CiME: Hardware-Based Detection Ransomware Using Capacitor in Memory for Entropy

HyeLim Jung[1*] and Ki-Woong Park[2†]
[1]SysCoreLab., Sejong University, Seoul, 05006, Republic of Korea
[2] Department of Information Security, Sejong University, Seoul, 05006, Republic of Korea.
hyello13@gmail.com, woongbak@sejong.ac.kr

Abstract

Ransomware is a growing threat in the cybersecurity landscape, often encrypting user data at high speeds, bypassing traditional detection methods. Existing software-based entropy detection methods, though effective, face limitations due to delayed response times caused by high computational overhead and resource demands for entropy calculation, which hinders real-time performance. This paper introduces a novel hardware-based approach to ransomware detection by measuring entropy in real time within memory controllers, overcoming the resource overhead limitations of software-based methods. The proposed Capacitor in Memory for Entropy (CiME) system consists of two components: CiME-Q and CiME-R. CiME-Q measures entropy in the memory controller's Read/Write queues by monitoring the flow of binary data through the electric properties of capacitors, tracking charge levels in response to the data flow in real time. This system captures the entropy of data flow by observing the charged state of capacitors. In parallel, CiME-R assesses the entropy in specific regions of memory by analyzing the uniformity of charge across DRAM cells through capacitors' electric properties. This allows it to detect high-entropy operations, such as encryption or obfuscation processes within ransomware activity. By directly analyzing data randomness at the hardware level, CiME offers a faster and more efficient detection method that overcomes the resource overhead limitations of software-based approaches. This research demonstrates that the CiME system can perform entropy calculations in a low-computation hardware-based way, allowing it to detect changes in entropy as data flows, providing a real-time response to entropy variations associated with ransomware behavior.

Keyword: Entropy, Hardware-Based, Memory Analysis, Ransomware

[*] Masterminded EasyChair and created the first stable version of this document
[†] Created the first draft of this document

1 Introduction

Ransomware not only poses a threat to the victim's data assets, but the victim is also compelled to sacrifice their computer resources in an effort to detect it (OzH, Aris A, Levi A., Uluagac A. S., 2022). Ransomware generally encrypts data very quickly, and bypasses or disables existing security systems, making detection and defense difficult. For this reason, various methodologies have been proposed to quickly detect and block ransomware, and among them, entropy-based detection techniques are attracting attention (Urooj U, Al-rimy B. A. S., Zainal A, GhalebF. A., Rassam M. A., 2021) (Gopinath M. & Sethuraman S. C., 2023).

Entropy refers to a measure of randomness or unpredictability in data. Unencrypted data typically has a low entropy value, and its structure or patterns are relatively predictable. Encrypted data, on the other hand, has a high entropy value because it appears to be highly randomized. By leveraging this distinction, it becomes possible to identify encrypted data more effectively. A sudden increase in entropy during the encryption process can serve as a key indicator of ransomware activity, enabling timely detection and response.

Previous research on entropy-based detection primarily utilizes software to monitor data and analyze entropy changes continuously. When ransomware begins encrypting data in a file system or memory, the entropy value increases sharply. This method of real-time entropy monitoring has been widely adopted as a core technique for ransomware detection.

However, software-based entropy detection methods present several limitations. First, monitoring large volumes of data in real time can degrade system performance, as it requires significant computational resources. Second, given the high speed at which ransomware encrypts data, software-based methods may not respond quickly enough, leading to delayed detection. These limitations make software-based methods resource-intensive and less effective in real-time scenarios.

