable to select both applications</td>
<td>Fail</td>
</tr>
<tr>
<td>6</td>
<td>Application throws error if required files are missing</td>
<td>Not required</td>
<td>Shows error</td>
<td>Showed errors, naming the missing files</td>
<td>Pass</td>
</tr>
<tr>
<td>7</td>
<td>Application closes when user exist the desired app</td>
<td>User closes web browser or document</td>
<td>Program closes</td>
<td>Program closed successfully</td>
<td>Pass</td>
</tr>
<tr>
<td>8</td>
<td>Prevent the user from running the secure mode if a file has not been selected</td>
<td>Not required</td>
<td>If file not selected the run button will not appear</td>
<td>Program prevents user from running secure mode without file selection</td>
<td>Pass</td>
</tr>
</tbody>
</table>
9 User can run task manager in “SafeMode” User uses hotkeys… Task manager should appear User was successful to getting the task manager running Pass

10 The software prevents key stroke capture from multiple keyloggers User needs to press random keys on keyboard Keystrokes are not captured. Keyloggers failed to capture any keystrokes Pass

11 Software runs on more than 1 version of windows operating system Not required. Runs on multiple windows operating system Software has successfully run on windows 7 and windows 10 OS Pass

From the table above, we can see the application has passed most of its test except multi-tasking since all required features were successfully achieved means the test was successful even though it contains one fail. Earlier, we prioritised the requirements that were needed to make this development clear and precise; multi-tasking was mentioned as a “would”, a feature that would be desired if more time was given for this application.

Multiple keylogger test

Testing against multiple keyloggers is a very important process, allowing us to see if the project fully fulfils its purpose. For this test, the report will be using a Trojan horse (that contain a key logger) call njrat and different keylogging software called spyrix free key logger. This test will check both malicious software to see what they capture during the testing before the program is in use and after the program is running in secure mode. The image below shows the two-malicious software running inside a virtual machine:
Each malicious software had to be tested separately due to them causing each other capturing issues. For this test, both malicious software should capture the username and password being typed into the website [www.facebook.com](http://www.facebook.com). The credentials will not be sent to the website as the login button will not be triggered, showing the keyloggers are capturing keystrokes and not by sniffing (man in the middle attack). The web browser that will be used for this testing is google chrome, and the test will be using Windows 7 Home premium x64, the virtual machine was also allocated 4GB ram (host computer has 8GB) and 200GB of storage (host has 1TB).

For this report, it was decided to start off with spyrix, the figure below shows the malicious software successfully capturing the login credentials (username: username1, Password: password1).

![Figure 20: spyrix successfully capture key strokes](image)

After the key logger has successfully captured the keystrokes as expected it was time to try the project that was developed. Google chrome was running inside the software that was developed to answer this papers’ problem and we did similar scenario. The only difference was the login credentials were username2 and password2. As expected the keylogger has failed to detect the keystrokes and the user credentials were saved from being captured by malicious software, as show in the figures below.
(Figure 21: enter new credentials in secure mode)

(Figure 22: spyrix failed to capture new credentials in secure mode)
As you can see from the figure above you cannot see the capture of the new credentials that were used, this has proven the program is working to this paper's expectations but one malicious software was not enough. The program needed to be tested against a different malicious software too, therefore a Trojan horse was installed on the virtual machine and given the same scenario with same credentials. As expected the application was successful in stopping malicious software from capturing keystrokes. As it can be seen from the two figures below.

(Figure 23: njrat successfully captures key strokes)

(Figure 24: NJRat fails to capture keystrokes in secure mode)
From these tests, we can confirm the project development was successful based on the black box testing and practical testing in a virtual machine. These positive results show project was successful in reaching its goals and solving the issue that we discussed earlier in this paper. The two malicious software that were being used focuses on hooking and API techniques to capture keystrokes, the testing cannot show if the program works against kernel keyloggers since they are not known as modern keyloggers and is difficult to find a kernel based keylogger. Overall, the testing was successful, and the report can move on to reviewing its finding and the developed system.

The report will go through the pros and cons of the application that was developed to tackle the problem the report has stated earlier. Discussing the benefits of the feature that has been implemented and the negativity of features that could have been implemented. This section is mainly about the cons and area of improvement that can be made with extra time and more research.

This project lacks multi-tasking which is highly important for users to be able to do multiple actions at one given time. This could have been improved and implemented with extra given time but will not be implemented for this report since it fulfils its requirements. Multi-tasking plays an important role in technology, however, listening to music and writing confidential emails/entering login credentials in secure mode is still possible since the application does not close any program you have open in the background.

The user interface of the software does not appeal to the human eye if the application could have a custom user interface; it would become more attractive to use for users. This could be done during spare time as this is not a required feature of playing a major role in the current problem that is trying to be solved.

The application comes with one major benefit, preventing keystrokes from being captured. This is highly important since it is the main purpose of this paper. Users will be allowed to write confidential emails or enter login credentials without the hacker capturing these important data. This was tested and proven earlier in the report when two types of test were conducted, black box test and a practical test within a virtual machine. All test results came back positive (except multi-tasking, Table 5) and showing the main requirements of the application was a success.

Another major benefit this software gives is freeing up resources, although computers come with expensive amount of memory and storage the application gives that benefit to computer users. Earlier in the report, two graphs were shown, showing the computer usage before and after the software was running. From the graphs, you can see the program did not use much of the resource, but it has also reduced the computer usage. Speed was also considered as a goal for this project and by freeing up the computer resource and not requiring a lot of resources.
7.0 Review and conclusion

The paper has come to the final section, the reviewing and evaluation of the dissertation. Earlier the report has mentioned the real-world issue that causes major security concerns worldwide; the paper has also declared the aims to solve this problem. Reviewing the entire dissertation is important as all points from positive to negative need to be mentioned. Identify areas of improvements allows future works and potentially this project to improve and progress. This section will start by mentioning what the requirements were; the paper will also review the literature and mention if it had any benefits on the outcome of this paper, the type of methodology that was used for researching and development. Finally, this section will also discuss future works.

7.1 Evaluation of the dissertation

The intentions of this dissertation are to solve a problem; this paper was decided to be based on keyloggers. In able to fully comprehend the problem the report needed to see the literature, this gave a better understanding of how keylogger techniques are used to capture keystrokes. This was very beneficial and had participated in the outcome of the software. The start of the dissertation the software aim was mentioned, the only main objective was to prevent keyloggers from capturing keystrokes, even though one objective may seem simple but it comes with sets of challenges and complexity. With the solution, it has developed people around the world would be able to benefit from this software, ensuring them their keystrokes are not being captured and that they may proceed with confidential data/information.

The methodology that was used for this dissertation was secondary research and scrum. Secondary research was the best option at the time of discussion making, due to a variable that played a factor such as time, problem and area of study. For development scrum was used. Scrum is an agile method of developing software, allowing to specify the requirements at an early stage, allowing us to know the project development duration to completion with the required feature. This allowed the project to adapt to any issues that occurred during the development process.

Towards the final part of this paper was testing of the application to determine the entire dissertation output was a success or not. After doing the black box testing, it was noticeable the main goal of this paper was achieved, and the outcome of this paper is a success. The program had a few issues that can be considered as desired features rather than necessary features. The few issues which includes multi-tasking has been considered as a side feature; if given more time this can be implemented. Other tests have been managed to successfully pass, these tests have been stated as required features, giving the application that was developed a purpose; which was stated earlier in the report.
7.2 future work

In this field, there is always room for improvements and possibly better solutions as technology grows and improve at a rapid rate. For this project, there are many improvements that can be made to increase its efficiency, usability and of course multi-tasking. One of the improvements that should be mainly considered is multi-tasking as the user is not able to run more than one application in this application, changing the integrity of the system to make it more efficient is another possibility. Since multi-tasking is currently not an option in this software, implementing this feature will increase the chances of the usability as it also becomes more user-friendly.

Another side feature that could be implemented in the future is keylogger detection, most anti-keylogger have this feature, alerting the users which application is capturing keystrokes. Allowing the user or organisation to believe they have full control makes them feel satisfied as users want to control. By working on multi-tasking, allowing the user to start new processes and close processes within this secure mode will make the user experience more enjoyable. Due to the program freeing up more resources of the computer, the program can easily evolve without having any worries about computer usage like most keyloggers do. In the future, the program can allow the taskbar to come back as the Windows taskbar is a very useful functionality in the Windows operating system, as it allows the user to navigate to the user desired location. Another functionality that can be added is preventing screen capture; this is another implementation that could possibly be useful to prevent hackers from watching your screen. Some malicious software like some keyloggers capture screenshots every minute, this does not provide any dangerous information unless the malicious software is installed on a business computer which can cause a problem like customer data being leaked.

Above, the paper has talked about how adding multitasking can be very positive for users but it can come with its own security issues. Adding multitasking could allow malicious software to capture keystrokes if the user decides to run the malicious software within the secure mode. However, malicious software is bind with other software, and when executed it requires its own process. The application can still prevent malicious application (keyloggers) from capturing keystrokes but more testing will be required in the future answer questions if multi-tasking capability is implemented.

7.3 Summary

This report has gone through the literature review, showing other researchers work relating to the field the paper is based on. Setting the background of the topic area before continuing to solve the problem. After the literature review, the paper has stated the type of methodologies were going to be used for research and software development. This section has talked about agile methods like scrum and waterfall; these are ways to develop software; scrum was decided and explained why. The research was followed up after methodology, containing useful information that is useful for this paper, making the reader more aware and the purpose of this topic. The paper has gone through past security
breaches that occurred due to keyloggers and why they are a problem, including the intentions of their usage. This can be from identity theft to stealing credentials or credit card details. The research section also contained information about antivirus and how they fail to protect our computers from malicious software, causing major security concerns for average users and organisations. Other solutions that have tried to tackle this problem was also mentioned, stating why they also have issues as they do not prevent keyloggers from capturing your keystrokes, they only encrypt.

The paper then went on to designing the solution that could solve the problem, in the designing process the paper has displayed some UML diagrams to show the structure of the software. Showing how the applications work, the type of classes being used and their relationships between each other. On the development process, the paper used visual studio to develop the program in C#. C# was used due to its capabilities that other languages like Java do not provide, C could not have been used due to the time that was given, learning a new language takes time and can possibly fail to reach deadline day.

Testing was also carried out, checking the program capabilities and seeing if the program reaches the goals and requirements that were set earlier in the report (4.0 requirements). The program must pass the black box testing; this is to test the program behaviour with user inputs and without user inputs, it has managed to pass all the required features successful. After the black box testing, the project was required to be tested against multiple keyloggers, for this two malicious software were downloaded, NJRat and spyrix. The software has successfully managed to prevent the user keystrokes from being captured from two malicious software. The results were discussed in detail under the testing section. From the black box test and from figure 21 and 16 you can see the application can successfully run on multiple versions of Windows operating system. However, the application is running from C#.NET; this means any computer that wants to run this application needs net framework 2.0 minimum for this application to run successfully otherwise the application will fail to start, throwing an error at the user.
7.4 Conclusion

Finally, the project has come to an end, the purpose was to improve security by preventing captures of keystrokes, and this was achieved. Doing this report allowed a conductive research, making the topic clearer as this report increases knowledge of this area of the field. The paper has shown great knowledge from the literature review allowing the readers to have a better understanding topic; this was done by reviewing journals that has been created by great minds around the world. Developing the software required a lot of planning in able to complete it within the time given. Unfortunately, this development has taken up more time than it was expected which lead to concerns for my academic paper to be completed on time. Although sticking to the plan has managed to pay off and the development process was completed leaving enough time for the academic paper due to reasonable time management. It was understood that language like Java did not meet the requirement of this topic due to its lack of hardware access, which leads to C# being the perfect language to meet the requirements to a satisfying level.

