BSP Project Description:
< CAUID: Chip Against Unintended Information Disclosure >

Monday 6th May, 2024 - 14:17

Georgi Bozhkov
University of Luxembourg
Email: georgi.bozhkov.001@student.uni.lu

This report has been produced under the supervision of:
Bernard Steenis
University of Luxembourg
Email: bernard.steenis@uni.lu

Abstract—**As technology continues to develop and become more accessible, becoming a victim of a cyber attack become more and more probable. Advancements in the sphere of cybersecurity are continuously made, but in some instance that might not be enough and rather than relying on a software which works in conjunction with the internet to prevent malicious attempts of data theft, a more convenient and responsive method could be implemented against security penetration attempts. A computer chip named CAUID collaborates directly with the CPU in order to immediately stop any attacking attempts.**

1. Introduction

The story of cybersecurity can be traced back to 1962, where the MIT developed a system which dynamically changed the password of the computers within the institution, reason being, to limit the time of students on said devices, while also ensuring their privacy. However, with the implementation of the Internet, the need for protection against digital assault became a necessity.

A method of penetrating the security system of a computer or more commonly referred to as ”hacking”, is an act that allows the gaining of personal information without the consent of an individual. Despite numerous advancements made to various security systems, people with technological expertise and malicious intent frequently manage to get past the barrier-like layers and access confidential information. Additionally, it is inevitable for the occurrence of such unethical events to rise, due to the globalisation of technology over time.

Even though, as previously mentioned, cybercriminals are able to circumvent security measures, it is undeniably true that during the procedure of various accessing methods, specifically the ones that require coding expertise, some lines of code and programs are more frequently used than others. With this information, it is possible to design a computer chip that locally prevents users from performing the act in question, and CAUID (Chip Against Unintended Information Disclosure) is exactly that.

2. Project description

2.1. Domains

2.1.1. Scientific. The objective of this scientific deliverable is to conduct a thorough investigation of the distinctions between software-based antivirus (SBA) and hardware-based antivirus (HBA) systems. The goal is to demonstrate the advantages and disadvantages of each system, with the conclusion of showing why a hardware-based system efficient than a software-based one. The comparison will begin with presenting various distinctions between software and hardware-based antiviruses, mainly focusing on the advantages of the hardware-based system. Furthermore, a table that displays the differences between software and hardware antivirus systems will be made, with the goal of creating a simple visual depiction of both systems' advantages and disadvantages.

2.1.2. Technical. The goal of the technical deliverable is to proof a concept, which is that the implementation of a CAUID antivirus system is achievable. This will be accomplished as a result of completing two tasks.

The first task is to describe the functionality of CAUID. This will be done by delivering an explanation on what the purpose of CAUID is and where it is situated within a Computer hardware system. After all of the components, of which CAUID is comprised are defined, the end of the task will be to explain in detail how CAUID works and captures malicious attacks.

The second task contains the the proof of concept and will be done through the development of a program in the Python language, which will simulate the working of CAUID. The program will include algorithms, similar to those of a CPU, in order to create a believable scenario for the working of CAUID. The program will also include the code for CAUID itself, which will represent the behaviour of the security chip.

2.2. Targeted Deliverables

2.2.1. Scientific deliverables. The goal of this scientific deliverable is to address the scientific question of the BSP: ”Are hardware-based antivirus systems better than software-based ones?” The question will be answered by assessing differences between the two antivirus systems, describing their advantages and drawbacks and proving that the hardware-based system is superior.

2.2.2. Technical deliverables. The goal of the technical deliverable design part of the project is to provide an explanation of CAUID’s operation and afterwards to produce a visual depiction of the many scenarios in which CAUID and a simple CPU would interact. Python programming will be used to create the visualizer.

3. Pre-requisites

3.1. Scientific pre-requisites

From a scientific standpoint, before beginning with the BSP, one must have basic understanding operating systems, functionality of a CPU and how a computer virus functions.

3.2. Technical pre-requisites

From a technical standpoint, before beginning with the BSP, one must have an understanding of how a computer is structured on a physical basis, knowledge about how communication between parts of the computer works, assembly code, and intermediate Python programming skills.

4. Scientific Deliverable

Question: Are hardware-based antivirus systems better than software-based ones?

4.1. Requirements

The aim of this scientific deliverable is to theoretically compare hardware-based antivirus systems to their software ones. This comparison will involve evaluating various features of both types of antivirus systems and determining which exceeds the other based on certain criteria. The goal is to provide insights into the advantages of hardware-based solutions. At the conclusion of the scientific deliverable, a comprehensive table will be created to illustrate the differences between hardware-based and software-based antivirus systems in a simple and understandable manner. This table will serve as a visual aid for analyzing and interpreting the findings of the comparison, providing valuable insights into the superiority of hardware-based antivirus systems.