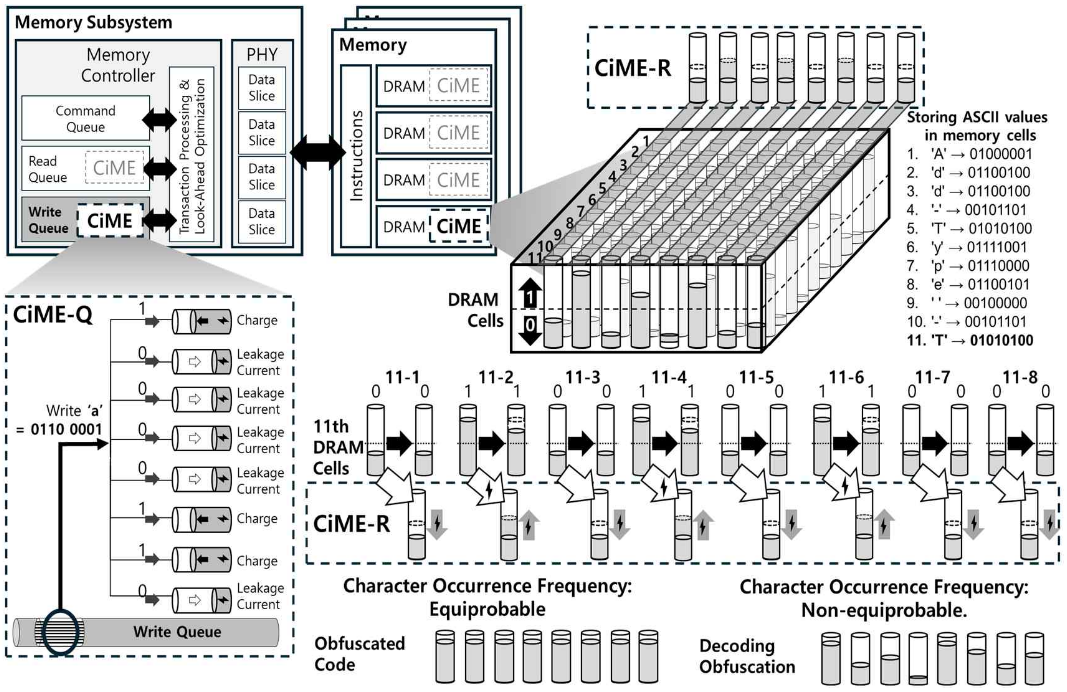

Figure 1: Overview of CiME(The Capacitor in Memory for Entropy)

To overcome these challenges, it is essential to explore hardware-based entropy measurement systems or more efficient detection techniques. Hardware-based entropy measurement offers a more accurate and faster alternative to software-based approaches, reducing the impact on system performance while improving the speed and accuracy of ransomware detection.

This paper proposes a novel method for real-time entropy measurement in hardware memory controllers. By leveraging hardware rather than software, we introduce a more efficient way to analyze data randomness. The Capacitor in Memory Controller for Entropy (*CiME*) system is introduced as a solution for real-time entropy analysis.

CiME measures entropy in two key areas of memory: *CiME-Q,* which monitors the Read/Write queues, and *CiME-R*, which focuses on the cells within DRAM. *CiME-Q* uses capacitors in the Read/Write queues of the memory controller to convert data into electrical signals and measure entropy in real-time. As shown in Figure 1, the memory controller contains queues through which data passes, including the write queue. *CiME* is integrated into the write queue to monitor the entropy of the data at the hardware level. In this system, data is converted into 8-bit binary form, with each bit recorded as a charge of 0 or 1 in the capacitors (JeongH. L., AhnS. K., BaekS. H., ParkK. W., 2019). The entropy is determined by analyzing the charge levels: plaintext data results in uneven charge distribution across certain capacitors, while encrypted data distributes the charge more evenly. This allows *CiME-Q* to measure the entropy of the data and detect encrypted files in real-time, making it possible to identify ransomware activity quickly. *CiME-R* takes a different approach by using capacitors in DRAM cells to measure entropy based on the charge stored in these cells. As shown in Figure 1, DRAM contains numerous cells where data is stored as electrical charges. There is a threshold for determining whether a cell holds a 0 or a 1, with the charge level needing to exceed this threshold to represent a 1. *CiME-R* draws a small amount of charge from each cell to measure entropy, ensuring that this process does not interfere with the stored data. The system consists of 8 capacitors, each connected to specific DRAM cells. By examining the uniformity or variation in the capacitor charge, *CiME-R* can assess entropy. This approach allows *CiME-R* to detect high-entropy operations, such as the encryption processes involved in ransomware attacks, where encrypted data increases entropy, and to recognize when entropy decreases as encrypted data is decoded. By monitoring the charge distribution, *CiME-R* provides valuable insights into the entropy changes that occur during these processes.

In this paper, we organize *CiME* into Q and R to view entropy from a macroscopic perspective and entropy from a microscopic perspective. *CiME-Q* allows for a macroscopic observation of the entropy of data loaded into memory. It can identify the abrupt changes in entropy within the queue that arise from ransomware encryption patterns, where large amounts of data are encrypted and loaded into memory. To examine memory entropy from a microscopic perspective, *CiME* applies *CiME-R* to DRAM. *CiME-R* is designed to detect entropy changes at the DRAM level, specifically to identify processes where ransomware loads keys into memory, uploads obfuscated code, and then decrypts it to expose the code. *CiME* enables low-computation entropy measurements with no resource overhead and can be performed quickly because *CiME* uses the natural electrical patterns of memory as data is read/written to measure entropy, rather than the traditional method. This opens up the possibility of measuring entropy during the encryption process of ransomware.

The remainder of this paper is organized as follows: Section II reviews prior research and discusses the flow of memory data and capacitors in ransomware attacks. Section III presents the *CiME* system architecture and the results of entropy measurement within this framework. Finally, Section IV concludes with a summary of findings and future research directions.

2 Related Works & Acknowledgements

2.1 Entropy Measure to Detecting Ransomware

Entropy-based detection techniques measure the randomness of data and have been used as an Entropy-based detection techniques have become a key method for measuring data randomness and detecting malicious threats, particularly ransomware. Entropy refers to the level of randomness or unpredictability within data, and encrypted files exhibit increased entropy due to their inherently random structure. Leveraging this property, various studies have explored methods to detect ransomware by monitoring changes in entropy during the encryption process. Most of these studies rely on software-based entropy measurement techniques, which analyze the data flowing through file systems or memory to identify increased randomness. However, these approaches often incur high processing costs and can degrade system performance.