From the research and testing which was done towards the end of the report (3.0 research, 6.0 GUI), the paper shows that it is possible to prevent keystrokes from being captured which fixes the problem this paper has led to solving. From the previous literature review, the paper managed to gather enough information on techniques which malicious software use to capture keystrokes e.g. hooking. It was important to ensure the literature review was done to increase the understanding of the problem and the current solution others have deployed. Due to this, the project could develop software to tackle all modern keyloggers using techniques like hooking and API-based. If more time is allocated towards this project multi-tasking can also be implemented making the program suitable for release, this software was also successful due to the .dll files created by another developer, allowing the software to monitor all mouse and keyboard behaviour. Although there are other solutions available to the user; those solutions mentioned earlier have issues of their own which can still cause some potential security concerns. All issues with other solutions have been mentioned earlier in the report, whereas this report addresses all security issues by preventing capture of keystrokes.
Appendix

Keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hacker</td>
<td>“A person who uses computers to gain unauthorised access to data”</td>
</tr>
<tr>
<td>Antivirus</td>
<td>Software “designed to detect and destroy computer viruses.”</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical user interface</td>
</tr>
<tr>
<td>OS</td>
<td>Operating System</td>
</tr>
<tr>
<td>AV</td>
<td>Antivirus</td>
</tr>
<tr>
<td>VM</td>
<td>Virtual Machine</td>
</tr>
<tr>
<td>JVM</td>
<td>Java Virtual Machine</td>
</tr>
<tr>
<td>GB</td>
<td>Gigabyte</td>
</tr>
<tr>
<td>TB</td>
<td>Terabyte</td>
</tr>
<tr>
<td>Ram</td>
<td>Random memory access</td>
</tr>
</tbody>
</table>

Ethic number: 2016D0461

Ethic form
### PART ONE

<table>
<thead>
<tr>
<th><strong>Name of applicant:</strong></th>
<th>Azeem Yousaf</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Supervisor (if student project):</strong></td>
<td>Dr Tom Crick</td>
</tr>
<tr>
<td><strong>School / Unit:</strong></td>
<td>School of management</td>
</tr>
<tr>
<td><strong>Student number (if applicable):</strong></td>
<td>ST20063690</td>
</tr>
<tr>
<td><strong>Programme enrolled on (if applicable):</strong></td>
<td>BSC (Hons) Software Engineering</td>
</tr>
<tr>
<td><strong>Project Title:</strong></td>
<td>Preventing capture of keystrokes in windows OS.</td>
</tr>
<tr>
<td><strong>Expected start date of data collection:</strong></td>
<td>01/01/2017</td>
</tr>
<tr>
<td><strong>Approximate duration of data collection:</strong></td>
<td>2 months</td>
</tr>
<tr>
<td><strong>Funding Body (if applicable):</strong></td>
<td>n/a</td>
</tr>
<tr>
<td><strong>Other researcher(s) working on the project:</strong></td>
<td>No one.</td>
</tr>
<tr>
<td><strong>Will the study involve NHS patients or staff?</strong></td>
<td>No</td>
</tr>
<tr>
<td><strong>Will the study involve human samples and/or human cell lines?</strong></td>
<td>No</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Does your project fall entirely within one of the following categories:</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Paper based, involving only documents in the public domain</strong></td>
</tr>
<tr>
<td><strong>Laboratory based, not involving human participants or human samples</strong></td>
</tr>
<tr>
<td>Practice based not involving human participants (e.g. curatorial, practice audit)</td>
</tr>
<tr>
<td>Compulsory projects in professional practice (e.g. Initial Teacher Education)</td>
</tr>
<tr>
<td>A project for which external approval has been obtained (e.g., NHS)</td>
</tr>
</tbody>
</table>

If you have answered YES to any of these questions, expand on your answer in the non-technical summary. No further information regarding your project is required.
If you have answered NO to all of these questions, you must complete Part 2 of this form.

In no more than 150 words, give a non-technical summary of the project:
My project is about preventing key-loggers capturing key strokes from our keyboard, what type of measures are already available and what can be done to improve the current measure that is available.

**DECLARATION:**
I confirm that this project conforms with the Cardiff Met Research Governance Framework.

I confirm that I will abide by the Cardiff Met requirements regarding confidentiality and anonymity when conducting this project.

**STUDENTS:** I confirm that I will not disclose any information about this project without the prior approval of my supervisor.

Signature of the applicant: Azeem Yousaf

| Date: |

FOR STUDENT PROJECTS ONLY

Name of supervisor: [Name]

Date: [Date]

Signature of supervisor: [Signature]

**Research Ethics Committee use only**

Decision reached: [Project approved] [Project approved in principle] [Decision deferred] [Project not approved] [Project rejected]

Project reference number: [Click here to enter text]

Name: [Click here to enter text]

Date: [Click here to enter a date]

Signature: [Click here to enter text]

Details of any conditions upon which approval is dependant: [Click here to enter text]
### PART TWO

#### A RESEARCH DESIGN

<table>
<thead>
<tr>
<th>A1 Will you be using an approved protocol in your project?</th>
<th>Choose an item.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A2 If yes, please state the name and code of the approved protocol to be used</td>
<td>Click here to enter text.</td>
</tr>
</tbody>
</table>
| A3 Describe the research design to be used in your project | In this section, include details (as appropriate) of:  
- research method(s);  
- sample and sampling;  
- recruitment of participants;  
- analytical techniques  
If your project does involve the use of an approved protocol, much less detail will be required but you should indicate which areas of the project are covered by the protocol. |
| A4 Will the project involve deceptive or covert research? | Choose an item. |
| A5 If yes, give a rationale for the use of deceptive or covert research | Click here to enter text. |
| A6 Will the project have security sensitive implications? | Choose an item. |
| A7 If yes, please explain what they are and the measures that are proposed to address them | Click here to enter text. |

#### B PREVIOUS EXPERIENCE

| B1 What previous experience of research involving human participants relevant to this project do you have? | Click here to enter text. |
| B2 Student project only | What previous experience of research involving human participants relevant to this project does your supervisor have? |

#### C POTENTIAL RISKS

| C1 What potential risks do you foresee? | Include details of risks to the participants, the researcher and the project as a whole. |
| C2 How will you deal with the potential risks? | Click here to enter text. |
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Drawing;
using System.IO;
using System.Runtime.InteropServices;

using System.Text;
using System.Threading;
using System.Windows.Forms;

namespace SafeMode_GUI
{
    public partial class Main : Form
    {
        BackgroundWorker background = new BackgroundWorker();
        public Main()
        {
            InitializeComponent();

            this.AllowDrop = true;
            this.DragEnter += delegate(object sender, DragEventArgs e)
            {
                if (e.Data.GetDataPresent(DataFormats.FileDrop))
                {
                    e.Effect = DragDropEffects.Copy;

                    LoadFile(((string[])e.Data.GetData(DataFormats.FileDrop))[0], true);
                }
            };
            this.DragDrop += delegate(object sender, DragEventArgs e)
            {
                LoadFile(((string[])e.Data.GetData(DataFormats.FileDrop))[0]);
            };

            background.DoWork += delegate
            {
                IntPtr hProc = IntPtr.Zero;

                WinAPI.STARTUPINFO si = new WinAPI.STARTUPINFO();
            }
        }
    }
}
WinAPI.PROCESS_INFORMATION pi = new WinAPI.PROCESS_INFORMATION();
WinAPI.CreateProcess(null, String.Format("\"{0}\" \"{1}\"", Program.location + \\
"\SafeMode.exe", fileloc), IntPtr.Zero, IntPtr.Zero, false,
WinAPI.CREATE_NO_WINDOW, IntPtr.Zero, null, ref si, out pi);
  hProc = pi.hProcess;
  try
  {
    uint code;
    while (!this.IsDisposed)
    {
      Thread.Sleep(500);
      if (!WinAPI.GetExitCodeProcess(hProc, out code) || code != 259) { break; }
    }
  }
  catch { }
};
background.RunWorkerCompleted += delegate
{
  btnRun.Enabled = true;
};

fileName.SizeChanged += delegate
{
  if (fileName.Width < 260) this.Width = 428;
  else
    this.Width = fileName.Width + 168;
};

Image GetIcon(string file)
{
  Icon ficon;
  WinAPI.SHFILEINFO shinfo = new WinAPI.SHFILEINFO();
  IntPtr ptr = WinAPI.SHGetFileInfo(file, WinAPI.FILE_ATTRIBUTE_NORMAL,
  ref shinfo, (uint)Marshal.SizeOf(shinfo), WinAPI.SHGFI_SYSICONINDEX);
  if (ptr == IntPtr.Zero) ficon = Icon.ExtractAssociatedIcon(file);
  else
  {
    int iconIndex = shinfo.iIcon;
    Guid iImageListGuid = new Guid("46EB5926-582E-4017-9FDF-E8998DAA0950");
    WinAPI.IImageList iml;
    int hres = WinAPI.SHGetImageList(0x04, ref iImageListGuid, out iml);
    IntPtr hIcon = IntPtr.Zero;
    hres = iml.GetIcon(iconIndex, 1, ref hIcon);
    ficon = System.Drawing.Icon.FromHandle(hIcon);
  }
}
Image img;
Icon temp = new Icon(ficon, 128, 128);
img = temp.ToBitmap();
temp.Dispose();
ficon.Dispose();
return img;
}

string fileloc = "";
void LoadFile(string file, bool preview = false)
{
    icon.Image = GetIcon(file);

    fileName.Text = Path.GetFileName(file);
    FileInfo info = new FileInfo(file);

    if (!preview)
    {

        fileloc = file;
        btnRun.Enabled = true;
    }
}

private void btnRun_Click(object sender, EventArgs e)
{
    if (!File.Exists(fileloc))
    {
        MessageBox.Show("The selected file no longer exists", "Error",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }

    if (!Program.CheckValidity()) return;

    btnRun.Enabled = false;
    background.RunWorkerAsync();
}

private void btnExit_Click(object sender, EventArgs e)
{
    this.Close();
}

private void menuFileOpen_Click(object sender, EventArgs e)
{
}
private void menuFile_Click(object sender, EventArgs e)
{
}

private void selectFileToolStripMenuItem_Click(object sender, EventArgs e)
{
using (var ofd = new OpenFileDialog())
{
if (ofd.ShowDialog() != DialogResult.OK)
    return;

    LoadFile(ofd.FileName);
}
}

private void closeToolStripMenuItem_Click(object sender, EventArgs e)
{
    this.Close();
}

}

namespace SafeMode_GUI
{
    public class WinAPI
    {
        [StructLayout(LayoutKind.Sequential, CharSet = CharSet.Auto)]
        public struct SHFILEINFO
        {
            public IntPtr hIcon;
            public int iIcon;
            public uint dwAttributes;
            [MarshalAs(UnmanagedType.ByValTStr, SizeConst = 260)]
            public string szDisplayName;
            [MarshalAs(UnmanagedType.ByValTStr, SizeConst = 80)]
            public string szTypeName;
        }
    }
}

using System;
using System.Collections.Generic;
using System.Runtime.InteropServices;
using System.Text;
using System.Collections.Generic;
using System.Runtime.InteropServices;
using System.Text;

namespace SafeMode_GUI
{
    public class WinAPI
    {
        [StructLayout(LayoutKind.Sequential, CharSet = CharSet.Auto)]
        public struct SHFILEINFO
        {
            public IntPtr hIcon;
            public int iIcon;
            public uint dwAttributes;
            [MarshalAs(UnmanagedType.ByValTStr, SizeConst = 260)]
            public string szDisplayName;
            [MarshalAs(UnmanagedType.ByValTStr, SizeConst = 80)]
            public string szTypeName;
        }
    }
}
[DllImport("user32.dll", SetLastError = true)]
public static extern bool DestroyIcon(IntPtr hIcon);