4.2. Design

4.2.1. Criteria. In order to justify the design of CAUID, we will compare the features of HBA and SBA systems; however, to do that correctly, it’s essential to establish specific criteria for evaluation. These criteria will serve as benchmarks to assess the capabilities and effectiveness of each type of antivirus solution.

Usability Usability in antivirus systems refers to the ease with which users can interact with and manage the software. It encompasses factors such as installation, configuration, navigation, and user interface design. A highly usable antivirus system is one that is straightforward to install and configure, requiring minimal technical expertise. The user interface should be intuitive, with clear navigation and easily accessible features. Users should be able to understand and utilize the various functionalities of the antivirus software without encountering unnecessary complexity. Additionally, usability includes the availability of comprehensive documentation and user support, ensuring that users can troubleshoot issues and utilize the software effectively. Ultimately, a usable antivirus system enhances the user experience and facilitates the efficient management of security measures.

Performance Performance refers to the utilization of system resources by the antivirus software during operation. It includes the consumption of CPU processing power, memory (RAM), disk input/output (I/O), and network bandwidth. Effective antivirus software should have minimal resource usage to avoid impacting the overall performance of the system. High usage can lead to slowdowns, system freezes, and decreased productivity. Therefore, antivirus solutions are evaluated based on their ability to maintain low resource usage while providing efficient protection against malware threats. Optimized usage ensures that the antivirus software runs smoothly in the background without significantly affecting the performance of other applications and tasks on the computer.

Effectiveness Effectiveness in antivirus systems refers to the ability of the software to accurately detect and neutralize threats. It should utilize multiple detection methods, including signature-based detection, heuristic analysis, behavior-based detection, and machine learning algorithms, to identify and block malicious files and activities. Additionally, the antivirus should have a low false positive rate, meaning it should correctly identify legitimate files and applications without flagging them as malware. Moreover, effectiveness also encompasses the speed and responsiveness of the antivirus system in detecting and responding to threats in real-time. A highly effective antivirus system provides timely protection, minimizing the risk of malware infection and ensuring the security of the computer system and its data.

4.3. Real-Time Detection

Real-time threat detection allows for the immediate identification and response to malicious activities. Software-based antivirus solutions continuously monitor system activities and incoming files, comparing them against a database of known malware signatures and employing heuristic analysis to detect suspicious behavior. While software-based solutions provide effective real-time protection, their reliance on the operating system can introduce delays in threat detection and response. On the other hand, hardware-based antivirus systems operate independently of the operating system, embedded directly into the CPU architecture. This integration enables hardware-based solutions to perform real-time threat detection with minimal latency, offering immediate identification and mitigation of threats. As a result, hardware-based antivirus systems are inherently superior in terms of real-time threat detection, providing faster and more efficient protection against malware.

4.4. System Resource Consumption

System resource consumption directly impacts the overall performance of the computer system. Software-based antivirus (SBA) systems typically require a significant amount of system resources, including CPU processing power, memory (RAM), and disk I/O, especially during scanning operations. This resource consumption can lead to system slowdowns, increased boot times, and reduced performance, particularly on older or less powerful computers. Additionally, frequent updates and background scanning processes further contribute to resource usage. In contrast, hardware-based antivirus (HBA) systems are integrated directly into the computer hardware. This setup generally has lower resource consumption compared to its software counterpart because it operates independently of the operating system and utilizes dedicated hardware resources for threat detection and mitigation. As a result, HBAs have minimal impact on system performance, allowing for smoother operation and an improved user experience, even during intensive tasks or high-demand scenarios.

4.5. Resistance to Malware Evasion

Malware creators continuously develop sophisticated techniques to evade detection; therefore, it is important for antivirus software to be able to apply countermeasures as fast as possible. SBA systems primarily rely on signature-based detection and heuristic analysis to identify malware patterns and behaviors. However, these methods can be easily bypassed by encrypted malware variants that alter their code to evade detection. Additionally, rootkit techniques and fileless malware can hide from traditional antivirus scanners by exploiting vulnerabilities in the operating system. In contrast, HBA systems offer enhanced resistance to malware evasion techniques. HBAs operate at a lower level of the system, monitoring CPU instructions and system behavior in real-time. This enables them to detect and block malware at the hardware level, making it more difficult for malware to evade detection.

Features	HBA	SBA
Real-Time Detection	Yes	Yes (Slower than HBA)
System Resource Consumption	Minimal	Noticable/Substantial
Resistance to Malware Evasion	String	Dependence on Updates