A prominent example of this approach is ShieldFS, a self-healing file system designed to mitigate the damage caused by ransomware encryption (Continella, et al., 2016). ShieldFS monitors low-level I/O activities within the file system, profiling normal behavior and automatically restoring changes if abnormal activity is detected. The system analyzes factors such as high-entropy write operations, file read/write frequency, and file name changes to detect ransomware in real-time. Upon detecting ransomware, ShieldFS immediately restores the file system to minimize the impact of encryption.

Another study by Lee et al. combined entropy analysis with machine learning to detect ransomware-infected files (LeeK., LeeS.Y., YimK., 2019). This approach specifically addresses the issue of ransomware-infected files being synchronized to backup systems by analyzing entropy changes during the encryption process. Since encrypted files typically exhibit higher entropy, the machine learning model classifies files based on this trait to identify infections. The study demonstrated improved detection rates and reduced false positives compared to traditional methods.

The study by Wenbo Zhang et al. proposes a method for detecting malware, such as ransomware, by measuring entropy in memory (ZhangW., Li X., Zhu T., 2023). It utilizes the large language model LLaMA-7B in combination with memory forensics to analyze ransomware activity in real-time. By performing entropy analysis on memory snapshots, the study identifies high-entropy patterns, such as those caused by ransomware encryption, and presents a method to distinguish between legitimate software and ransomware. The research emphasizes the importance of entropy measurement in detecting complex ransomware activities, such as data exfiltration, by visually analyzing the entropy changes in data stored in memory.

This study by Igor Korkin et al. proposes a method for detecting zero-day malware in memory dumps by combining statistical analysis techniques with GPU acceleration (KorkinIgor & NesterovIvan, 2016). Using CUDA-enabled GPU hardware, this approach accelerates entropy analysis through parallel processing, quickly detecting abnormal entropy changes within memory dumps. In this study, entropy analysis measures data unpredictability to differentiate between malware and benign data.

Thus, various techniques utilizing entropy for ransomware detection have been developed, ranging from software-based to machine learning and hardware-based entropy measurement methods. However, existing methods increase computational processes and consume system resources to detect ransomware entropy. Software-based entropy measurement studies have shown effective results in ransomware detection, yet they face limitations due to high computational resource consumption and challenges in real-time detection. Software methods are often time-consuming when processing large data flows, leading to potential detection delays when ransomware quickly encrypts data. Additionally, the high computational cost burdens system performance, highlighting the need for more efficient, real-time entropy measurement methods. This need arises from the complexity of measuring entropy and analyzing ransomware patterns.

The method proposed in this study reduces resource usage and computation time in the entropy measurement process and suggests a hardware-based entropy measurement technique through further research in pattern analysis. This method measures entropy directly within memory, addressing bottlenecks in software-based approaches and providing a faster solution for ransomware detection.

2.2 Memory Data Flow of Ransomware

When a system is infected with ransomware, the process of reading data from the file system or memory, encrypting it, and saving it back is repeatedly executed. In this process, the memory controller and the physical layer (PHY) play critical roles, and the flow of data occurs at high speed and complexity. The memory access pattern of ransomware can be divided into three stages: the Command Queue, the Read Queue, and the Write Queue. Each stage provides an opportunity to detect the distinctive behavior of ransomware.

In the Command Queue, ransomware issues multiple commands to initiate the encryption process, primarily through read and write operations. These command patterns differ from normal system behavior, often showing repetitive or irregular activity accompanied by abrupt changes. The Read Queue is responsible for retrieving a large volume of data from memory in preparation for encryption, which may lead to a backlog of read commands. In the Write Queue, the encrypted data is written back to memory, with an unusually high number of write commands compared to regular programs. The encrypted data at this stage exhibits complex patterns, increasing the randomness and, thus, the entropy of the data. To optimize memory access, ransomware employs techniques such as transaction processing and look-ahead optimization to efficiently handle large-scale write operations. Data is written in parallel using a method called data slicing, allowing for the rapid encryption of substantial amounts of data.

Additionally, ransomware often employs obfuscation techniques to make its code complex and unpredictable in order to avoid detection (NaeemH., DongS., FalanaO. J., UllahF., 2023). Obfuscated code typically exhibits random patterns, resulting in high entropy (MuralidharanT., CohenA., GersonN., NissimN., 2022). When obfuscated ransomware code is loaded into memory, it temporarily increases the entropy of the data stored in DRAM. Once loaded, the ransomware waits until certain conditions are met before decoding or descrambling the obfuscated code to restore it to its original form for execution. After the decoding process is complete, the obfuscated code is transformed into executable code, which generally has a more structured and patterned format, often with repetitive or regular bit sequences. Compared to the obfuscated code, the decoded version has lower randomness, and as a result, the entropy in DRAM decreases when the decoded code is stored in memory.