[DllImport("shell32")]
public static extern IntPtr SHGetFileInfo(
    string pszPath,
    uint dwFileAttributes,
    ref SHFILEINFO psfi,
    uint cbFileInfo,
    uint uFlags);

public const int FILE_ATTRIBUTE_NORMAL = 0x80;

public const int
SHGFI_SYSICONINDEX = 0x4000,
SHGFI_JUMBO = 0x4;

public const int CREATE_NO_WINDOW = 0x08000000;

[DllImport("shell32.dll", EntryPoint = "+727")]
public extern static int SHGetImageList(int iImageList, ref Guid riid, out IImageList ppv);

[StructLayout(LayoutKind.Sequential)]
public struct RECT
{
    public int left, top, right, bottom;
}

[StructLayout(LayoutKind.Sequential)]
public struct POINT
{
    int x;
    int y;
}

[StructLayout(LayoutKind.Sequential)]
public struct IMAGELISTDRAWPARAMS
{
    public int cbSize;
    public IntPtr himl;
    public int i;
    public IntPtr hdcDst;
    public int x;
    public int y;
    public int cx;
    public int cy;
    public int xBitmap;    // x offest from the upperleft of bitmap
    public int yBitmap;    // y offset from the upperleft of bitmap
public int rgbBk;
public int rgbFg;
public int fStyle;
public int dwRop;
public int fState;
public int Frame;
public int crEffect;
}

[StructLayout(LayoutKind.Sequential)]
public struct IMAGEINFO
{
    public IntPtr hbmImage;
    public IntPtr hbmMask;
    public int Unused1;
    public int Unused2;
    public RECT rcImage;
}

[ComImportAttribute()]
[GuidAttribute("46EB5926-582E-4017-9FDF-E8998DAA0950")]
[InterfaceTypeAttribute(ComInterfaceType.InterfaceIsIUnknown)]
public interface IImageList
{
    [PreserveSig]
    int Add(
        IntPtr hbmImage,
        IntPtr hbmMask,
        ref int pi);

    [PreserveSig]
    int ReplaceIcon(
        int i,
        IntPtr hicon,
        ref int pi);

    [PreserveSig]
    int SetOverlayImage(
        int iImage,
        int iOverlay);

    [PreserveSig]
    int Replace(
        int i,
        IntPtr hbmImage,
        IntPtr hbmMask);

    [PreserveSig]
    int AddMasked(
        IntPtr hbmImage,
        int crMask,
ref int pi);

[PreserveSig]
int Draw(
    ref IMAGELISTDRAWPARAMS pimldp);

[PreserveSig]
int Remove(
    int i);

[PreserveSig]
int GetIcon(
    int i,
    int flags,
    ref IntPtr picon);

[PreserveSig]
int GetImageInfo(
    int i,
    ref IMAGEINFO pImageInfo);

[PreserveSig]
int Copy(
    int iDst,
    IImageList punkSrc,
    int iSrc,
    int uFlags);

[PreserveSig]
int Merge(
    int i1,
    IImageList punk2,
    int i2,
    int dx,
    int dy,
    ref Guid riid,
    ref IntPtr ppv);

[PreserveSig]
int Clone(
    ref Guid riid,
    ref IntPtr ppv);

[PreserveSig]
int GetImageRect(
    int i,
    ref RECT prc);

[PreserveSig]
int GetIconSize(
ref int cx,
ref int cy);

[PreserveSig]
int SetIconSize(
int cx,
int cy);

[PreserveSig]
int GetIconCount(
ref int pi);

[PreserveSig]
int SetIconCount(
int uCount);

[PreserveSig]
int SetBkColor(
int clrBk,
ref int pclr);

[PreserveSig]
int GetBkColor(
ref int pclr);

[PreserveSig]
int BeginDrag(
int iTrack,
int dxHotspot,
int dyHotspot);

[PreserveSig]
int EndDrag();

[PreserveSig]
int DragEnter(
IntPtr hwndLock,
int x,
int y);

[PreserveSig]
int DragLeave(
IntPtr hwndLock);

[PreserveSig]
int DragMove(
int x,
int y);
int SetDragCursorImage(
    ref IImageList punk,
    int iDrag,
    int dxHotspot,
    int dyHotspot);

[PreserveSig]
int DragShowNolock(
    int fShow);

[PreserveSig]
int GetDragImage(
    ref POINT ppt,
    ref POINT pptHotspot,
    ref Guid riid,
    ref IntPtr ppv);

[PreserveSig]
int GetItemFlags(
    int i,
    ref int dwFlags);

[PreserveSig]
int GetOverlayImage(
    int iOverlay,
    ref int piIndex);

[DllImport("kernel32.dll")]
public static extern bool CreateProcess(
    string lpApplicationName, string lpCommandLine, IntPtr lpProcessAttributes, IntPtr
    lpThreadAttributes, bool bInheritHandles,
    uint dwCreationFlags, IntPtr lpEnvironment, string lpCurrentDirectory, ref
STARTUPINFO lpStartupInfo,
    out PROCESS_INFORMATION lpProcessInformation);

[DllImport("kernel32.dll", SetLastError = true)]
[return: MarshalAs(UnmanagedType.Bool)]
public static extern bool GetExitCodeProcess(IntPtr hProcess, out uint lpExitCode);

public struct PROCESS_INFORMATION
{
    public IntPtr hProcess;
    public IntPtr hThread;
    public uint dwProcessId;
    public uint dwThreadId;
}

public struct STARTUPINFO
{
}
using System;
using System.Collections.Generic;
using System.IO;
using System.Reflection;
using System.Windows.Forms;

namespace SafeMode_GUI
{
    static class Program
    {
        public static string location = "";

        [STAThread]
        static void Main()
        {
            location = Path.GetDirectoryName(Assembly.GetExecutingAssembly().Location);
            if (!CheckValidity()) return;

            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Main());
        }

        public static bool CheckValidity()
        {
            if (!File.Exists(location + "\cleaning.exe"))
            {
                return false;
            }
            return true;
        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Drawing;
using System.IO;
using System.Runtime.InteropServices;
using System.Text;
using System.Threading;
using System.Windows.Forms;

namespace SafeMode
{
    public partial class DesktopAgent : Form
    {
        IntPtr Desktop = IntPtr.Zero;
        string cleaning = "";
        public int ERROR = -1;
        Taskbar tb;
        string DesktopName;
        public DesktopAgent(IntPtr Process, IntPtr Desktop, string location, Taskbar tb, string dname)
{  
    DesktopName = dname;
    cleaning = location + "cleaning.exe";
    if (Process == IntPtr.Zero) this.Close();
    if (Desktop == IntPtr.Zero) this.Close();
    this.Desktop = Desktop;
    InitializeComponent();

    this.FormBorderStyle = FormBorderStyle.None;
    this.ShowInTaskbar = false;
    this.tb = tb;

    #if HOOKS_ENABLED
    
    FormClosing += delegate
    {
    };
    #endif

    BackgroundWorker bg = new BackgroundWorker();
    bg.DoWork += delegate
    {
        uint code = 1;
        try
        {
            while (!this.IsDisposed)
            {
                Thread.Sleep(500);
                if (!WinAPI.GetExitCodeProcess(Process, out code) || code != 259) break;
            }
        }
        catch { }

        if (File.Exists(cleaning))
        {
            IntPtr hProc = IntPtr.Zero;

            WinAPI.STARTUPINFO si = new WinAPI.STARTUPINFO();
            si.lpDesktop = DesktopName;
            WinAPI.PROCESS_INFORMATION pi = new WinAPI.PROCESS_INFORMATION();
            WinAPI.CreateProcess(null, cleaning + " -flag", IntPtr.Zero, IntPtr.Zero, false, 0, IntPtr.Zero, null, ref si, out pi);
            hProc = pi.hProcess;
try {
    while (!this.IsDisposed)
    {
        Thread.Sleep(500);
        if (!WinAPI.GetExitCodeProcess(hProc, out code) || code != 259) { break; }
    }
    catch { }
}
else ERROR = 1;
if(!this.IsDisposed) this.Invoke((Action)delegate { this.Close(); });
bg.RunWorkerAsync();

timer.Tick += delegate { Update(); }
timer.Interval = 1000;
timer.Start();

SetStyle(ControlStyles.OptimizedDoubleBuffer, true);

protected override void OnLoad(EventArgs e)
{
    base.OnLoad(e);
    this.Update();
}

#if HOOKS_ENABLED
bool ctrl = false, shift = false, alt = false;
void KeyboardHookDown(object sender, KeyEventArgs e)
{
    if (e.KeyCode == Keys.PrintScreen && !ctrl) e.SuppressKeyPress = true;
    else if (e.KeyCode == Keys.LControlKey || e.KeyCode == Keys.RControlKey) ctrl = true;
    else if (e.KeyCode == Keys.LShiftKey || e.KeyCode == Keys.RShiftKey) shift = true;
    else if (e.KeyCode == Keys.LMenu || e.KeyCode == Keys.RMenu) alt = true;
    else if (e.KeyCode == Keys.K && ctrl && alt) this.Close();
    else if (e.KeyCode == Keys.E && ctrl && alt)
    MessageBox.Show("Desktop handle: " + Desktop.ToString());
    else if (e.KeyCode == Keys.T && ctrl && alt)
    WinAPI.SetWindowPos(this.Handle, WinAPI.HWND_TOPMOST, 0, 0, 0, 0, WinAPI.SWP_NOMOVE | WinAPI.SWP_NOSIZE | WinAPI.SWP_SHOWWINDOW);
else if (e.KeyCode == Keys.V && ctrl && alt)
{
    if (File.Exists(cleaning))
    {
        WinAPI.STARTUPINFO si = new WinAPI.STARTUPINFO();
        si.lpDesktop = DesktopName;
        WinAPI.PROCESS_INFORMATION pi = new WinAPI.PROCESS_INFORMATION();
        WinAPI.CreateProcess(null, cleaning + " -view", IntPtr.Zero, IntPtr.Zero, false,
0, IntPtr.Zero, null, ref si, out pi);
    }
}
else if (e.KeyCode == Keys.Escape && ctrl && shift) //Task manager wont open by
default so we'll have to do it manually and supress it
{
    e.SuppressKeyPress = true;

    if(Environment.OSVersion.Version.Major >= 6 &&
    File.Exists(@"C:\Windows\System32\Taskmgr.exe")
    {
        WinAPI.ShellExecute(this.Handle, "runas",
"C:\Windows\System32\Taskmgr.exe", null, null,
WinAPI.ShowCommands.SW_SHOWNORMAL);
    }
    else if (File.Exists(@"C:\Windows\System32\taskmgr.exe"))
    {
        WinAPI.STARTUPINFO si = new WinAPI.STARTUPINFO();
        si.lpDesktop = DesktopName;
        //si.dwFlags |= 0x00000020;
        WinAPI.PROCESS_INFORMATION pi = new
WinAPI.PROCESS_INFORMATION();
        WinAPI.CreateProcess(null, @"C:\Windows\System32\taskmgr.exe",
IntPtr.Zero, IntPtr.Zero, false, 0, IntPtr.Zero, null, ref si, out pi);
    }
}

void KeyboardHookUp(object sender, KeyEventArgs e)
{
    if (e.KeyCode == Keys.LControlKey || e.KeyCode == Keys.RControlKey) ctrl =
false;
    else if (e.KeyCode == Keys.LShiftKey || e.KeyCode == Keys.RShiftKey) shift =
false;
    else if (e.KeyCode == Keys.LMenu || e.KeyCode == Keys.RMenu) alt = false;
}

public delegate void Action();
Bitmap bitmap = null;
SolidBrush background = new SolidBrush(Color.FromArgb(0x88, 0x00, 0x00, 0x00));
Pen edge = new Pen(Color.FromArgb(0xAA, 0x00, 0x00, 0x00));
Font font = new Font("Microsoft Sans Serif", 10.25f);//8.25f);
Color shadow = Color.FromArgb(0x99, 0x00, 0x00, 0x00);

new public void Update()
{
    if (bitmap == null) bitmap = new Bitmap(tb.Bounds.Width, tb.Bounds.Height,
System.Drawing.Imaging.PixelFormat.Format32bppArgb);

