WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Analysing Randomness of Encrypted Data to Reevaluate the Futility of its Compression

[1]Divya K S, [2]Gokul. S. Unnikrishnan, [3]Kavya Gopal

[1]Assistant Professor, Department of Computer Science(UG), Kristu Jayanti College, Bengaluru
[2,3]Student, Department of Physical Sciences, Kristu Jayanti College, Bengaluru
[1]divyaks@kristujayanti.com,[2] gokul.unnikrishnan@gmail.com, [3]kavyagopal295@gmail.com

ABSTRACT

This paper explores the commonly cautioned futility of attempting to compress encrypted data by investigating the level of randomness present in such data. The term "randomness" in this context refers to the extent to which data can be compressed effectively. The objective is to demonstrate that encrypted data exhibits statistical randomness, at least within the specified compression margins. Additionally, the paper examines the limitations of dictionary compression in this scenario. The argument is extended to propose a paradigm shift in perceiving encrypted data as pseudo-random, providing the rationale behind this perspective. In conclusion, a new compressor model is suggested, equipped with the insights gained from this study.

1. Introduction

In today's digital era, where information flows through digital channels, ensuring data security is of utmost importance. Cryptographic techniques have evolved from being specialized methods to becoming the standard for securing information transmitted across vast digital networks. However, as encryption becomes the norm, there's a growing need to optimize data transmission by reducing the volume of encrypted data sent over the network. Data compression emerges as a promising solution, aiming to streamline information transmission while maintaining its integrity.

Data compression involves the clever transformation of binary streams, usually represented as ASCII or similar formats, into more condensed forms without sacrificing information integrity. However, a major hurdle arises: encrypted data, by its very nature, is resistant to efficient compression due to its inherent randomness. This concept arises from the fundamental principle that randomness enhances the resilience of data against cracking attempts, serving as the foundation of modern cryptography algorithms[5].Keeping the above principle in mind, this paper attempts to measure the randomness that is claimed to be present in such data, and further analyse the trends observed and investigate its implications in data compression.

2. Methodology

2.1 Compression Margin

This passage introduces a new measure for evaluating text compression effectiveness: the **compression margin** (**C_M**). This metric differs from the traditional **compression ratio** (**C_R**), defined as the size reduction between uncompressed and compressed data streams. While C_R reflects the achieved size reduction, the proposed C_M establishes a **minimum compression threshold** for deeming compressed data "adequately compressed."

Formally, C_R is defined as:

$$C_R = \frac{\textit{Compressed size}}{\textit{Uncompressed size}}$$

Higher C_R values indicate greater size reduction, suggesting more efficient compression. In this context, surpassing the established C_M signifies **achievement of adequate compression**. Consequently, C_M serves as a benchmark for algorithm performance in minimizing file sizes.

The paper exclusively investigates **lossless text compression algorithms**. To establish a meaningful reference point for the compression margin (C_M), a comparative analysis of various compression algorithm classes is taken, as outlined in [1]. This analysis yields an average compression ratio (C_R) of 67.1195%. Consequently, for simplicity and clarity, C_M is subsequently defined as 67%.

This metric complements the traditional C_R by establishing a benchmark for assessing compressed text streams based on efficiency and conformity to industry standards set by earlier studies.

2.2 Statistical Randomness

Defining Randomness:

This section establishes a precise understanding of "randomness" within the context of encrypted data. Probability theory defines randomness as a state where **every possible selection** of elements from a set has an **equal probability** of occurring. This aligns with the principle of **uniformity in probability** [8], stating that without new information, no specific subset holds an advantage. In essence, randomness implies the **absence of bias** towards any particular outcome during selection, independent of long-term frequencies or subjective interpretations.

Data Generation and Encryption Process:

To analyze the innate properties of the encryption process, this study employs the Python module "faker" to generate realistic test data with diverse attributes. This tool facilitates the creation of realistic data points like **name, address, email, account number, transaction ID, product code, clause, IP address, device ID, and system log**. Notably, this approach avoids introducing bias stemming from specific plain-text content, ensuring the focus solely remains on studying the encryption process itself [7].

Algorithm 1 Generating Random Test Data

```
function GenerateData(targetFields)
    Initialize an empty data structure data
    for all field ∈ targetFields do
        Generate random data value for field field using appropriate methods
        Add value to data with key field
    end for
    return data
end function
```

Factors Affecting Output Variations:

Output variations in the encryption process originate from two primary factors: **changes in encryption methods** and **alterations in the plain-text**. Unlike the encryption key, the plain-text does not directly influence the core encryption process. Consequently, it is kept constant to isolate and analyse the specific characteristics and effects of the encryption process itself.