Ransomware's encryption and obfuscation processes significantly increase the randomness of data, and this can be detected by monitoring changes in entropy. The shifts in command patterns in the Command Queue, the recording of encrypted data in the Write Queue, the parallel processing in the Data Slice stage, and the entropy variations in DRAM cells all contribute to entropy changes. These sharp increases in entropy can serve as a crucial indicator for detecting ransomware activity in real time.

2.3 Capacitor

Capacitors are essential elements in electronic circuits, known for their capacity to store and release electrical charge. They perform a crucial role across diverse hardware systems, enabling key functions such as data storage, power stabilization, and signal filtering. Capacitors are essential in memory systems, where they are responsible for managing data storage and retrieval. This study explores how capacitors can be employed to analyze the electrical behavior of data flow.

A capacitor's primary function is to accumulate and release charge. When a voltage is applied, charge is stored on its plates, effectively holding energy until it is needed by the circuit. When the external circuit demands it, this stored charge is released, allowing current to flow. The capacitor's charge state is highly sensitive to changes in voltage, making it an ideal component for storing electrical signals or data. Its ability to store and release charge in extremely short periods makes capacitors especially suited for memory systems that require fast response times.

In hardware memory systems, particularly Dynamic Random-Access Memory (DRAM), capacitors play an indispensable role. Data in DRAM is stored in bit cells, each composed of one transistor and one capacitor. The capacitor is the core component responsible for data storage: when charge is present, the bit is stored as "1," and when charge is absent, it is stored as "0."

Capacitors also offer significant advantages in real-time data analysis. Their ability to quickly accumulate and release charge enables rapid data flow analysis without placing heavy demands on system performance. Due to their compact size, capacitors can easily be integrated into hardware structures, such as data queues within memory controllers. However, optimizing capacitor size and performance is critical; if the capacitance is too small, it may fail to store sufficient charge for accurate data measurement, whereas too large a capacitance can result in slower response times. Achieving the right balance is essential for ensuring system efficiency and reliability.

3 CiME Mechanism

The Capacitor in Memory for Entropy (*CiME*) system is an approach that integrates capacitors into the read/write queues of a memory controller and data of DRAM cells to monitor data in real time and measure entropy. This method enables the detection of encryption activities, such as those initiated by ransomware, by evaluating the randomness of data through electrical signals, all while maintaining the existing structure of memory systems.

3.1 CiME Structure

The *CiME* system introduces a hardware-based approach for measuring entropy in memory to detect encrypted data or ransomware activity. It does this through two main components*: CiME-Q and CiME-R,* which monitor data changes by tracking entropy levels. Figure 1 provides a detailed explanation of how these two systems function.

CiME-Q measures entropy in the Read/Write queue of the memory controller. The system begins entropy measurement the moment data passes through the Write queue. As shown in the Figure 1, the memory controller includes both Command and Write Queues, with *CiME* positioned in the Write Queue. As data flows through, it is converted into an 8-bit format, with each bit stored in capacitors. These capacitors hold different charges based on whether the bit is 0 or 1: when the bit is 1, the capacitor charges, and when it's 0, the capacitor remains uncharged, potentially losing some charge due to leakage.

The core functionality of *CiME-Q* is to analyze these bit-charging patterns to measure entropy. In the case of plaintext data, certain bits receive more charge, leading to an uneven charging pattern. However, encrypted data results in uniform capacitor charging, indicating higher entropy. This allows *CiME-Q* to analyze data in real-time, detecting encrypted data or ransomware activity. When ransomware encrypts data, the bit charging pattern becomes more consistent, causing a sharp rise in entropy, which *CiME-Q* detects to identify the attack.

CiME-R operates at the DRAM cell level to measure entropy. Each DRAM cell stores data, and whether a cell holds a 0 or 1 is determined by the amount of charge stored. *CiME-R* reads the charge in each DRAM cell to determine the bit value. The key advantage is that *CiME-R* can detect even very small amounts of current, allowing it to measure entropy without disturbing the state of the DRAM cells.

CiME-R assesses entropy by examining the uniformity of charge across the DRAM cells. As shown in Figure 1, capacitors attached to each DRAM cell perform this role. When data is encrypted or obfuscated, all cells receive similar levels of charge, increasing entropy. When the obfuscation is reversed, such as when the data is decoded or descrambled, the charge levels become uneven again, reducing entropy. Through this process, *CiME-R* can determine in real time whether the data stored in memory is plaintext, encrypted, obfuscated, or decoded. This allows *CiME-R* to detect both encryption processes and the reversal of obfuscation as they happen in memory.