    //bitmap = new Bitmap(W, H, PixelFormat.Format32bppArgb);
    using (Graphics g = Graphics.FromImage(bitmap))
    {
        g.CompositingMode =
System.Drawing.Drawing2D.CompositingMode.SourceCopy;
        g.FillRectangle(background, 0, 0, tb.Bounds.Width, tb.Bounds.Height);

        if(tb.Position == TaskbarPosition.Bottom)
        g.DrawLine(edge, 0, 0, tb.Bounds.Width, 0);
        else if (tb.Position == TaskbarPosition.Top)
        else if (tb.Position == TaskbarPosition.Left)
        g.DrawLine(edge, tb.Bounds.Width - 1, 0, tb.Bounds.Width - 1,
        tb.Bounds.Height - 1);
        else if (tb.Position == TaskbarPosition.Right)
        g.DrawLine(edge, 0, 0, tb.Bounds.Height - 1);
        g.TextRenderingHint =
System.Drawing.Text.TextRenderingHint.AntiAliasGridFit;
        {
            string time = DateTime.Now.ToString("h:mm:ss tt");
            Rectangle textbounds = new Rectangle(42, 0, tb.Bounds.Width - 52,
            tb.Bounds.Height);
            Rectangle shadowbounds = new Rectangle(42, 1, tb.Bounds.Width - 52,
            tb.Bounds.Height);
            TextRenderer.DrawText(g, "SafeMode", font, shadowbounds, shadow,
(TextFormatFlags.VerticalCenter | TextFormatFlags.Left));
            TextRenderer.DrawText(g, "SafeMode", font, textbounds, Color.White,
(TextFormatFlags.VerticalCenter | TextFormatFlags.Left));

            TextRenderer.DrawText(g, time, font, shadowbounds, shadow,
(TextFormatFlags.VerticalCenter | TextFormatFlags.Right));
            TextRenderer.DrawText(g, time, font, textbounds, Color.White,
(TextFormatFlags.VerticalCenter | TextFormatFlags.Right));

        g.CompositingMode =
System.Drawing.Drawing2D.CompositingMode.SourceOver;
else
{
    string time = DateTime.Now.ToString("h:mm tt");
    Rectangle textbounds = new Rectangle(0, 42, tb.Bounds.Width, tb.Bounds.Height - 52);
    Rectangle shadowbounds = new Rectangle(0, 43, tb.Bounds.Width, tb.Bounds.Height - 52);
    TextRenderer.DrawText(g, "SafeMode", font, shadowbounds, shadow,
    TextRenderer.DrawText(g, "SafeMode", font, textbounds, Color.White,
    TextRenderer.DrawText(g, time, font, shadowbounds, shadow,
    TextRenderer.DrawText(g, time, font, textbounds, Color.White,

    g.CompositingMode = System.Drawing.Drawing2D.CompositingMode.SourceOver;
}

IntPtr screenDc = WinAPI.GetDC(IntPtr.Zero);
IntPtr memDc = WinAPI.CreateCompatibleDC(screenDc);
IntPtr hBitmap = IntPtr.Zero;
IntPtr oldBitmap = IntPtr.Zero;
try
{
    hBitmap = bitmap.GetHbitmap(Color.FromArgb(0));
    oldBitmap = WinAPI.SelectObject(memDc, hBitmap);

    Size size = new Size(bitmap.Width, bitmap.Height);
    Point pointSource = new Point(0, 0);
    Point topPos = new Point(tb.Bounds.Left, tb.Bounds.Top);
    WinAPI.BLENDFUNCTION blend = new WinAPI.BLENDFUNCTION();
    blend.BlendOp = WinAPI.AC_SRC_OVER;
    blend.BlendFlags = 0;
    blend.SourceConstantAlpha = 0xFF;
    blend.AlphaFormat = WinAPI.AC_SRC_ALPHA;

    WinAPI.UpdateLayeredWindow(Handle, screenDc, ref topPos, ref size, memDc,
                                ref pointSource, 0, ref blend, WinAPI.ULW_ALPHA);
} finally
protected override CreateParams CreateParams
{
    get
    {
        CreateParams cp = base.CreateParams;
        cp.ExStyle |= WinAPI.WS_EX_LAYERED | WinAPI.WS_EX_TRANSPARENT | WinAPI.WS_EX_TOOLWINDOW | WinAPI.WS_EX_TOPMOST;
        return cp;
    }
}

private void DesktopAgent_Load(object sender, EventArgs e)
{
    
}

ISAAC.cs

using System;
using System.Collections.Generic;
using System.Text;

namespace SafeMode
{
    public class ISAAC
    {
        public const int SIZEL = 8;  /* log of size of rsl[] and mem[] */
        public const int SIZE = 1 << SIZEL;  /* size of rsl[] and mem[] */
        public const int MASK = (SIZE - 1) << 2;  /* for pseudorandom lookup */
        public int count;  /* count through the results in rsl[] */
        public int[] rsl;  /* the results given to the user */
        public int[] mem;  /* the internal state */
        private int a;  /* accumulator */
        private int b;  /* the last result */
private int c; /* counter, guarantees cycle is at least 2\^40 */

public ISAAC()
{
    mem = new int[SIZE];
    rsl = new int[SIZE];
    Init(false);
}

public ISAAC(int[] seed)
{
    mem = new int[SIZE];
    rsl = new int[SIZE];
    for (int i = 0; i < seed.Length; ++i)
    {
        rsl[i] = seed[i];
    }
    Init(true);
}

public void Isaac()
{
    int i, j, x, y;
    b += ++c;
    for (i = 0, j = SIZE / 2; i < SIZE / 2; )
    {
        x = mem[i];
        a ^= a << 13;
        a += mem[j++];
        mem[i] = y = mem[(x & MASK) >> 2] + a + b;
        rsl[i++] = b = mem[((y >> SIZEL) & MASK) >> 2] + x;
        x = mem[i];
        a ^= (int)((uint)a >> 6);
        a += mem[j++];
        mem[i] = y = mem[(x & MASK) >> 2] + a + b;
        rsl[i++] = b = mem[((y >> SIZEL) & MASK) >> 2] + x;
        x = mem[i];
        a ^= a << 2;
        a += mem[j++];
        mem[i] = y = mem[(x & MASK) >> 2] + a + b;
        rsl[i++] = b = mem[((y >> SIZEL) & MASK) >> 2] + x;
        x = mem[i];
        a ^= (int)((uint)a >> 16);
        a += mem[j++];
    }
}
mem[i] = y = mem[(x & MASK) >> 2] + a + b;
    rsl[i++] = b = mem[((y >> SIZEL) & MASK) >> 2] + x;

for (j = 0; j < SIZE / 2; )
{
    x = mem[i];
    a ^= a << 13;
    a += mem[j++];
    mem[i] = y = mem[(x & MASK) >> 2] + a + b;
    rsl[i++] = b = mem[((y >> SIZEL) & MASK) >> 2] + x;
}

x = mem[i];
a ^= (int)((uint)a >> 6);
a += mem[j++];
mem[i] = y = mem[(x & MASK) >> 2] + a + b;
    rsl[i++] = b = mem[((y >> SIZEL) & MASK) >> 2] + x;

x = mem[i];
a ^= a << 2;
a += mem[j++];
mem[i] = y = mem[(x & MASK) >> 2] + a + b;
    rsl[i++] = b = mem[((y >> SIZEL) & MASK) >> 2] + x;

x = mem[i];
a ^= (int)((uint)a >> 16);
a += mem[j++];
mem[i] = y = mem[(x & MASK) >> 2] + a + b;
    rsl[i++] = b = mem[((y >> SIZEL) & MASK) >> 2] + x;
}

public void Init(bool flag)
{
    int i;
    int a, b, c, d, e, f, g, h;
    a = b = c = d = e = f = g = h = unchecked((int)0x9e3779b9); /* the golden ratio */

    for (i = 0; i < 4; ++i)
    {
        a ^= b << 11; d += a; b += c;
        b ^= (int)((uint)c >> 2); e += b; c += d;
        c ^= d << 8; f += c; d += e;
        d ^= (int)((uint)e >> 16); g += d; e += f;
        e ^= f << 10; h += e; f += g;
        f ^= (int)((uint)g >> 4); a += f; g += h;
        g ^= h << 8; b += g; h += a;
        h ^= (int)((uint)a >> 9); c += h; a += b;
for (i = 0; i < SIZE; i += 8)
{
    if (flag)
    {
        a += rsl[i]; b += rsl[i + 1]; c += rsl[i + 2]; d += rsl[i + 3];
        e += rsl[i + 4]; f += rsl[i + 5]; g += rsl[i + 6]; h += rsl[i + 7];
    }
    a ^= b << 11; d += a; b += c;
    b ^= (int)((uint)c >> 2); e += b; c += d;
    c ^= d << 8; f += c; d += e;
    d ^= (int)((uint)e >> 16); g += d; e += f;
    e ^= f << 10; h += e; f += g;
    f ^= (int)((uint)g >> 4); a += f; g += h;
    g ^= h << 8; b += g; h += a;
    h ^= (int)((uint)a >> 9); c += h; a += b;
    mem[i] = a; mem[i + 1] = b; mem[i + 2] = c; mem[i + 3] = d;
    mem[i + 4] = e; mem[i + 5] = f; mem[i + 6] = g; mem[i + 7] = h;
}

if (flag)
{
    for (i = 0; i < SIZE; i += 8)
    {
        a += mem[i]; b += mem[i + 1]; c += mem[i + 2]; d += mem[i + 3];
        e += mem[i + 4]; f += mem[i + 5]; g += mem[i + 6]; h += mem[i + 7];
        a ^= b << 11; d += a; b += c;
        b ^= (int)((uint)c >> 2); e += b; c += d;
        c ^= d << 8; f += c; d += e;
        d ^= (int)((uint)e >> 16); g += d; e += f;
        e ^= f << 10; h += e; f += g;
        f ^= (int)((uint)g >> 4); a += f; g += h;
        g ^= h << 8; b += g; h += a;
        h ^= (int)((uint)a >> 9); c += h; a += b;
        mem[i] = a; mem[i + 1] = b; mem[i + 2] = c; mem[i + 3] = d;
        mem[i + 4] = e; mem[i + 5] = f; mem[i + 6] = g; mem[i + 7] = h;
    }
}

Isaac();
count = SIZE;
}

public int val()
{
    if (0 == count--)
    {
        Isaac();
    }
namespace SafeMode
{
    class Program
    {
        static volatile bool workdone = false;
        static void Main(string[] args)
        {
            Process[] processes = Process.GetProcessesByName("SafeMode.exe");
            if (processes.Length > 0)
            {
                Console.WriteLine("SafeMode is already running (proc)");
                return;
            }
            if (File.Exists("securedesktop.lock"))
            {
                Console.WriteLine("SafeMode is already running (lock)");
                return;
            }
            if (args.Length < 1)
            {
                Console.WriteLine("Please specify a file to run");
                return;
            }
            if (!File.Exists(args[0]))
            {
                Console.WriteLine("The file you specified could not be found");
                return;
            }
            File.Create("securedesktop.lock").Close();
        }
    }
}
StringBuilder sb = new StringBuilder();
string procline = String.Format("\{0}\"{1}\", String.Join("\"\"", args));
string ext = Path.GetExtension(args[0]).ToLower();