Encryption Algorithm Details:

The data undergoes a two-step encryption process, utilizing both the **RSA (Rivest-Shamir-Adleman)** and **AES (Advanced Encryption Standard)** algorithms [9]. RSA, an asymmetric encryption method, generates a public key for encryption and a private key for decryption. The public key is shared, while the private key remains confidential, ensuring secure communication. AES, a symmetric encryption method, serves as the data encryption layer. Initially, an RSA key pair is generated, along with an AES key. This AES key is subsequently secured by encrypting it with the RSA public key using the PKCS1_OAEP padding. Finally, the encrypted AES key is used to encrypt the data using the AES algorithm. As a further step, the binary output of the encryption in converted to human-readable format using the **base85 encoding scheme** [13].

Algorithm 2 Hybrid Encryption Algorithm with RSA and AES

```
function HybridEncrypt($message$)
    Generate an RSA keypair ($private\_key, public\_key$) using $RSA.KeyGen$
    $encrypted\_key \leftarrow$ EncryptKey($AES.KeyGen(), private\_key$)    ▷ Encrypt AES key with RSA
    $encrypted\_message \leftarrow$ EncryptData($message, AES.Decrypt(encrypted\_key, private\_key)$)    ▷ Decrypt key, then encrypt message with AES
    return $public\_key, encrypted\_key, encrypted\_message$
end function
function EncryptKey($key, private\_key$)
    Output: RSA-encrypted $key$ using appropriate padding scheme
end function
function EncryptData($data, key$)
    Output: AES-encrypted $data$ with a suitable mode and IV
end function
```

Hybrid Encryption Algorithm with Iterative Testing:

This study employs a hybrid encryption approach combining RSA and AES algorithms. The process iterates 1,000 times, generating 1,000 distinct encryption keys for each of the 10 chosen attributes. This approach allows for analysing variations induced by encryption across numerous key generations. The focus is to examine how **plain-text character count influences relative entropy**. The entire procedure is repeated 3 times (per attribute), resulting in 30,000 data points. It is noteworthy that as character count increases, the text's uniformity progressively decreases with each iteration

Algorithm 3 Main Script for Data Analysis and Encryption

```
Import necessary modules and libraries
Define constants and directories
if Main script is executed then
    $start\_time \leftarrow$ GetCurrentTime
    for each $target\_field$ in $TARGET\_FIELDS$ do
        Start processing $target\_field$
        for $i$ in range($div\_num$) do
            Prepare for iteration
            $nsum \leftarrow num\_samples//div\_num$
            $target\_message \leftarrow$ GenerateRandomData[$target\_field$]
            Initialize lists for storing analysis results
            for _in range($nsum$) do
                Generate RSA key pair for encryption
                Encrypt AES key using RSA
                Calculate character frequency probabilities
                Compute entropy metrics
                Log completion of iteration
                Update analysis lists
            end for
            Save analysis data to files
            Analyze entropy statistics
        end for
    end for
    Calculate elapsed time
    Log script execution time
end if
```

Simulating Uniformity and Computing Entropy:

The entropy of a collection of unique characters is computed in order to model a uniform probability distribution. [12]. This signifies a scenario where all unique characters have equal occurrence probability. As established by Shannon's theorem, the number of bits necessary to represent information hinges on the encoded data.

Shannon's entropy is given as:

$$H(X) = -\sum_{i=1}^{M} p_i \log_2 p_i$$

where

$H(X)$ is the entropy of the random variable *X*.

p_i is the probability of the *i*-th outcome [10].

Specifically, **Shannon's entropy** serves as a quantitative measure of uncertainty or randomness within a given probability distribution. For a uniform distribution, the entropy attains its maximum value.

Evaluating Randomness through Relative Entropy:

The study assesses the randomness and unpredictability of encrypted data using **relative entropy**. Information theory defines entropy as a metric for quantifying a data stream's unpredictability.

Relative entropy, also known as **Kullback-Leibler (KL) divergence**, measures the excess of cross-entropy over true entropy [3]. Cross-entropy, utilized in information theory and statistics, measures the disparity between two probability distributions.

$$D_{KL}(P||Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}$$

When the predicted distribution (q) aligns with the true distribution (p), the cross-entropy equals the entropy. Conversely, if q diverges from p, the cross-entropy differs from the entropy [4].