When *CiME-Q* detects a large increase in entropy, *CiME-R,* configured for each DRAM, senses entropy at the DRAM cell level. Here, it can distinguish between cases where all DRAM cells show high entropy and cases where only certain cells do. If entropy is high across all cells, it likely indicates large-scale data encryption. If only certain cells exhibit high entropy, it could indicate frequent data changes within a specific memory region, potentially due to obfuscation.

When *CiME-Q* detects a gradual increase in entropy over time, *CiME-R* checks the entropy of individual regions to determine whether any specific area shows a change. If a high-entropy pattern appears in specific cells, it suggests that ransomware is performing incremental encryption, rather than bulk encryption. Alternatively, obfuscation may be occurring; this can be confirmed through a sharp drop in entropy when obfuscated code in memory is decrypted and exposed.

However, these entropy fluctuations are not exclusive to ransomware and may also occur during normal operations such as compression, deep learning processes, or media tasks, all of which can increase entropy. Therefore, a more precise entropy analysis is required. The *CiME* system, as proposed in this paper, demonstrates that it can measure entropy with minimal resource overhead. Future research will focus on analyzing memory entropy patterns over time to determine if they can be identified without resource overhead and used to detect specific patterns.

3.2 Capacitor Calculator

CiME has eight capacitors attached to it, and we built a capacitor calculator to see if we could measure entropy with eight capacitors. This calculator models the behavior of the capacitor, specifically addressing charging and leakage currents within the circuit.

The formula used in the calculation is as follows:

$$Q_t = Q_{prev} \cdot e^{\frac{-t}{R \cdot C}} + C \cdot V \cdot \left(1 - e^{\frac{-t}{R \cdot C}}\right) - (I_{leakage})$$

Q_t represents the charge stored in the capacitor at time t. *C* is the capacitance (measured in farads). *V* is the applied voltage (measured in volts). *R* is the circuit resistance (measured in ohms). *t* refers to the elapsed time (measured in seconds). e is the natural constant (approximately 2.71828). Q_{prev} represents the previously stored charge in the capacitor. $I_{leakage}$ is the leakage current from the capacitor.

This formula models the charging process and accounts for the leakage current in the capacitor, considering variables such as capacitance, voltage, resistance, and time. To accurately calculate the capacitor's charge in response to a given voltage input, the voltage change over time must be factored in. By calculating the capacitor’s voltage at each time step, the flow of charge into the capacitor can be effectively monitored and analyzed.

3.3 Implementation

This section presents a set of three charts from Figure 2 that visually compare how capacitor charge and standard deviation(STDDEV) differ between plaintext and ciphertext, illustrating the distinct electrical properties exhibited by encrypted and unencrypted data within a memory system.

The first chart shows the variation in capacitor charge, represented as STDDEV, for both plaintext and ciphertext data. The y-axis reflects the STDDEV values, while the x-axis tracks data flow. For plaintext data, the STDDEV is relatively high, indicating significant variability in capacitor charge. This suggests that plaintext data, which has more predictable patterns and lower randomness, causes uneven charging. In contrast, the STDDEV for ciphertext is much lower, reflecting a more uniform