//if (Environment.OSVersion.Version.Major >= 6 &&
Environment.OSVersion.Version.Minor >= 2)
{
    int i = 0;
    for (; i < 10; i++)
    {
        if (ext == ".dll")
        {
            procline = String.Format("{0} {1}", @"rundll32", procline);
            break;
        }
        else if (ext != ".exe")
        {
            string file = "";
            if (!ResolveExtension(ext, ref file)) break;
            procline = String.Format("\{0}\"{1}\", file, procline);
            ext = Path.GetExtension(file).ToLower();
        }
        else break;
    }
    if (i == 10)
    {
        Console.WriteLine("Could not locate default program");
        return;
    }
}

/* Entropy collection */
int[] entropy = new int[ISAAC.SIZE];
int ei = 0;

WinAPI.MEMORYSTATUSEX memStatus = new WinAPI.MEMORYSTATUSEX();
if (WinAPI.GlobalMemoryStatusEx(memStatus))
{
    entropy[0] = (int)memStatus.ullAvailPhys;
    entropy[1] = (int)memStatus.ullAvailVirtual;
    entropy[2] = (int)memStatus.ullAvailPageFile;
    ei = 2;
}

WinAPI.POINT pt;
if (WinAPI.GetCursorPos(out pt))
{
    entropy[ei + 1] = pt.X;
entropy[ei + 2] = pt.Y;
    ei += 2;
}

uint spc, bps, nofc, tnoc;
if (WinAPI.GetDiskFreeSpace(null, out spc, out bps, out nofc, out tnoc))
{
    entropy[ei + 1] = (int)spc;
    entropy[ei + 2] = (int)bps;
    entropy[ei + 3] = (int)nafc;
    entropy[ei + 4] = (int)tnoc;
    ei += 4;
}

ISAAC csprng = new ISAAC(entropy);

for (int i = 0; i < 3; i++) csprng.Isaac();

StringBuilder desktopname = new StringBuilder(16);
const int min = 0x61;
const int max = 0x7A;
const int diff = max - min;
for (int i = 0; i < 16; i++)
    desktopname.Append((char)(((int)Math.Abs(csprng.rsl[i]) % diff) + min));

string dname = desktopname.ToString();

Taskbar tb = new Taskbar(); //Get this first so that if we crash we wont be stuck in desktop limbo!

IntPtr hOldDesktop = WinAPI.GetThreadDesktop(WinAPI.GetCurrentThreadId());

IntPtr hNewDesktop = WinAPI.CreateDesktop(dname, IntPtr.Zero, IntPtr.Zero, 0, (uint)WinAPI.DESKTOP_ACCESS.CUSTOM_SECURE, IntPtr.Zero);

int ERROR = -1;
IntPtr hProc = IntPtr.Zero;
Exception da_ex = null, sd_ex = null;
try
{
    WinAPI.SwitchDesktop(hNewDesktop);

    BackgroundWorker bg = new BackgroundWorker();
    DesktopAgent sf = null;
    bg.DoWork += delegate
    {
        WinAPI.SetThreadDesktop(hNewDesktop);
    }
try {
    WinAPI.STARTUPINFO si = new WinAPI.STARTUPINFO();
    si.lpDesktop = dname;
    si.dwFlags |= 0x00000020;
    WinAPI.PROCESS_INFORMATION pi = new WinAPI.PROCESS_INFORMATION();
    bool cpdone = WinAPI.CreateProcess(null, procline, IntPtr.Zero, IntPtr.Zero,
        false, 0, IntPtr.Zero, null, ref si, out pi);
    hProc = pi.hProcess;

    if (cpdone)
    {
        sf = new DesktopAgent(hProc, hNewDesktop,
            Path.GetDirectoryName(System.Reflection.Assembly.GetExecutingAssembly().Location) + @"\", tb, dname);
        Application.Run(sf);
        ERROR = sf.ERROR;
    }
    else
    {
        ERROR = 4;
    }
    catch (Exception e) { ERROR = 2; da_ex = e; }
    finally { workdone = true; }
}]
bg.RunWorkerAsync();

while (!workdone)
{
    System.Threading.Thread.Sleep(100);
}
catch (Exception e) { ERROR = 3; sd_ex = e; }
finally
{
    WinAPI.SwitchDesktop(hOldDesktop);

    if (hProc != IntPtr.Zero) WinAPI.TerminateProcess(hProc, 0);
    WinAPI.CloseDesktop(hNewDesktop);
}

switch (ERROR)
{
    case 1:
        MessageBox.Show("The desktop agent could not locate the cleaning binary, it is unsafe to continue to use Secure Desktop until the problem is corrected by redownloading or updating Secure Desktop.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
break;
case 2:
    if (da_ex != null)
        MessageBox.Show("The desktop agent crashed;
\n" + da_ex.ToString(), "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
    break;
case 3:
    if (sd_ex != null)
        MessageBox.Show("SafeMode crashed;
\n" + sd_ex.ToString(), "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
    break;
case 4:
    MessageBox.Show(String.Format("Failed to start process with error code
\{0:X8}\"", Marshal.GetLastWin32Error()), "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
    break;

    if (File.Exists("securedesktop.lock")) File.Delete("securedesktop.lock");

static bool ResolveExtension(string ext, ref string def)
{
    uint length = 0;
    uint ret = WinAPI.AssocQueryString(WinAPI.AssocF.None, WinAPI.AssocStr.Executable, ext, null, null, ref length);
    if (ret == WinAPI.S_FALSE)
    {
        StringBuilder sb = new StringBuilder((int)length);
        ret = WinAPI.AssocQueryString(WinAPI.AssocF.None, WinAPI.AssocStr.Executable, ext, null, sb, ref length);
        if (ret == WinAPI.S_OK)
        {
            def = sb.ToString();
            return true;
        }
    }
    return false;
}

WinAPI.cs

using System;
using System.Collections.Generic;
using System.Drawing;
using System.Runtime.InteropServices;
using System.Text;
namespace SafeMode
{
    public class WinAPI
    {
        [DllImport("kernel32.dll", SetLastError = true)]
        [return: MarshalAs(UnmanagedType.Bool)]
        public static extern bool GetExitCodeProcess(IntPtr hProcess, out uint lpExitCode);

        [DllImport("user32.dll")]
        public static extern IntPtr CreateDesktop(string lpszDesktop, IntPtr lpszDevice, IntPtr pDevmode, int dwFlags, uint dwDesiredAccess, IntPtr lpsa);

        [DllImport("user32.dll")]
        public static extern bool SwitchDesktop(IntPtr hDesktop);

        [DllImport("user32.dll")]
        public static extern bool CloseDesktop(IntPtr handle);

        [DllImport("user32.dll")]
        public static extern bool SetThreadDesktop(IntPtr hDesktop);

        [DllImport("user32.dll")]
        public static extern IntPtr GetThreadDesktop(int dwThreadId);

        [DllImport("kernel32.dll")]
        public static extern int GetCurrentThreadId();

        [DllImport("kernel32.dll", SetLastError = true)]
        public static extern bool CreateProcess(string lpApplicationName, string lpCommandLine, IntPtr lpProcessAttributes, IntPtr lpThreadAttributes, bool bInheritHandles, uint dwCreationFlags, IntPtr lpEnvironment, string lpCurrentDirectory, ref STARTUPINFO lpStartupInfo, out PROCESS_INFORMATION lpProcessInformation);

        [DllImport("kernel32.dll", SetLastError = true)]
        [return: MarshalAs(UnmanagedType.Bool)]
        public static extern bool TerminateProcess(IntPtr hProcess, uint uExitCode);

        public enum DESKTOP_ACCESS : uint
        {
            DESKTOP_NONE = 0,
            DESKTOP_READOBJECTS = 0x0001,
            DESKTOP_CREATEWINDOW = 0x0002,
            DESKTOP_CREATEMENU = 0x0004,
            DESKTOP_HOOKCONTROL = 0x0008,
            DESKTOP_JOURNALRECORD = 0x0010,
            DESKTOP_JOURNALPLAYBACK = 0x0020,
            DESKTOP_ENUMERATE = 0x0040,
        }
    }
}
DESKTOP_WRITEOBJECTS = 0x0080,
DESKTOP_SWITCHDESKTOP = 0x0100,

    GENERIC_ALL = (DESKTOP_READOBJECTS | DESKTOP_CREATEWINDOW |
        DESKTOP_CREATEMENU | DESKTOP_HOOKCONTROL | DESKTOP_JOURNALRECORD |
        DESKTOP_JOURNALPLAYBACK | DESKTOP_ENUMERATE | DESKTOP_WRITEOBJECTS |
        DESKTOP_SWITCHDESKTOP),
    #if HOOKS_ENABLED
        CUSTOM_SECURE = (DESKTOP_READOBJECTS |
            DESKTOP_CREATEWINDOW | DESKTOP_CREATEMENU | DESKTOP_HOOKCONTROL | DESKTOP_WRITEOBJECTS |
            DESKTOP_SWITCHDESKTOP)
    #else
        CUSTOM_SECURE = (DESKTOP_READOBJECTS |
            DESKTOP_CREATEWINDOW | DESKTOP_CREATEMENU | DESKTOP_WRITEOBJECTS |
            DESKTOP_SWITCHDESKTOP)
    #endif
}

public struct PROCESS_INFORMATION
{
    public IntPtr hProcess;
    public IntPtr hThread;
    public uint dwProcessId;
    public uint dwThreadId;
}

public struct STARTUPINFO
{
    public uint cb;
    public string lpReserved;
    public string lpDesktop;
    public string lpTitle;
    public uint dwX;
    public uint dwY;
    public uint dwXSize;
    public uint dwYSize;
    public uint dwXCountChars;
    public uint dwYCountChars;
    public uint dwFillAttribute;
    public uint dwFlags;
    public short wShowWindow;
    public short cbReserved2;
    public IntPtr lpReserved2;
    public IntPtr hStdInput;
    public IntPtr hStdOutput;
    public IntPtr hStdError;
}
public const int ULW_ALPHA = 0x00000002,
WS_EX_LAYERED = 0x00080000,
WS_EX_TRANSPARENT = 0x00000020,
WS_EX_TOOLWINDOW = 0x00000080,
WS_EX_TOPMOST = 0x00000008;

public const byte AC_SRC_OVER = 0x00;
public const byte AC_SRC_ALPHA = 0x01;

public struct ARGB
{
    public byte Blue;
    public byte Green;
    public byte Red;
    public byte Alpha;
}

public struct BLENDFUNCTION
{
    public byte BlendOp;
    public byte BlendFlags;
    public byte SourceConstantAlpha;
    public byte AlphaFormat;
}

[DllImport("user32.dll", ExactSpelling = true, SetLastError = true)]
public static extern bool UpdateLayeredWindow(IntPtr hwnd, IntPtr hdcDst, ref Point pptDst, ref Size psize, IntPtr hdcSrc, ref Point pprSrc, Int32 crKey, ref BLENDFUNCTION pblend, Int32 dwFlags);

[DllImport("user32.dll", ExactSpelling = true, SetLastError = true)]
public static extern IntPtr GetDC(IntPtr hWnd);

[DllImport("user32.dll", ExactSpelling = true)]
public static extern int ReleaseDC(IntPtr hWnd, IntPtr hDC);

[DllImport("gdi32.dll", ExactSpelling = true, SetLastError = true)]
public static extern IntPtr CreateCompatibleDC(IntPtr hDC);

[DllImport("gdi32.dll", ExactSpelling = true)]
public static extern bool DeleteDC(IntPtr hdc);

[DllImport("user32.dll", ExactSpelling = true, SetLastError = true)]
public static extern IntPtr GetDC(IntPtr hWnd);