Algorithm 4 Functions for Entropy Calculation

```
    function CalculateEntropy(text)
        Input: text                                        ▷ The input text for entropy calculation
        Output: entropy                                    ▷ The calculated entropy of the text
4:  end function

    function CalculateDifference(encrypted_message)
        Input: encrypted_message                           ▷ The encrypted message
8:      Output: difference, base85_difference              ▷ The difference in entropy
    end function
```

This test aims to discern the information content and complexity of the encrypted data by calculating entropy differences between the encrypted string and a uniform distribution encompassing all unique characters. If the encrypted message's entropy approaches the maximum entropy of unique characters, it suggests a high degree of unpredictability resembling pseudo-randomness throughout encryption. Conversely, a significant increase in relative entropy implies that the encrypted data exhibits structure and reduced randomness.

Results

Once all the data has been collected, analysing it:

Attribute(per 1000 iterations)		$\mu_{\text{Encrypted}}$	μ_{Unique}	μ_{RE}	σ_{RE}
Name	It. Set 1	4.90	5.00	0.10	0.03
	It. Set 2	4.90	5.00	0.10	0.03
	It. Set 3	4.91	5.00	0.10	0.03
Address	It. Set 1	5.54	5.70	0.16	0.03
	It. Set 2	5.54	5.70	0.16	0.03
	It. Set 3	5.55	5.70	0.16	0.03
Email	It. Set 1	5.30	5.43	0.13	0.03
	It. Set 2	5.30	5.43	0.13	0.03
	It. Set 3	5.29	5.43	0.13	0.03
Account Number	It. Set 1	4.90	5.00	0.10	0.03
	It. Set 2	4.90	5.00	0.10	0.03
	It. Set 3	5.30	5.43	0.13	0.03
Transaction ID	It. Set 1	5.54	5.70	0.16	0.03
	It. Set 2	5.54	5.70	0.16	0.03
	It. Set 3	5.54	5.70	0.16	0.03
Product Code	It. Set 1	4.90	5.00	0.10	0.03
	It. Set 2	4.90	5.00	0.10	0.03
	It. Set 3	4.91	5.00	0.10	0.03
Clause	It. Set 1	5.91	6.10	0.20	0.03
	It. Set 2	5.71	5.88	0.17	0.03
	It. Set 3	5.91	6.10	0.20	0.03
IP Address	It. Set 1	4.90	5.00	0.10	0.03
	It. Set 2	4.90	5.00	0.10	0.03
	It. Set 3	4.90	5.00	0.10	0.03
Device ID	It. Set 1	5.30	5.43	0.13	0.03
	It. Set 2	5.30	5.43	0.13	0.03
	It. Set 3	5.30	5.43	0.13	0.03
System Log	It. Set 1	5.54	5.70	0.16	0.03
	It. Set 2	5.54	5.70	0.16	0.03
	It. Set 3	5.30	5.43	0.13	0.03

Table 1: Relative Entropy Table of Selected Attributes

Attribute	$\mu_{\text{Encrypted}}$	μ_{Unique}	μ_{RE}	σ_{RE}
Name	4.903	5.00	0.10	0.03
Address	5.543	5.70	0.16	0.03
Email	5.296	5.43	0.13	0.03
Account Number	5.034	5.143	0.166	0.03
Transaction ID	5.54	5.70	0.16	0.03
Product Code	4.903	5.00	0.10	0.03
Clause	5.843	6.026	0.19	0.03
IP Address	4.90	5.00	0.10	0.03
Device ID	5.30	5.43	0.13	0.03
System Log	5.46	5.609	0.15	0.03

Table 2: Simplified Relative Entropy Table of Selected Attributes

Table 1 summarizes the trends observed in relative entropy across various attributes over three sets of 1000 iterations. The attributes chosen are commonly encountered in information security and cryptography.

The analysis reveals several key findings:

- **Consistent Mean Entropy:** The mean entropy of encrypted strings and simulated uniform distributions remains consistent across attributes within each iteration set, suggesting uniform entropy levels.

- **Stable Mean Relative Entropy**: Mean relative entropy values exhibit stability across iteration sets for most attributes, indicating **consistent divergence between encrypted strings and uniform distributions**. This stability highlights the reliability and reproducibility of the measurements. **The mean across all μ_{RE} is 0.13**, which is the same as the **median 0.13**, and close by to the **mode 0.10**.

- **Uniform Standard Deviation of Relative Entropy**: The standard deviation of relative entropy shows **consistent values across iteration sets for each attribute**, reflecting the inherent variability in divergence between distributions across iterations. The relatively small value of 0.03, shows a modest **15-30% deviation from the mean.**

In conclusion, the table offers valuable insights into the behaviour of relative entropy metrics across various attributes and iteration sets. These findings provide implications for understanding the distributional characteristics and patterns within encrypted data samples.

Looking further upon the distributions themselves,

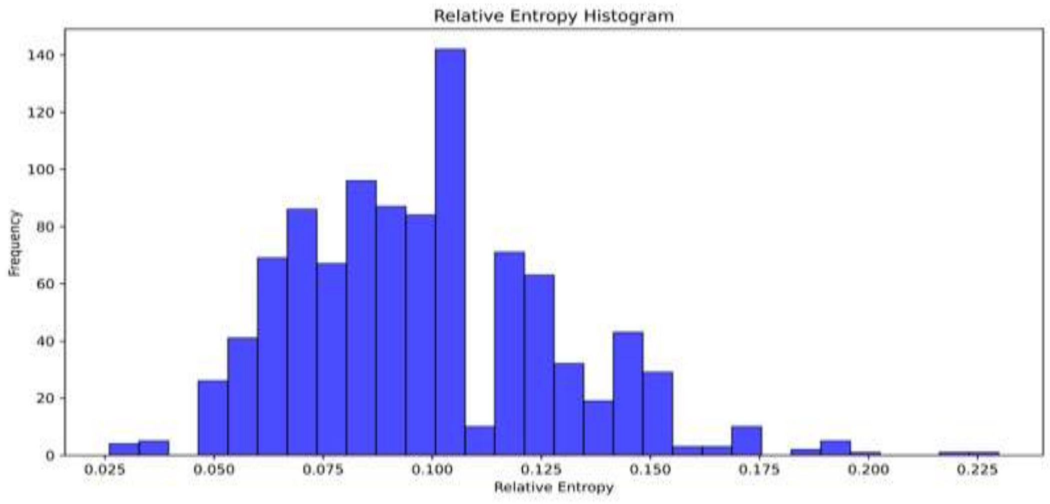

Figure 1: Relative Entropy distribution of Name: Plain-Text Iteration 1

Figure 1 depicts the frequency distribution of entropy values associated with a generated name. The relative entropy histogram reveals a **striking surge in frequency within the 0.10 - 0.105 range**, approximately 40% of all total observations.

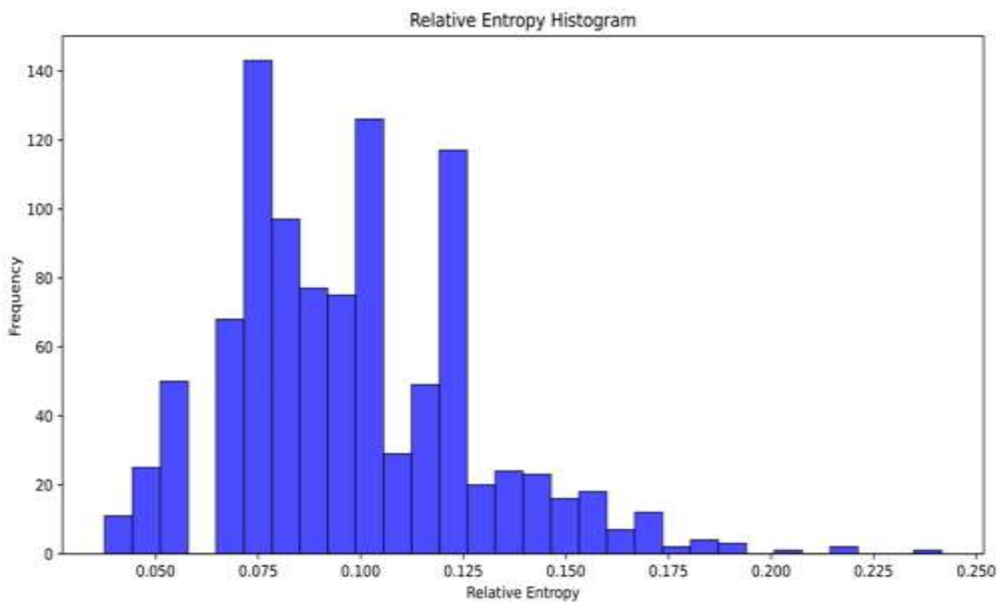

Figure 2: Relative Entropy distribution of Name: Plain-Text Iteration 2

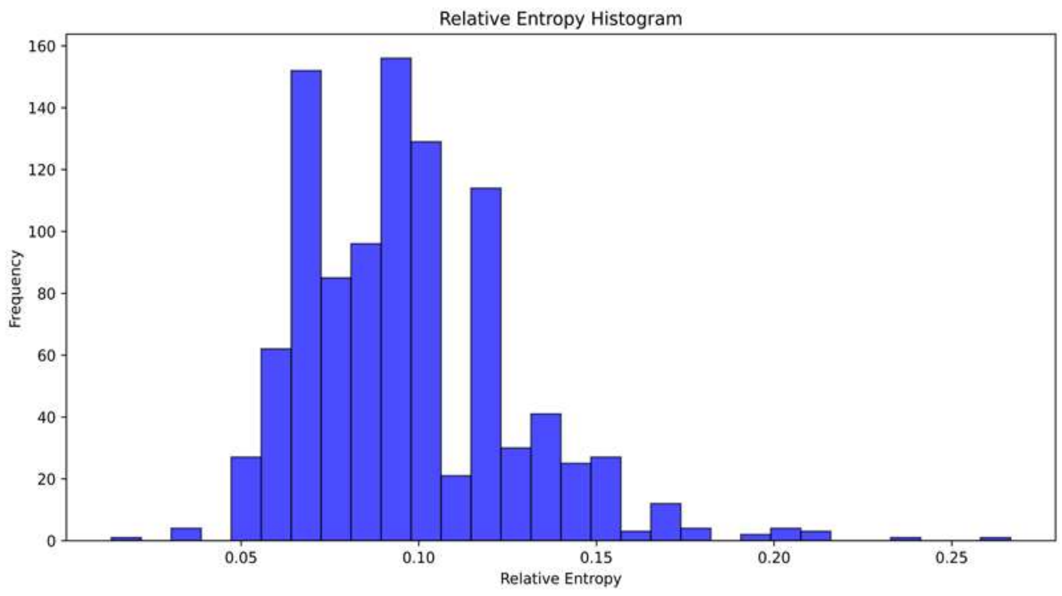

Figure 3: Relative Entropy distribution of Name: Plain-Text Iteration 3

Figures 2 and 3, similar to Figure 1 exhibit histograms with pronounced dominant peak(s). This consistent presence of dominant peak(s) across multiple iterations and figures suggests a **predominant distribution pattern of characters** within the generated names. Such characteristics are observed across all graphs, which are left out for the sake of brevity.

3. Discussion

3.1 Dictionary Compression

Encrypted data, as established above, exhibits statistical randomness, resulting in minimal inherent redundancy. This poses a significant challenge for further compression using dictionary-based methods, raising a crucial question: despite the demonstrably high redundancy within the original plain-text, why does dictionary compression fail to achieve significant compression ratios for encrypted data?

The answer lies in the very principle of non-determinism, a cornerstone of robust encryption. This principle ensures that identical plain-text characters encrypted with the same key produce distinct cipher-texts. This seemingly paradoxical behaviour arises from the deliberate incorporation of cryptographic randomness within the encryption process. While some algorithms might appear deterministic on the surface, this randomness injects an element of unpredictability, guaranteeing that even identical inputs (both plain-text and key) produce disparate and unpredictable outputs (cipher-text) [11].

3.2 Pseudo-randomness

An occurrence may seem random if there's inadequate data or processing power to make predictions about it. An event's degree of randomness is relative to the information and computing resources at our disposal. This is where the idea of pseudo-randomness comes in. Pseudo-randomness refers to the property of something appearing random, but being generated by a deterministic process. In other words, it is not truly random but behaves as if it were[6]

An effective encryption algorithm generates an encryption key that produces a pseudo-random sequence with finite randomness, derived from its own seed and a deterministic process. The output of the process should look completely random to the observer. In other words, the encrypted result ought to be indistinguishable from a truly random sequence of identical length.

4. Conclusion

Based on the patterns found in the data analysis, it is evident that encrypted data exhibits discernible patterns of statistical randomness, which influences the compressibility of the data. It suggests that, even though encryption makes the data unreadable in the absence of right decryption algorithm, it doesn't remove the structure of the data or the predictability of characters in the data. The analysis shows the presence of statistical unpredictability within encrypted data. This poses a challenge for compression algorithms, which strive to identify patterns within it. Since encryption obscures these patterns, applying these patterns directly to encrypted data often leads to minimal compression ratios.

Analysing the existing compression techniques, in particular dictionary compression, the paper concludes that encrypted data should be viewed as pseudorandom rather than considering it as completely patternless. Expanding on these observations, a new compressor model is proposed that is designed to effectively compress encrypted data considering its inherent randomness. The compressor recognises the intrinsic entropy within the encrypted data which may not be visible apparently.

An adaptive modelling method, coupled with a modified arithmetic encoder, produces compressed and encrypted output using a key. [2] The modified arithmetic decoder deciphers data similar to the encoder model, utilizing the encoded data and the key. This model presents a substantial model, unique by its structural flexibility. The research provides a practical way to introduce variability and unpredictability into the model's structure allowing it for compression while preserving statistical harmony to the fullest extent possible. The proposed encoder provides compression of between 67.5% and 70.5% for standard text files, with encryption reducing zeroth-order compression by roughly 6%. The model aims to achieve efficient compression while maintaining the security of the encrypted data and helps in quantifying the level of randomness, examining the statistical randomness and developing compression algorithms specially for the encrypted data.

Integrating compression and encryption together in a synergistic manner, it enhances the security of the data and also improves the data management and transmission efficiency. By optimizing the performances of existing compression algorithms for encrypted datasets, the research brings up methods for secure data storage and transmission.

5. References

[1]Haroon Altarawneh and Mohammad Altarawneh. Data compression techniques on text files: A comparison study. *International Journal of Computer Applications*, 26(5):42–54, 2011.

[2]R. Bose and S. Pathak. A novel compression and encryption scheme using variable model arithmetic coding and coupled chaotic system. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 53(4):848–857, 2006.

[3]Divya K.S,Dr.RoopaShree H.R,Dr.Yogesh A.C, "Non-Repudiation-based Network Security System using Multiparty Computation ", International Journal of Advanced Computer Science and Applications, Vol. 13, No. 3, 2022,PP 282- 289,DOI:10.14569/IJACSA.2022.0130335

[4]Min Chen and Mateu Sbert. On the upper bound of the kullback-leibler divergence and cross entropy. *arXiv preprint arXiv:1911.08334*, 2019.

[5]Fabio De Gaspari, Dorjan Hitaj, Giulio Pagnotta, Lorenzo De Carli, and Luigi V Mancini. Reliable detection of compressed and encrypted data. *Neural Computing and Applications*, 34(22):20379–20393, 2022.

[6]Oded Goldreich. Pseudorandomness. *Notices of the American Mathematical Society*, 46, 1999.

[7]Matea Ignatoski, Jonatan Lerga, Ljubiša Stanković, and Miloš Daković. Comparison of entropy and dictionary based text compression in english, german, french, italian, czech, hungarian, finnish, and croatian. *Mathematics*, 8(7):1059, 2020.

[8] Divya K.S,Dr.RoopaShree H.R,Dr.Yogesh A.C, "Framework for Optimizing Multiparty Computation for Secure Authentication in Internet-of-Things" Europian Chemical Bulletin, Eur. Chem. Bull. 2023, 12(Issue 8),3327-3335

[9]Nasrin Khanezaei and Zurina Mohd Hanapi. A framework based on rsa and aes encryption algorithms for cloud computing services. In *2014 IEEE conference on systems, process and control (ICSPC 2014)*, pages 58–62. IEEE, 2014.

[10]Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, 37(1):145–151, 1991.

[11] Divya K.S, Roopashree H.R, Yogeesh A.C, "Framework of Multiparty Computation for Higher Non-

Repudiation in Internet-of-Things (IoT)", International Journal of Computer Networks and Applications, vol.10, Iss.1, pp.84-94, 2023.DOI: 10.22247/ijcna/2023/218513

[12] Yuheng Bu, Shaofeng Zou, Yingbin Liang, and Venugopal V Veeravalli. Estimation of kl divergence: Optimal minimax rate. *IEEE Transactions on Information Theory*, 64(4):2648–2674, 2018.

[13] Maurice G Kendall and B Babington Smith. Randomness and random sampling numbers. *Journal of the royal Statistical Society*, 101(1):147–166, 1938.