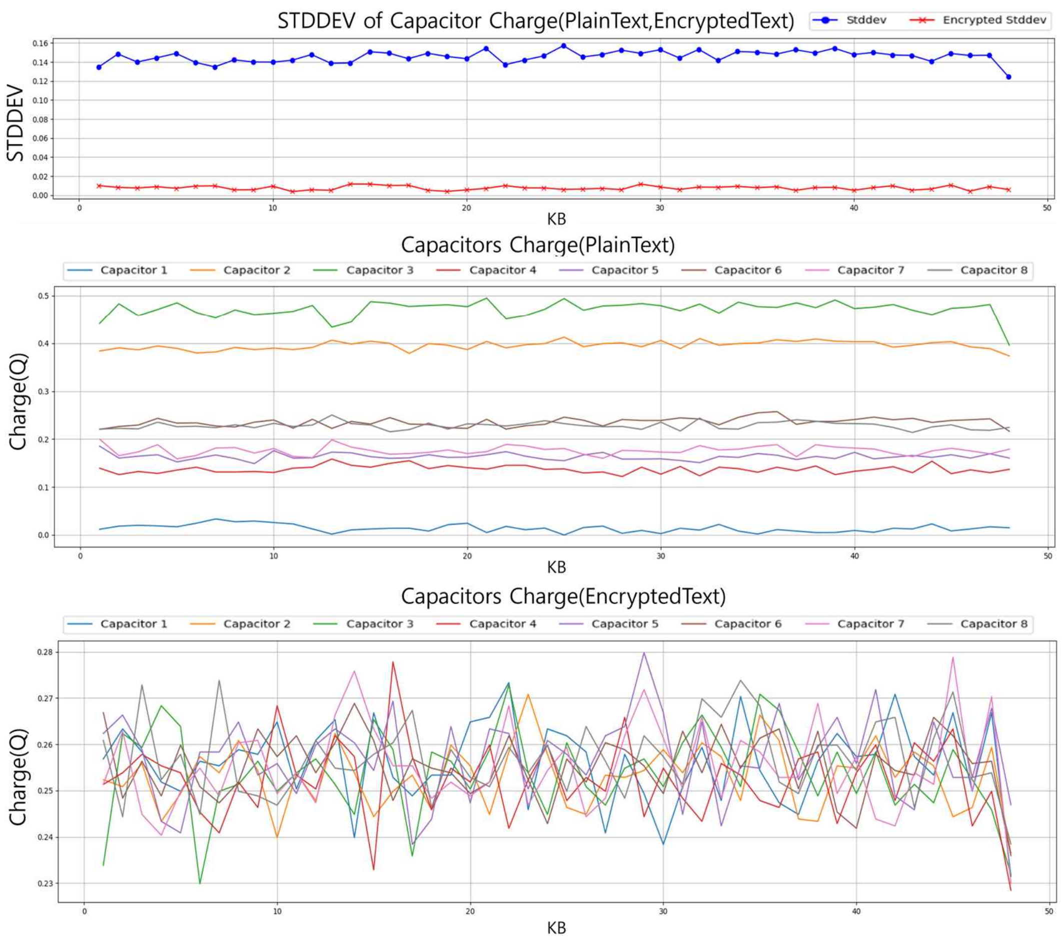

Figure 2: STDDEV and charge graph of 8 capacitors for entropy measurement by plain and encrypted texts

charge distribution across capacitors, which signifies higher randomness. This distinction arises because encrypted data disperses charge more evenly, demonstrating its higher entropy.

The second chart illustrates how capacitors charge during the processing of plaintext data. The y-axis represents the amount of charge stored in the capacitors, while the x-axis denotes data flow. In this case, Capacitors 1 and 4 show lower charge levels, whereas Capacitors 2 and 3 accumulate relatively higher charges. This pattern suggests that recurring structures in plaintext data cause certain capacitors to store more charge than others, leading to an imbalanced charge distribution. This uneven distribution is indicative of the lower entropy associated with plaintext.

The third chart shows the changes in capacitor charge during the processing of encrypted data. The y-axis displays the charge levels, while the x-axis represents data flow. For ciphertext, the charge is distributed much more evenly across all capacitors, with minimal variation. This uniform charge distribution is characteristic of the high randomness in encrypted data. Unlike plaintext, the variation in charge levels between capacitors is minimal, highlighting the significantly higher entropy of ciphertext.

CiME offers an effective hardware-based solution for detecting encryption activities by analyzing the electrical properties of data flow within the memory controller and DRAM cells. Plaintext data,