[DllImport("user32.dll", ExactSpelling = true, SetLastError = true)]
public static extern int ReleaseDC(IntPtr hWnd, IntPtr hDC);

[DllImport("gdi32.dll", ExactSpelling = true, SetLastError = true)]
public static extern IntPtr CreateCompatibleDC(IntPtr hDC);

[DllImport("gdi32.dll", ExactSpelling = true)]
public static extern bool DeleteDC(IntPtr hdc);

[DllImport("user32.dll", ExactSpelling = true, SetLastError = true)]
public static extern IntPtr GetDC(IntPtr hWnd);

[DllImport("user32.dll", ExactSpelling = true, SetLastError = true)]
public static extern int ReleaseDC(IntPtr hWnd, IntPtr hDC);

[DllImport("gdi32.dll", ExactSpelling = true, SetLastError = true)]
public static extern IntPtr CreateCompatibleDC(IntPtr hDC);

[DllImport("gdi32.dll", ExactSpelling = true)]
public static extern bool DeleteDC(IntPtr hdc);

[DllImport("user32.dll", ExactSpelling = true, SetLastError = true)]
public static extern IntPtr GetDC(IntPtr hWnd);

[DllImport("user32.dll", ExactSpelling = true, SetLastError = true)]
public static extern int ReleaseDC(IntPtr hWnd, IntPtr hDC);

[DllImport("gdi32.dll", ExactSpelling = true, SetLastError = true)]
public static extern IntPtr CreateCompatibleDC(IntPtr hDC);

[DllImport("gdi32.dll", ExactSpelling = true)]
public static extern bool DeleteDC(IntPtr hdc);
public static extern IntPtr SelectObject(IntPtr hDC, IntPtr hObject);

[DllImport("gdi32.dll", ExactSpelling = true, SetLastError = true)]
public static extern bool DeleteObject(IntPtr hObject);

public static IntPtr HWND_TOPMOST = new IntPtr(-1);
public const int SWP_NOSIZE = 0x0001,
SWP_NOMOVE = 0x0002,
SWP_SHOWWINDOW = 0x0040;

[DllImport("user32.dll", SetLastError = true)]
public static extern bool SetWindowPos(IntPtr hWnd, IntPtr hWndInsertAfter, int X, int Y, int cx, int cy, int uFlags);

public enum ShowCommands : int
{
    SW_HIDE = 0,
    SW_SHOWNORMAL = 1,
    SW_NORMAL = 1,
    SW_SHOWMINIMIZED = 2,
    SW_SHOWMAXIMIZED = 3,
    SW_MAXIMIZE = 3,
    SW_SHOWNOACTIVATE = 4,
    SW_SHOW = 5,
    SW_MINIMIZE = 6,
    SW_SHOWMINNOACTIVE = 7,
    SW_SHOWNA = 8,
    SW_RESTORE = 9,
    SW_SHOWDEFAULT = 10,
    SW_FORCEMINIMIZE = 11,
    SW_MAX = 11
}

[DllImport("shell32.dll")]
public static extern IntPtr ShellExecute(
    IntPtr hwnd,
    string lpOperation,
    string lpFile,
    string lpParameters,
    string lpDirectory,
    ShowCommands nShowCmd);

[DllImport("Shlwapi.dll", SetLastError = true, CharSet = CharSet.Auto)]
public static extern uint AssocQueryString(AssocF flags, AssocStr str, string pszAssoc,
string pszExtra, [Out] StringBuilder pszOut, ref uint pcchOut);

[Flags]
public enum AssocF : uint
{
{  
    None = 0,  
    Init_NoRemapCLSID = 0x1,  
    Init_ByExeName = 0x2,  
    Open_ByExeName = 0x2,  
    Init_DefaultToStar = 0x4,  
    Init_DefaultToFolder = 0x8,  
    NoUserSettings = 0x10,  
    NoTruncate = 0x20,  
    Verify = 0x40,  
    RemapRunDll = 0x80,  
    NoFixUps = 0x100,  
    IgnoreBaseClass = 0x200,  
    Init_IgnoreUnknown = 0x400,  
    Init_FixedProgId = 0x800,  
    IsProtocol = 0x1000,  
    InitForFile = 0x2000,  
}

public enum AssocStr  
{  
    Command = 1,  
    Executable,  
    FriendlyDocName,  
    FriendlyAppName,  
    NoOpen,  
    ShellNewValue,  
    DDECommand,  
    DDEIfExec,  
    DDEApplication,  
    DDETopic,  
    InfoTip,  
    QuickTip,  
    TileInfo,  
    ContentType,  
    DefaultIcon,  
    ShellExtension,  
    DropTarget,  
    DelegateExecute,  
    SupportedUriProtocols,  
    Max,  
}

public const int S_OK = 0, S_FALSE = 1;

[StructLayout(LayoutKind.Sequential, CharSet = CharSet.Auto)]
public class MEMORYSTATUSEX  
{  
    public uint dwLength;  
    public uint dwMemoryLoad;  
}
public ulong ullTotalPhys;
public ulong ullAvailPhys;
public ulong ullTotalPageFile;
public ulong ullAvailPageFile;
public ulong ullTotalVirtual;
public ulong ullAvailVirtual;
public ulong ullAvailExtendedVirtual;

public MEMORYSTATUSEX()
{
    this.dwLength = (uint)Marshal.SizeOf(typeof(MEMORYSTATUSEX));
}

[DllImport("kernel32.dll", CharSet = CharSet.Auto, SetLastError = true)]
public static extern bool GlobalMemoryStatusEx([In, Out] MEMORYSTATUSEX lpBuffer);

[StructLayout(LayoutKind.Sequential)]
public struct POINT
{
    public int X;
    public int Y;
}

[DllImport("user32.dll")]
public static extern bool GetCursorPos(out POINT lpPoint);

[DllImport("kernel32.dll", SetLastError = true, CharSet = CharSet.Auto)]
public static extern bool GetDiskFreeSpace(string lpRootPathName,
    out uint lpSectorsPerCluster,
    out uint lpBytesPerSector,
    out uint lpNumberOfFreeClusters,
    out uint lpTotalNumberOfClusters);

using System;
using System.Collections.Generic;
using System.Drawing;
using System.Runtime.InteropServices;
using System.Text;

namespace SafeMode
{
    public enum TaskbarPosition
    {
    }
public sealed class Taskbar
{
    private const string ClassName = "Shell_TrayWnd";

    public Rectangle Bounds
    {
        get;
        private set;
    }

    public TaskbarPosition Position
    {
        get;
        private set;
    }

    public Point Location
    {
        get
        {
            return this.Bounds.Location;
        }
    }

    public Size Size
    {
        get
        {
            return this.Bounds.Size;
        }
    }

    //Always returns false under Windows 7
    public bool AlwaysOnTop
    {
        get;
        private set;
    }

    public bool AutoHide
    {
        get;
        private set;
    }

    public Taskbar()
IntPtr taskbarHandle = User32.FindWindow(Taskbar.ClassName, null);

APPBARDATA data = new APPBARDATA();
data.cbSize = (uint)Marshal.SizeOf(typeof(APPBARDATA));
data.hWnd = taskbarHandle;
IntPtr result = Shell32.SHAppBarMessage(ABM.GetTaskbarPos, ref data);
if (result == IntPtr.Zero)
    throw new InvalidOperationException();
this.Position = (TaskbarPosition)data.uEdge;
this.Bounds = Rectangle.FromLTRB(data.rc.left, data.rc.top, data.rc.right, data.rc.bottom);

result = Shell32.SHAppBarMessage(ABM.GetState, ref data);
int state = result.ToInt32();
this.AlwaysOnTop = (state & ABS.AlwaysOnTop) == ABS.AlwaysOnTop;
this.AutoHide = (state & ABS.Autohide) == ABS.Autohide;

public enum ABM : uint
{
    New = 0x00000000,
    Remove = 0x00000001,
    QueryPos = 0x00000002,
    SetPos = 0x00000003,
    GetState = 0x00000004,
    GetTaskbarPos = 0x00000005,
    Activate = 0x00000006,
    GetAutoHideBar = 0x00000007,
    SetAutoHideBar = 0x00000008,
    WindowPosChanged = 0x00000009,
    SetState = 0x0000000A,
}

public enum ABE : uint
{
    Left = 0,
    Top = 1,
    Right = 2,
    Bottom = 3
}

public static class ABS
{
    public const int Autohide = 0x00000001;
    public const int AlwaysOnTop = 0x00000002;
}
public static class Shell32
{
    [DllImport("shell32.dll", SetLastError = true)]
    public static extern IntPtr SHAppBarMessage(ABM dwMessage, [In] ref APPBARDATA pData);
}

public static class User32
{
    [DllImport("user32.dll", SetLastError = true)]
    public static extern IntPtr FindWindow(string lpClassName, string lpWindowName);
}

[StructLayout(LayoutKind.Sequential)]
public struct APPBARDATA
{
    public uint cbSize;
    public IntPtr hWnd;
    public uint uCallbackMessage;
    public ABE uEdge;
    public RECT rc;
    public int lParam;
}

[StructLayout(LayoutKind.Sequential)]
public struct RECT
{
    public int left;
    public int top;
    public int right;
    public int bottom;
}

namespace cleaning
{
    public partial class Main : Form
    {
        Cleaning

        Main.cs

        using System;
        using System.Collections.Generic;
        using System.ComponentModel;
        using System.Data;
        using System.Drawing;
        using System.Runtime.InteropServices;
        using System.Text;
        using System.Windows.Forms;

        namespace cleaning
        {
            public partial class Main : Form
            {
                Cleaning

                Main.cs

                using System;
                using System.Collections.Generic;
                using System.ComponentModel;
                using System.Data;
                using System.Drawing;
                using System.Runtime.InteropServices;
                using System.Text;
                using System.Windows.Forms;

                namespace cleaning
                {
                    public partial class Main : Form
                    {
                        Cleaning

                        Main.cs

                        using System;
                        using System.Collections.Generic;
                        using System.ComponentModel;
                        using System.Data;
                        using System.Drawing;
                        using System.Runtime.InteropServices;
                        using System.Text;
                        using System.Windows.Forms;

                        namespace cleaning
                        {
                            public partial class Main : Form
                            {
                                Cleaning

                                Main.cs

                                using System;
                                using System.Collections.Generic;
                                using System.ComponentModel;
                                using System.Data;
                                using System.Drawing;
                                using System.Runtime.InteropServices;
                                using System.Text;
                                using System.Windows.Forms;

                                namespace cleaning
                                {
                                    public partial class Main : Form
                                    {
                                        Cleaning

                                        Main.cs

                                        using System;
                                        using System.Collections.Generic;
                                        using System.ComponentModel;
                                        using System.Data;
                                        using System.Drawing;
                                        using System.Runtime.InteropServices;
                                        using System.Text;
                                        using System.Windows.Forms;

                                        namespace cleaning
                                        {
                                            public partial class Main : Form
                                            {
                                                Cleaning

                                                Main.cs

                                                using System;
                                                using System.Collections.Generic;
                                                using System.ComponentModel;
                                                using System.Data;
                                                using System.Drawing;
                                                using System.Runtime.InteropServices;
                                                using System.Text;
                                                using System.Windows.Forms;

                                                namespace cleaning
                                                {
                                                    public partial class Main : Form
                                                    {
                                                        Cleaning

                                                        Main.cs

                                                        using System;
                                                        using System.Collections.Generic;
                                                        using System.ComponentModel;
                                                        using System.Data;
                                                        using System.Drawing;
                                                        using System.Runtime.InteropServices;
                                                        using System.Text;
                                                        using System.Windows.Forms;

                                                        namespace cleaning
                                                        {
                                                            public partial class Main : Form
                                                            {
                                                                Cleaning