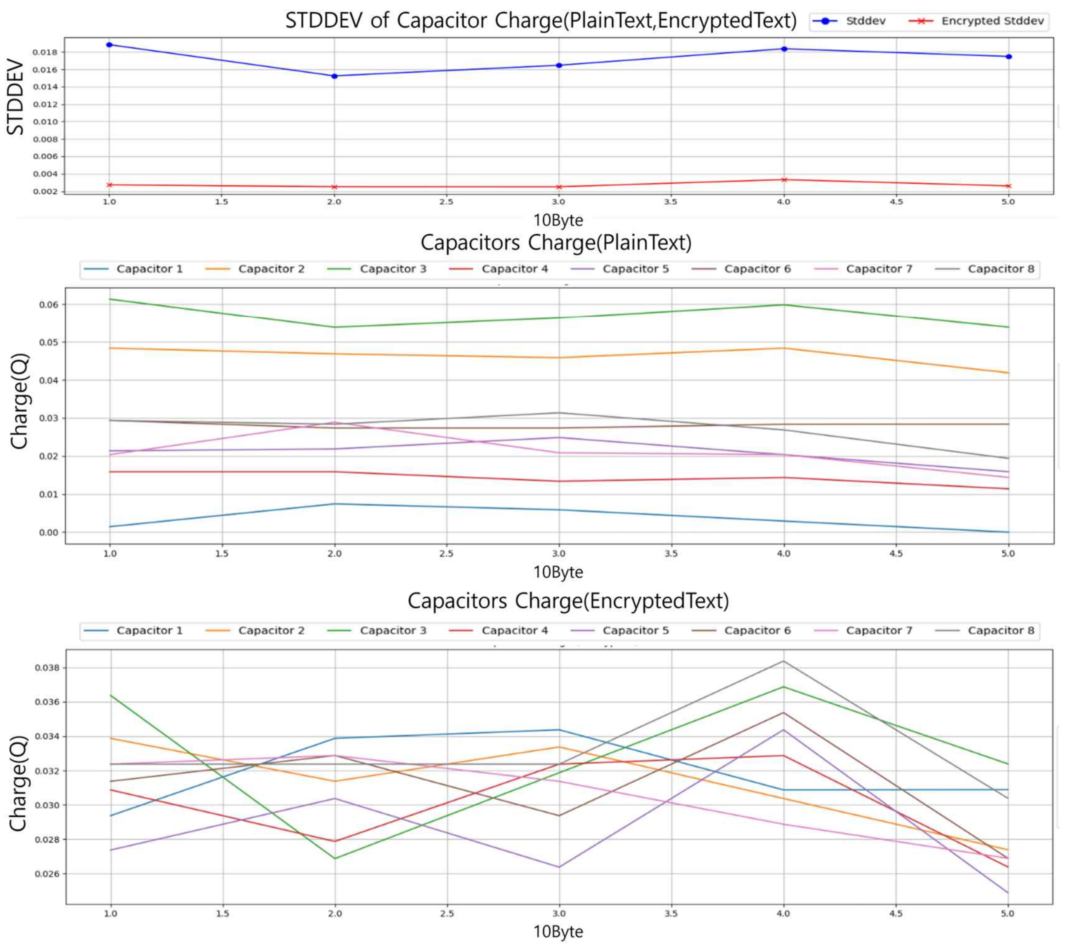

Figure 3: STDDEV and charge graph of 8 capacitors for entropy measurement by plain and encrypted texts(50B)

with its lower randomness, results in more variation in capacitor charge, while encrypted data, with its higher randomness, distributes charge more evenly across capacitors. This characteristic allows the *CiME* system to detect high-entropy data in real time as it enters memory. Ransomware operates by encrypting data, which inherently increases the entropy compared to plaintext. The *CiME* (Capacitor in Memory for Entropy) system can detect these changes in entropy in real-time, allowing it to identify when encrypted data is being transmitted. This capability enables early detection of ransomware activity, triggering security alerts or initiating immediate countermeasures to block the attack.

The *CiME* system proposed in this paper measures entropy in memory, with *CiME-Q* observing the overall entropy flow of data loaded into memory and *CiME-R* monitoring the entropy flow for each specific region of memory. In our experiments, *CiME* measured data sizes ranging from the minimum detectable size of 50 bytes to 2GB and 16GB. RAM sizes vary, with capacities such as 512 bytes and up to 2GB.

As shown in Figure 3, we could observe entropy results even with the minimum size of 50 bytes. Even with a data size of 50 bytes, our *CiME* was able to measure entropy changes according to the data flow. Furthermore, as shown in the graph in Figure 4, *CiME* was also able to detect changes in entropy based on data flow when executing text containing a mix of plain text and encrypted text. A decrease

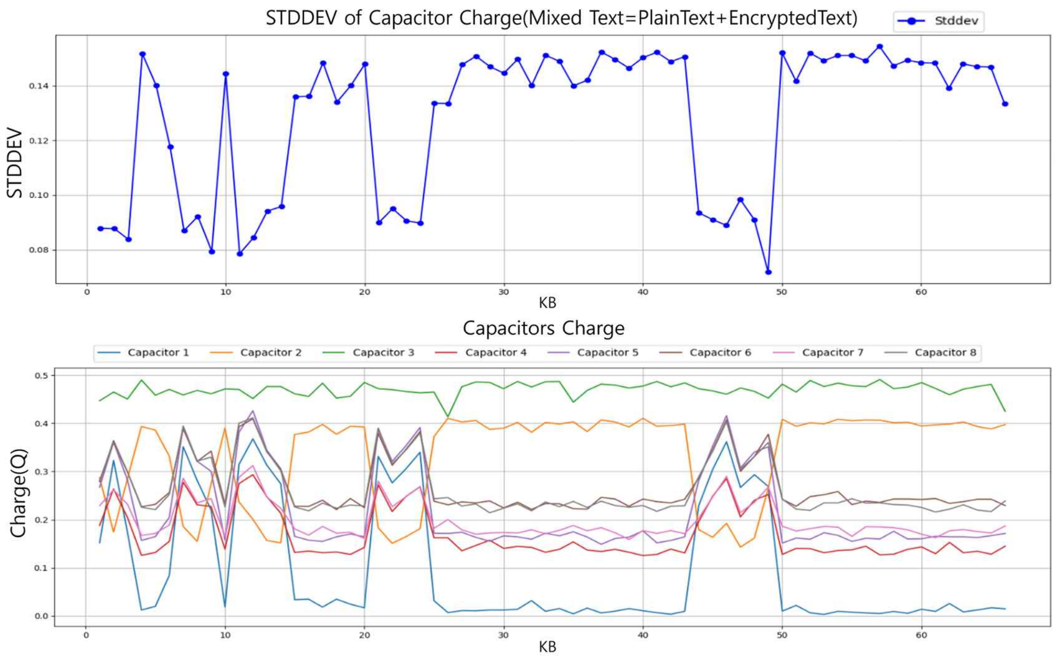

Figure 4: STDDEV and Charge graph of 8 capacitors for entropy measurement by Mixed Text(60KB)

in deviation can be interpreted as a state where high-entropy data has been injected into memory. This demonstrates that changes in entropy can be assessed within memory.