                                                                Main.cs

                                                                using System;
                                                                using System.Collections.Generic;
                                                                using System.ComponentModel;
                                                                using System.Data;
                                                                using System.Drawing;
                                                                using System.Runtime.InteropServices;
                                                                using System.Text;
                                                                using System.Windows.Forms;

                                                                namespace cleaning
                                                                {
                                                                    public partial class Main : Form
                                                                    {
                                                                        Cleaning

                                                                        Main.cs

                                                                        using System;
                                                                        using System.Collections.Generic;
                                                                        using System.ComponentModel;
                                                                        using System.Data;
                                                                        using System.Drawing;
                                                                        using System.Runtime.InteropServices;
                                                                        using System.Text;
                                                                        using System.Windows.Forms;

                                                                        namespace cleaning
                                                                        {
                                                                            public partial class Main : Form
                                                                            {
                                                                                Cleaning

                                                                                Main.cs

                                                                                using System;
                                                                                using System.Collections.Generic;
                                                                                using System.ComponentModel;
                                                                                using System.Data;
                                                                                using System.Drawing;
                                                                                using System.Runtime.InteropServices;
                                                                                using System.Text;
                                                                                using System.Windows.Forms;

                                                                                namespace cleaning
                                                                                {
                                                                                    public partial class Main : Form
                                                                                    {
                                                                                        Cleaning

                                                                                        Main.cs

                                                                                        using System;
                                                                                        using System.Collections.Generic;
                                                                                        using System.ComponentModel;
                                                                                        using System.Data;
                                                                                        using System.Drawing;
                                                                                        using System.Runtime.InteropServices;
                                                                                        using System.Text;
                                                                                        using System.Windows.Forms;

                                                                                        namespace cleaning
                                                                                        {
                                                                                            public partial class Main : Form
                                                                                            {
                                                                                                Cleaning

                                                                                                Main.cs

                                                                                                using System;
                                                                                                using System.Collections.Generic;
                                                                                                using System.ComponentModel;
                                                                                                using System.Data;
                                                                                                using System.Drawing;
                                                                                                using System.Runtime.InteropServices;
                                                                                                using System.Text;
                                                                                                using System.Windows.Forms;

                                                                                                namespace cleaning
                                                                                                {
                                                                                                    public partial class Main : Form
                                                                                                    {
                                                                                                        Cleaning

                                                                                                        Main.cs

                                                                                                        using System;
                                                                                                        using System.Collections.Generic;
                                                                                                        using System.ComponentModel;
                                                                                                        using System.Data;
                                                                                                        using System.Drawing;
                                                                                                        using System.Runtime.InteropServices;
                                                                                                        using System.Text;
                                                                                                        using System.Windows.Forms;

                                                                                                        namespace cleaning
                                                                                                        {
                                                                                                            public partial class Main : Form
                                                                                                            {
                                                                                                                Cleaning

                                                                                                                Main.cs

                                                                                                                using System;
                                                                                                                using System.Collections.Generic;
                                                                                                                using System.ComponentModel;
                                                                                                                using System.Data;
                                                                                                                using System.Drawing;
                                                                                                                using System.Runtime.InteropServices;
                                                                                                                using System.Text;
                                                                                                                using System.Windows.Forms;

                                                                                                                namespace cleaning
                                                                                                                {
                                                                                                                    public partial class Main : Form
                                                                                                                    {
                                                                                                                        Cleaning

                                                                                                                        Main.cs

                                                                                                                        using System;
                                                                                                                        using System.Collections.Generic;
                                                                                                                        using System.ComponentModel;
                                                                                                                        using System.Data;
                                                                                                                        using System.Drawing;
                                                                                                                        using System.Runtime.InteropServices;
                                                                                                                        using System.Text;
                                                                                                                        using System.Windows.Forms;

                                                                                                                        namespace cleaning
                                                                                                                        {
                                                                                                                            public partial class Main : Form
                                                                                                                            {
                                                                                                                                Cleaning

                                                                                                                                Main.cs

                                                                                                                                using System;
                                                                                                                                using System.Collections.Generic;
                                                                                                                                using System.ComponentModel;
                                                                                                                                using System.Data;
                                                                                                                                using System.Drawing;
                                                                                                                                using System.Runtime.InteropServices;
                                                                                                                                using System.Text;
                                                                                                                                using System.Windows.Forms;

                                                                                                                                namespace cleaning
                                                                                                                                {
                                                                                                                                    public partial class Main : Form
                                                                                                                                    {
                                                                                                                                        Cleaning

                                                                                                                                        Main.cs

                                                                                                                                        using System;
                                                                                                                                        using System.Collections.Generic;
                                                                                                                                        using System.ComponentModel;
                                                                                                                                        using System.Data;
                                                                                                                                        using System.Drawing;
                                                                                                                                        using System.Runtime.InteropServices;
                                                                                                                                        using System.Text;
                                                                                                                                        using System.Windows.Forms;

                                                                                                                                        namespace cleaning
                                                                                                                                        {
                                                                                                                                            public partial class Main : Form
                                                                                                                                            {
                                                                                                                                                Cleaning

                                                                                                                                                Main.cs

                                                                                                                                                using System;
                                                                                                                                                using System.Collections.Generic;
                                                                                                                                                using System.ComponentModel;
                                                                                                                                                using System.Data;
                                                                                                                                                using System.Drawing;
                                                                                                                                                using System.Runtime.InteropServices;
                                                                                                                                                using System.Text;
                                                                                                                                                using System.Windows.Forms;

                                                                                                                                                namespace cleaning
                                                                                                                                                {
                                                                                                                                                    public partial class Main : Form
                                                                                                                                                    {
                                                                                                                                                        Cleaning

                                                                                                                                                        Main.cs

                                                                                                                                                        using System;
                                                                                                                                                        using System.Collections.Generic;
                                                                                                                                                        using System.ComponentModel;
                                                                                                                                                        using System.Data;
                                                                                                                                                        using System.Drawing;
                                                                                                                                                        using System.Runtime.InteropServices;
                                                                                                                                                        using System.Text;
                                                                                                                                                        using System.Windows.Forms;

                                                                                                                                                        namespace cleaning
                                                                                                                                                        {
                                                                                                                                                            public partial class Main : Form
                                                                                                                                                            {
                                                                                                                                                                Cleaning

                                                                                                                                                                Main.cs

                                                                                                                                                                using System;
                                                                                                                                                                using System.Collections.Generic;
                                                                                                                                                                using System.ComponentModel;
                                                                                                                                                                using System.Data;
                                                                                                                                                                using System.Drawing;
                                                                                                                                                                using System.Runtime.InteropServices;
                                                                                                                                                                using System.Text;
                                                                                                                                                                using System.Windows.Forms;

                                                                                                                                                                namespace cleaning
                                                                                                                                                                {
                                                                                                                                                                    public partial class Main : Form
                                                                                                                                                                    {
                                                                                                                                                                        Cleaning

                                                                                                                                                                        Main.cs

                                                                                                                                                                        using System;
                                                                                                                                                                        using System.Collections.Generic;
                                                                                                                                                                        using System.ComponentModel;
                                                                                                                                                                        using System.Data;
                                                                                                                                                                        using System.Drawing;
                                                                                                                                                                        using System.Runtime.InteropServices;
                                                                                                                                                                        using System.Text;
                                                                                                                                                                        using System.Windows.Forms;

                                                                                                                                                                        namespace cleaning
                                                                                                                                                                        {
                                                                                                                                                                            public partial class Main : Form
                                                                                                                                                                            {
                                                                                                                                                                                Cleaning

                                                                                                                                                                                Main.cs

                                                                                                                                                                                using System;
                                                                                                                                                                                using System.Collections.Generic;
                                                                                                                                                                                using System.ComponentModel;
                                                                                                                                                                                using System.Data;
                                                                                                                                                                                using System.Drawing;
                                                                                                                                                                                using System.Runtime.InteropServices;
                                                                                                                                                                                using System.Text;
                                                                                                                                                                                using System.Windows.Forms;

                                                                                                                                                                                namespace cleaning
                                                                                                                                                                                {
                                                                                                                                                                                    public partial class Main : Form
                                                                                                                                                                                    {
                                                                                                                                                                                        Cleaning

                                                                                                                                                                                        Main.cs

                                                                                                                                                                                        using System;
                                                                                                                                                                                        using System.Collections.Generic;
                                                                                                                                                                                        using System.ComponentModel;
                                                                                                                                                                                        using System.Data;
                                                                                                                                                                                        using System.Drawing;
                                                                                                                                                                                        using System.Runtime.InteropServices;
                                                                                                                                                                                        using System.Text;
                                                                                                                                                                                        using System.Windows.Forms;

                                                                                                                                                                                        namespace cleaning
                                                                                                                                            82
public Main()
{
    InitializeComponent();

    foreach (ProcessInfo proc in Program.ProcList)
        listBox1.Items.Add(proc.Name);

    if (Program.ViewOnly)
    {
        btnTerminate.Text = "Ok";
        btnTerminate.Enabled = true;
        label1.Text = "Processes open in this desktop:";
        return;
    }

    Timer t = new Timer();
    int time = 5;
    btnTerminate.Text = time.ToString();
    t.Tick += delegate
    {
        time--;

        if (time < 1)
        {
            btnTerminate.Text = "Terminate";
            btnTerminate.Enabled = true;
            t.Stop();
        }
        else
            btnTerminate.Text = time.ToString();
    };
    t.Interval = 1000;
    t.Start();

    WinAPI.FLASHWINFO fInfo = new WinAPI.FLASHWINFO();
    fInfo.cbSize = Convert.ToUInt32(Marshal.SizeOf(fInfo));
    fInfo.hwnd = this.Handle;
    fInfo.dwFlags = WinAPI.FLASHW_CAPTION;
    fInfo.uCount = 6;
    fInfo.dwTimeout = 50;

    WinAPI.FlashWindowEx(ref fInfo);    
}

private void btnTerminate_Click(object sender, EventArgs e)
{
    if (Program.ViewOnly)
    {
        this.Close();
    }
if (MessageBox.Show("Are you sure you would like to close these processes?", "Warning", MessageBoxButtons.YesNo, MessageBoxIcon.Warning) != System.Windows.Forms.DialogResult.Yes) return;

uint code;
foreach (ProcessInfo info in Program.ProcList)
{
    if (handle == null || handle == IntPtr.Zero)
    {
        MessageBox.Show("Could not open handle for " + info.Name + ", manually close these processes", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
        continue;
    }
    if (WinAPI.GetExitCodeProcess(handle, out code) && code == 259)
        WinAPI.TerminateProcess(handle, 1);
}

this.Close();
}
[STAThread]
static void Main(string[] args)
{
    if (args.Length != 1)
    {
        MessageBox.Show("failed to load program, ensure all files are avilable....");
        return;
    }

    ViewOnly = (args[0] == "-view");

    IntPtr Desktop = WinAPI.GetThreadDesktop(WinAPI.GetCurrentThreadId());

    IntPtr snapshot = IntPtr.Zero;
    snapshot = WinAPI.CreateToolhelp32Snapshot((uint)WinAPI.SnapshotFlags.Process | (uint)WinAPI.SnapshotFlags.Thread, 0);
    List<uint> Procs = new List<uint>();

    WinAPI.THREADENTRY32 proct = new WinAPI.THREADENTRY32();
    proct.dwSize = (UInt32)Marshal.SizeOf(typeof(WinAPI.THREADENTRY32));
    if (WinAPI.Thread32First(snapshot, ref proct))
    {
        do
        {
            if (WinAPI.GetThreadDesktop(proct.th32ThreadID) == Desktop)
            {
                bool flag = true;
                foreach (uint i in Procs) if (i == proct.th32OwnerProcessID) { flag = false; break; }
                if (flag) Procs.Add(proct.th32OwnerProcessID);
            }
        } while (WinAPI.Thread32Next(snapshot, ref proct));
    }