Traditional software-based entropy measurement methods are limited by high computational costs and difficulties with real-time processing. *CiME*, by contrast, enhances efficiency by directly measuring entropy within the memory controller and DRAM cells, thereby reducing performance overhead. By incorporating eight capacitors into the memory queue to monitor data flow, *CiME* strengthens security while minimizing its impact on overall system performance. Additionally, *CiME* successfully identified entropy characteristics with low computational overhead through hardware. However, since various calculations can influence entropy, a detailed pattern analysis is necessary to confirm whether the observed entropy is caused by ransomware. Various techniques have been previously researched for this level of entropy analysis, and studies have shown that memory entropy analysis is effective for ransomware detection. However, these existing methods often require substantial resources, like machine learning, to measure memory entropy. *CiME* proposes a resource-efficient method for measuring memory entropy by attaching capacitors, a hardware component, to memory to naturally monitor electrical flow. By analyzing these entropy patterns, we expect to detect ransomware-specific entropy patterns.

4 Conclusion

The Capacitor in Memory for Entropy (*CiME*) system offers a powerful hardware-based solution for detecting ransomware through real-time entropy measurement, while also reducing the computational resource overhead typically associated with software-based methods. By incorporating capacitors into the memory controller's Read/Write queues and DRAM cells, *CiME* can efficiently monitor entropy changes that signal encryption processes. *CiME-Q* tracks the charge distribution within memory queues, while *CiME-R* measures entropy within DRAM cells, detecting high-entropy operations such as ransomware encryption. This method addresses critical challenges in traditional detection systems by reducing the performance overhead and computational resource demands, ensuring faster detection without compromising system efficiency. The *CiME* system demonstrates significant potential in enhancing ransomware detection capabilities with minimal resource consumption, making it an ideal solution for real-time, high-performance cybersecurity applications. Future research should focus on optimizing the system further and expanding its use to broader threat detection scenarios.

Acknowledgements

This work was supported by the Ministry of Science and ICT (MSIT), South Korea, through the Institute of Information & Communications Technology Planning & Evaluation (IITP) under the International Collaborative Research Program (Project No. RS-2022-00165794, 20%), the National Research Foundation of Korea (NRF) under the Mid-career Researcher Program (Project No. RS-2023-00208460, 20%), the Defense ICT Convergence Research Program (Project No. 2022-11220701, 20%), the Core Technology Development Program for Realistic Content (Project No. RS-2023-00228996, 20%), and the Core Technology Development Program for Information Security (Project No. RS-2024-00438551, 20%).

References

Continella, A., Guagnelli, A., Zingaro, G., De Pasquale, G., Barenghi, A., Zanero, S., & Maggi, F. (2016). Shieldfs: a self-healing, ransomware-aware filesystem. *Proceedings of the 32nd annual conference on computer security applications* (pp. 336-347). ACM.

Gopinath , M., & Sethuraman , S. (2023). A comprehensive survey on deep learning based malware detection techniques. *Computer Science Review. 47*, p. 100529. Elsevier.

Jeong, H., Ahn, S., Baek, S., & Park, K. (2019). Anomaly Detection Technology Using Potential Difference Displacement Detection of Data Bus. *J. Internet Serv. Inf. Secur. 9(4)*, pp. 68-77. ISSN.

Korkin, I., & Nesterov, I. (2016). *Acceleration of Statistical Detection of Zero-day Malware in the Memory Dump Using CUDA-enabled GPU Hardware.* arXiv.

Lee, K., Lee, S., & Yim, K. (2019). Machine learning based file entropy analysis for ransomware detection in backup systems. *IEEE access. 7*, pp. 110205-110215. IEEE.

Muralidharan, T., Cohen, A., Gerson, N., & Nissim, N. (2022). File packing from the malware perspective: Techniques, analysis approaches, and directions for enhancements. *ACM Computing Surveys. 55(5)*, pp. 1-45. ACM.

Naeem, H., Dong, S., Falana, O., & Ullah, F. (2023). Development of a deep stacked ensemble with process based volatile memory forensics for platform independent malware detection and classification. *Expert Systems with Applications. 223*, p. 119952. Elsevier.

Oz, H., Aris , A., Levi , A., & Uluagac , A. (2022). A survey on ransomware: Evolution, taxonomy, and defense solutions. *ACM Computing Surveys (CSUR). 54(11s)*, pp. 1-37. ACM.

Urooj , U., Al-rimy , B., Zainal , A., Ghaleb, F., & Rassam , M. (2021). Ransomware detection using the dynamic analysis and machine learning: A survey and research directions. *Applied Sciences. 12(1)*, p. 172. MDPI.

Zhang, W., Li , X., & Zhu , T. (2023). Entropy and memory forensics in ransomware analysis: Utilizing llama-7b for advanced pattern recognition. *Authorea Preprints.* Authorea.