    WinAPI.PROCESSENTRY32 proc = new WinAPI.PROCESSENTRY32();
    proc.dwSize = (UInt32)Marshal.SizeOf(typeof(WinAPI.PROCESSENTRY32));
    if (WinAPI.Process32First(snapshot, ref proc))
    {
        do
        {
            bool flag = false;
            foreach (uint i in Procs) if (i == proc.th32ProcessID) { flag = true; break; }
            //TODO: Check if ctfmon.exe closes correctly, if not remove it from the filter
            // - It does not close correctly on Windows 8, check on Windows 7 later.
            if (flag)
            {
                string name = proc.szExeFile.ToLower();
            }
        } while (WinAPI.Process32Next(snapshot, ref proc));
    }
bool bautoterm = false;
foreach(string s in autoterm)
{
    if (name == s)
    {
        bautoterm = true;
        break;
    }
}

if (bautoterm)
else if (proc.th32ProcessID != currpid)
}
} while (WinAPI.Process32Next(snapshot, ref proc));

else
{
    throw new ApplicationException(string.Format("Failed with win32 error code {0}", Marshal.GetLastWin32Error()));
}

WinAPI.CloseHandle(snapshot);

if (ProcList.Count > 0)
{
    Application.EnableVisualStyles();
    Application.SetCompatibleTextRenderingDefault(false);
    Application.Run(new Main());
}
if (AutoProcList.Count > 0)
{
    uint code;
    foreach (ProcessInfo info in Program.AutoProcList)
    {
        IntPtr handle = WinAPI.OpenProcess(WinAPI.ProcessAccessFlags.Terminate |
        WinAPI.ProcessAccessFlags.QueryInformation, false, (int)info.ID);
        if (handle == null || handle == IntPtr.Zero)
        {
            MessageBox.Show("Could not open handle for " + info.Name + ", manually
close this process?", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
            continue;
        }
        if (WinAPI.GetExitCodeProcess(handle, out code) && code == 259)
            WinAPI.TerminateProcess(handle, 1);
    }
}
}
public class ProcessInfo
{
    public uint ID;
    public string Name;

    public ProcessInfo(uint ID, string Name)
    {
        this.ID = ID;
        this.Name = Name;
    }
}

WinAPI.cs

using System;
using System.Collections.Generic;
using System.Runtime.InteropServices;
using System.Text;

namespace cleaning
{
    public class WinAPI
    {
        [DllImport("kernel32.dll", SetLastError = true)]
        [return: MarshalAs(UnmanagedType.Bool)]
        public static extern bool GetExitCodeProcess(IntPtr hProcess, out uint lpExitCode);

        [DllImport("kernel32.dll", SetLastError = true)]
        [return: MarshalAs(UnmanagedType.Bool)]
        public static extern bool TerminateProcess(IntPtr hProcess, uint uExitCode);

        [DllImport("kernel32.dll")]
        public static extern IntPtr OpenProcess(
            ProcessAccessFlags processAccess,
            bool bInheritHandle,
            int processId
        );

        [Flags]
        public enum ProcessAccessFlags : uint
        {
            All = 0x001F0FFF,
            Terminate = 0x00000001,
            CreateThread = 0x00000002,
            VirtualMemoryOperation = 0x00000008,
            VirtualMemoryRead = 0x00000010,
            VirtualMemoryWrite = 0x00000020,
            DuplicateHandle = 0x00000040,
            CreateProcess = 0x000000080,
            CreateThread = 0x00000002,
            VirtualMemoryOperation = 0x00000008,
            VirtualMemoryRead = 0x00000010,
            VirtualMemoryWrite = 0x00000020,
            DuplicateHandle = 0x00000040,
            CreateProcess = 0x000000080,
            CreateThread = 0x00000002,
            VirtualMemoryOperation = 0x00000008,
            VirtualMemoryRead = 0x00000010,
            VirtualMemoryWrite = 0x00000020,
            DuplicateHandle = 0x00000040,
            CreateProcess = 0x000000080,
SetQuota = 0x00000100,
SetInformation = 0x00000200,
QueryInformation = 0x00000400,
QueryLimitedInformation = 0x00001000,
Synchronize = 0x00100000

[DllImport("user32.dll")]
[return: MarshalAs(UnmanagedType.Bool)]
public static extern bool FlashWindowEx(ref FLASHINFO pwfi);

public const uint FLASHW_CAPTION = 0x00000001;

[StructLayout(LayoutKind.Sequential)]
public struct FLASHINFO
{
    public uint cbSize;
    public IntPtr hwnd;
    public uint dwFlags;
    public uint uCount;
    public uint dwTimeout;
}

[Flags]
public enum SnapshotFlags : uint
{
    HeapList = 0x00000001,
    Process = 0x00000002,
    Thread = 0x00000004,
    Module = 0x00000008,
    Module32 = 0x00000010,
    Inherit = 0x80000000,
    All = 0x0000001F,
    NoHeaps = 0x40000000
}

//inner struct used only internally
[StructLayout(LayoutKind.Sequential, CharSet = CharSet.Auto)]
public struct PROCESSENTRY32
{
    const int MAX_PATH = 260;
    internal UInt32 dwSize;
    internal UInt32 cntUsage;
    internal UInt32 th32ProcessID;
    internal IntPtr th32DefaultHeapID;
    internal UInt32 th32ModuleID;
    internal UInt32 cntThreads;
    internal UInt32 th32ParentProcessID;
    internal Int32 pcPriClassBase;
    internal UInt32 dwFlags;
    [MarshalAs(UnmanagedType.ByValTStr, SizeConst = MAX_PATH)]
internal string szExeFile;

[StructLayout(LayoutKind.Sequential, CharSet = CharSet.Auto)]
public struct THREADENTRY32
{
    internal UInt32 dwSize;
    internal UInt32 cntUsage;
    internal UInt32 th32ThreadID;
    internal UInt32 th32OwnerProcessID;
    internal UInt32 tpBasePri;
    internal UInt32 tpDeltaPri;
    internal UInt32 dwFlags;
}

[DllImport("kernel32", SetLastError = true, CharSet = System.Runtime.InteropServices.CharSet.Auto)]
public static extern IntPtr CreateToolhelp32Snapshot([In]UInt32 dwFlags, [In]UInt32 th32ProcessID);

[DllImport("kernel32", SetLastError = true, CharSet = System.Runtime.InteropServices.CharSet.Auto)]
public static extern IntPtr GetThreadDesktop(uint dwThreadId);

[DllImport("kernel32", SetLastError = true, CharSet = System.Runtime.InteropServices.CharSet.Auto)]
public static extern IntPtr GetThreadDesktop(int dwThreadId);

[DllImport("kernel32.dll")]
public static extern int GetCurrentThreadId();
using System;
using System.Collections.Generic;
using System.Runtime.InteropServices;
using System.Text;

namespace cleaning
{
    class Program
    {
        [DllImport("kernel32.dll", SetLastError = true)]
        [return: MarshalAs(UnmanagedType.Bool)]
        static extern bool GetExitCodeProcess(IntPtr hProcess, out uint lpExitCode);

        [Flags]
        private enum SnapshotFlags : uint
        {
            HeapList = 0x00000001,
            Process = 0x00000002,
            Thread = 0x00000004,
            Module = 0x00000008,
            Module32 = 0x00000010,
            Inherit = 0x80000000,
            All = 0x0000001F,
            NoHeaps = 0x40000000
        }

        [StructLayout(LayoutKind.Sequential, CharSet = CharSet.Auto)]
        private struct PROCESSENTRY32
        {
            const int MAX_PATH = 260;
            internal UInt32 dwSize;
            internal UInt32 cntUsage;
            internal UInt32 th32ProcessID;
            internal IntPtr th32DefaultHeapID;
            internal UInt32 th32ModuleID;
            internal UInt32 cntThreads;
            internal UInt32 th32ParentProcessID;
            internal Int32 pcPriClassBase;
            internal UInt32 dwFlags;

            [MarshalAs(UnmanagedType.ByValTStr, SizeConst = MAX_PATH)]
            internal string szExeFile;
        }

        [StructLayout(LayoutKind.Sequential, CharSet = CharSet.Auto)]
        public struct THREADENTRY32
internal UInt32 dwSize;
internal UInt32 cntUsage;
internal UInt32 th32ThreadID;
internal UInt32 th32OwnerProcessID;
internal UInt32 tpBasePri;
internal UInt32 tpDeltaPri;
internal UInt32 dwFlags;
}

[DllImport("kernel32", SetLastError = true, CharSet = System.Runtime.InteropServices.CharSet.Auto)]
static extern IntPtr CreateToolhelp32Snapshot([In] UInt32 dwFlags, [In] UInt32 th32ProcessID);

[DllImport("kernel32", SetLastError = true, CharSet = System.Runtime.InteropServices.CharSet.Auto)]
static extern bool Process32First([In] IntPtr hSnapshot, ref PROCESSENTRY32 lppe);

[DllImport("kernel32", SetLastError = true, CharSet = System.Runtime.InteropServices.CharSet.Auto)]
static extern bool Process32Next([In] IntPtr hSnapshot, ref PROCESSENTRY32 lppe);

[DllImport("kernel32.dll")]
static extern bool Thread32First(IntPtr hSnapshot, ref THREADENTRY32 lpte);

[DllImport("kernel32.dll")]
static extern bool Thread32Next(IntPtr hSnapshot, ref THREADENTRY32 lpte);

[DllImport("kernel32", SetLastError = true)]
[return: MarshalAs(UnmanagedType.Bool)]
private static extern bool CloseHandle([In] IntPtr hObject);

[DllImport("user32.dll", SetLastError = true)]
public static extern IntPtr GetThreadDesktop(uint dwThreadId);

[DllImport("user32.dll")]
public static extern IntPtr GetThreadDesktop(int dwThreadId);

[DllImport("kernel32.dll")]
public static extern int GetCurrentThreadId();

static void Main(string[] args)
{
    Console.Write("Desktop handle (" + GetThreadDesktop(GetCurrentThreadId()) + ":
    ");
    IntPtr Desktop = (IntPtr)uint.Parse(Console.ReadLine());
    IntPtr snapshot = IntPtr.Zero;
}
snapshot = CreateToolhelp32Snapshot((uint)SnapshotFlags.Process | (uint)SnapshotFlags.Thread, 0);
List<uint> Procs = new List<uint>();

THREADENTRY32 proct = new THREADENTRY32();
proct.dwSize = (UInt32)Marshal.SizeOf(typeof(THREADENTRY32));
if (Thread32First(snapshot, ref proct))
{
    do
    {
        if (GetThreadDesktop(proct.th32ThreadID) == Desktop)
        {
            bool flag = true;
            foreach (uint i in Procs) if (i == proct.th32OwnerProcessID) { flag = false; break; }
            if (flag) Procs.Add(proct.th32OwnerProcessID);
        }
    } while (Thread32Next(snapshot, ref proct));
}

PROCESSENTRY32 proc = new PROCESSENTRY32();
proc.dwSize = (UInt32)Marshal.SizeOf(typeof(PROCESSENTRY32));
if (Process32First(snapshot, ref proc))
{
    do
    {
        bool flag = false;
        foreach (uint i in Procs) if (i == proc.th32ProcessID) { flag = true; break; }
        if (flag)
            Console.WriteLine("Proc id: " + proc.th32ProcessID + "\nProc name: " + proc.szExeFile);
    } while (Process32Next(snapshot, ref proc));
} else
    { throw new ApplicationException(string.Format("Failed with win32 error code {0}", Marshal.GetLastWin32Error())); }

CloseHandle(snapshot);

Console.ReadKey();

Reference:


