CEaaS: Constrained Encryption-as-a-Service in Fog-Enabled IoT

Pallav Kumar Deb, *Graduate Student Member, IEEE*, Anandarup Mukherjee, *Graduate Student Member, IEEE*, and Sudip Misra, *Senior Member, IEEE*

Abstract—In this work, we present a solution towards facilitating dynamic encryption schemes – Constrained Encryption-as-a-Service (CEaaS) – in fog-enabled IoT environments. CEaaS is a two-level Fuzzy Inference System (FIS) in the fog layer which offers customized encryption algorithm decisions to the IoT user devices based on the current configuration, data size, and network state. Fog nodes use the two-tier FIS system to determine the category of the requesting IoT device at the first level and then the encryption scheme at the second level. Existing research on encryption focuses on developing new lightweight algorithms as a global solution for all devices without considering the heterogeneity and corresponding communication links. The device and network configurations collectively add operational delays, which elevates time and security threats. Under such circumstances, a solution that considers both the conditions (varying) for determining the appropriate encryption scheme and the key is important. Through extensive implementation and deployment of heterogeneous fog nodes, we observe that CEaaS is feasible for both powerful and resource-constrained IoT user devices with CPU and memory usage as low as 0.24% and 0.9%, respectively. CEaaS also incurs delays in the range of 0.7 seconds and energy consumption of 0.07 Joules while securing data transmission. With the feasibility of CEaaS, the dynamic encryption schemes ensure secure communications irrespective of the device types in a fog-enabled IoT environment.

Index Terms—Internet of Things, Industry 4.0, Fuzzy Inference Systems, dynamic encryption, security, fog computing

1 Introduction

With the advent of IoT, manufacturers, and service providers have been deploying devices, which are relatively small in size and Resource-Constrained (RC) in terms of *computational capability*, *storage*, and *battery*. With external platforms like cloud and fog computing, these IoT devices overcome operational limitations and deliver tasks that involve complex computations, enabling services over the network. These services include but are not limited to, industrial applications, personal healthcare, transportation, education, agriculture, and other businesses. To summarize, the IoT environments are highly dynamic and heterogeneous, consuming significant communication resources, leading to congestion and packet drops in the network. Further, these devices need to exchange sensitive data, which mandates the need for encryption routines. Although numerous encryption schemes exist, IoT devices usually do not have the necessary hardware for executing them in real-time, which induces undesirable delays. Towards this, most of the current research work on encryption focus on designing lightweight encryption algorithms [1]. However, there is no unified solution that addresses the heterogeneity of these devices and the network conditions for providing dynamic encryption decisions as a service.

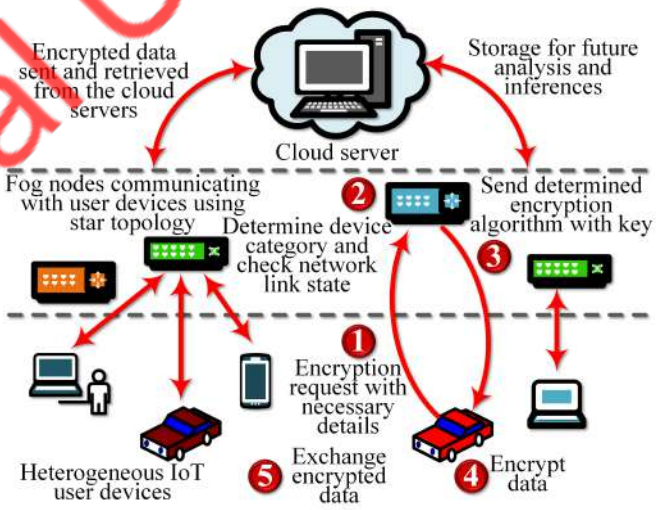


Figure 1: Network architecture and flow of information

In this work, we propose Constrained Encryption-as-a-Service (CEaaS), as an easy-to-deploy scheme for dynamically securing data from a vast range of IoT devices through the fog layer. We consider the heterogeneity of the devices in terms of its *category*, *computational capability*, *storage*, and the available *network quality* for deciding the appropriate encryption schemes in real-time. As shown in Fig. 1, the IoT devices employing CEaaS notify the Fog Nodes (FNs) regarding an impending data transfer. These request packets contain the current state of the IoT user devices (available CPU clock cycle, RAM, data type, and data size) based on which, the FNs first decide the category of the devices. Then, depending on the category, network condition, and data type, the FNs determine an encryption algorithm and the corresponding key for further communication. To avoid

P. K. Deb, A. Mukherjee, and S. Misra are with the Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, India. e-mail: (pallv.deb,sudipm)@iitkgp.ac.in, anandarupmukherjee@ieee.org

binary assessments of the mentioned parameters, we rely on Fuzzy Inference Systems (FIS) in realizing CEaaS. Without loss of generality, such dynamic selection of encryption algorithms by accounting for the device and network conditions makes CEaaS suitable for deployment on applications such as agriculture, healthcare, and other wireless sensor networks that involve low-powered IoT devices.

1.1 Motivation

IoT devices usually consist of RC devices in diverse domains and handling critical information from sensors based on which the local processing units make inferences and decisions. They also send an essential portion of the data to remote cloud servers for storage. Such data are prone to malicious attacks by adversaries, and decisions from tampered data have catastrophic consequences. Existing research usually addresses this issue by considering limited IoT device parameters and propose new lightweight encryption algorithms as a global solution. Encrypting the data at the FN is also a possible solution. However, the data remains open to attacks in the last mile link from adversaries, which leads to the reception of faulty data at the FN [2]. In robust environments like IoT, a global solution may not be beneficial due to the vast diversity of devices. A need for a common platform that dynamically decides the appropriate encryption scheme concerning the device category and network link state is advantageous. Since fog computing is a promising solution for IoT towards significantly reducing operational latencies, we propose and implement CEaaS as an easy-to-deploy robust encryption scheme in the fog layer. CEaaS chooses appropriate encryption algorithms and the corresponding keys for the IoT devices by exploiting the FNs in real-time. In this work, we assume that the IoT devices have data that needs secure transmission over a wireless network and that the access points for these devices are fog-enabled base stations. Since parameters such as CPU clock cycles ($\mathcal{C}$), RAM ($\mathcal{R}$), bandwidth ($\mathcal{B}$), data size ($\mathcal{D}_{sz}$), and others cannot be categorized precisely, we rely on a fuzzy inference system (FIS). In summary, we try to solve the equation, $\mathcal{P}_{ec} = \min \mathrm{g}(\mathcal{C}, \mathcal{R}. \mathcal{B}, \mathcal{D}_{sz})$, where $g()$ is the fuzzy function and $\mathcal{P}_{ec}$ is the final encryption scheme. Moreover, due to the relatively small size of the keys, compared to that of the transmission data, the proposed CEaaS scheme adds negligible load on the network.

1.2 Contribution

In this work, we address the issue of *dynamically* securing critical data for a vast range of *heterogeneous* RC IoT devices. We propose the selection of appropriate encryption schemes as a service for these devices via the *fog layer* in real-time. Since making binary assessments of link quality, device category, available resources is challenging, we adopt a FIS as our solution method. The following constitute our major *contributions* in this work:

- ***CEaaS:*** We present a novel scheme for aiding the selection of encryption schemes and keys as a service for heterogeneous RC IoT devices. Such a scheme gives flexibility to the devices to adjust according to their hardware and network conditions while securing last-mile transmissions with minimum overhead.
- ***Flexible Decisions:*** To avoid binary assessments of the devices and encryption algorithms, we develop a two-level FIS scheme. This allows CEaaS to accommodate a wide range of IoT devices and applications.
- ***Implementation:*** To show the feasibility of CEaaS, we implement the same on RC devices and discuss our observations based on primary parameters.

2 Related Work

In this section, we briefly discuss some of the recent efforts by researchers towards securing data for IoT environments. Although symmetric key encryption schemes are less secure than public-key schemes, researchers actively develop secure versions of the same as they are fast and suitable for streaming data in real-time. Some of these include the use of key-dependent substitution boxes that create block ciphers [3]. Most of these schemes use Elliptic Curve Cryptography (ECC) [4] for key generation. Devices in Industry 4.0 generate a huge amount of data. Lightweight chaos-based symmetric encryption schemes for such devices that generate ciphertexts of the same size are beneficial. Devices in industries also need to overcome communications over lossy channels. Solutions like appropriate key finder (AKF) [5] that chains block ciphers by assigning dedicated keys with the hash similar to some part of the payload messages is promising. Chaining block ciphers are not retrievable unless the previous blocks are decrypted correctly, which is suitable for lossy channels. The authors in [6] proposed a Transport Layer Security (TLS) protocol for securing the HTTP communication in industrial supply chains. On separate lines, blockchain may also be used for sharing data in a transparent and immutible manner such as in [7].

Researchers have also been working on securing data using various other security schemes. Access control based on group [8] being one of them as well as based on fine-grained attributes with the option of revoking permissions [9] being another. Encryption algorithms based on content are also a promising method of securing data. Gai *et al.* [10] proposed one such scheme where they encrypt data such that the private and personal details of the users are not visible to adversaries. The authors considered the computation cost of the devices for performing certain operations, size of data, and their corresponding privacy weights into their dynamic privacy protection (DPP) model. This DPP model is used to determine whether to implement a low or high-security scheme. Data aggregation in the fog layer incurs high computation and communication costs. Towards this, Saleem *et al.* [11] proposed FESDA, which is immune to false data injection attacks. Apart from these schemes, some approaches target securing data at the physical layer by using dynamic keys in OFDM encryption schemes [12] and also by incorporating randomized encryption scheme based on channel condition [13]. Boakye-Boateng *et al.* [14] proposed a one-time pad (OTP) based encryption scheme over wireless sensor networks (WSN) consisting of RC devices.

Synthesis: In the current literature, we observe that most approaches consider specific parameters and develop a global encryption solution, irrespective of device type. Moreover, they also propose encrypting the data from the IoT devices at the FN, which leads to unsecure data transmission in the last mile link, resulting in faulty data

reception at the FN. In this work, we acknowledge the heterogeneity of the IoT devices as well as the network conditions and determine appropriate encryption algorithms as a service. Such a decision service makes the IoT network robust. It heightens the flexibility of the devices by enabling data security without affecting performance. Additionally, as the size of the keys is relatively smaller than that of the transmission data, it adds negligible overhead on the network.

3 Network Architecture

Towards the implementation of CEaaS, we consider that the IoT devices connect to the network using fog-enabled access points/base stations. These devices have the capability of performing computations as well as networking operations simultaneously. As shown in Fig. 1, the IoT devices that need to transfer data first connect to an FN and awaits the determination of the encryption scheme for further communication. In the CEaaS scheme, we consider a set of FNs $F = \{f_1, f_2, ...f_p\}$, and a set of IoT devices $D = \{d_1, d_2, ..., d_q\}$, where p and q are the numbers of deployed units of each type. The IoT devices have critical data that needs to be sent securely over the network. We consider wire/wireless connections from these IoT devices to one of the nearby FNs. Each of the FNs may be connected to and receiving decision requests from multiple IoT devices (one-to-many). Once the encryption algorithm's decision and the key reach back to the IoT devices, it sends the ciphertext to the FNs or the destined devices. Upon requirement, the FN forwards the data to the cloud for storage. This scheme makes the CEaaS comprehend a star network/topology for operating. It may be noted that although star networks are prone to defects pertaining to a single point of failure, a huge deployment of FNs overcomes this issue as the user devices may choose a different access point/FN. In this work, we consider RC devices and FNs that exchange information in the IoT environment and categorize them into three types of devices – 1) Type 1, 2) Type 2, and Type 3. Type 1 devices are bare basic microcontrollers and microprocessors that do not have much computation capability and are relatively small in size like smart wearables, smart lights, and others. On the other hand, Type 2 devices are relatively rich (compared to Type 1) in computation capabilities and are not limited to single appliance functions like Alexa, smart home assistants, and others. Type 3 devices are powerful workstations that are capable of handling complex tasks with ease. For our implementation of CEaaS, we use NodeMCUs as the Type 1, Raspberry Pis (RPi) as Type 2, and a workstation with i5 core processor as Type 3 devices, respectively.

3.1 On-Field IoT Device Implementation

We use Type 1 and Type 2 devices for assuming the role of the IoT users on a laboratory scale. Fig. 2 shows a collection of NodeMCUs (bottom left) in our laboratory setup. The NodeMCU has an onboard ESP8266 microcontroller, which is a self-contained WiFi networking solution. This feature gives a hassle-free solution to connect to the Internet. It needs a voltage of 3.3V to power up with 16MB of flash memory. In this work, we consider the CPU clock cycles ($\mathcal{C}$) and RAM ($\mathcal{R}$) to determine the category of the devices. The NodeMCUs have a 32 bit CPU with $\mathcal{C}$ in the range of $80 - 160$MHz and have around 48KB of usable $\mathcal{R}$. The top left in Fig. 2 shows a collection of RPi devices in use. It also needs a voltage of 3.3V to power up. It has a 64 bit CPU with 1.4 GHz $\mathcal{C}$ and 1 GB $\mathcal{R}$. Raspberry Pi devices use a Linux-based operating system, and the NodeMCUs use dedicated execution codes, which are fit for appliance usage.

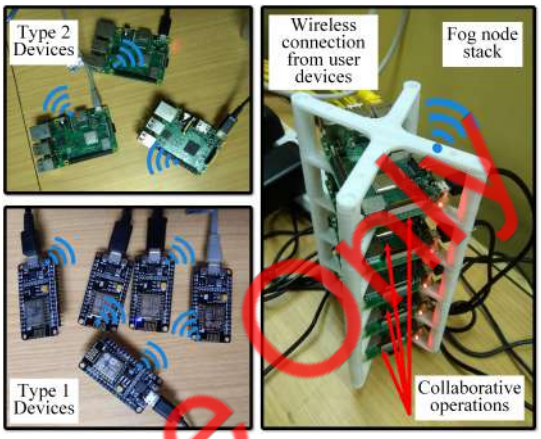


Figure 2: Combinations of NodeMCUs and RPis as Type 1 and Type 2 devices forming the on-field device layer

3.2 Fog-Layer Device Implementation

In our implementation of CEaaS, we use a collection of Type 2 devices as shown in the top left of Fig. 2 and a workstation (Type 3) as FNs. As a workstation, we use a Dell Inspiron 15 laptop with an i5 core processor as the FN. Additionally, we also use a stack of Type 2 devices, as shown in Fig. 2 (right). It may be noted that this stack is designed to dynamically balance the workload among themselves, the operational details of which is beyond the scope of this work.

4 Provisioning CEaaS

Due to the vast diversity of IoT devices, it is often difficult to determine their absolute configuration. To mitigate this issue, we assume that the devices send their configuration details to a nearby FN before applying any encryption algorithm. As shown in Fig. 1, the IoT devices entering a new network initially send the required information to a nearby FN while notifying a data transfer (Step 1). Upon receiving the information, the FN decides the category of the device, checks the network link quality, and determines the appropriate encryption scheme (Step 2). The FN sends this encryption scheme to the device as a decision along with a suitable key (Step 3). Using the received algorithm and key, the IoT device encrypts the data (Step 4) and sends it to the destined devices (Step 5). Since it is difficult to characterize the devices precisely, we design a FIS for the FNs to determine the user device category based on the received data. The fuzzy system relies on literal antecedents and consequents, making it easier to solve problems with obscure parameters. We further design another FIS for the FNs to determine the best encryption scheme for the user devices based on the computed category, data size, and network bandwidth.

4.1 Encryption Algorithms Considered

As a feasibility test for CEaaS, among the popular encryption techniques, we use AES-128, AES-256, and RSA as the encryption choices for CEaaS. For each of these schemes, we use ECC [4] for selecting the keys as it uses the symmetric elliptical curve equation, which increases security. *Advanced encryption standard (AES)-128* is a symmetric encryption algorithm, which takes the plaintext as input into a substitution table, and shifts/shuffles it row and column-wise. It then performs an exclusive or (XOR) operation on each column using different parts of the encryption key, the size of which determines the number of rounds to complete. AES-128 uses a 128-bit key and needs 10 rounds. Mathematically, for a plaintext $\mathbb{P}$, the ciphertext $\mathbb{C}$ is represented as $C = AES(\mathbb{P}, 128 - \text{bit key})$. *AES-256* on the other hand is the same as AES-128 with a 256-bit encryption key and needs 14 rounds to produce the ciphertext. Although it is more secure than AES-128, there exists a tradeoff against its operational time. It is mathematically represented as $C = AES(\mathbb{P}, 256 - \text{bit key})$. *RSA* is an asymmetric encryption algorithm with the public key consisting of a pair of two numbers. One of them is the product of two large prime numbers, and the private key is a derivative of the same. Since correct factorization may identify the keys, the strength of RSA depends on the key size. Typically, the key sizes here are 1024 or 2048 bits long, implying that RSA is computationally higher than the AES-128 and AES-256. Mathematically, for any two prime numbers α_1 and α_2, and $\phi = (\alpha_1 - 1)(\alpha_2 - 1)$, the public key is (n, e), where $n = \alpha_1 \times \alpha_2$ and $1 < e < \phi$. Meanwhile, RSA represents the private key as (d, α_1, α_2), where $1 < d < e$ such that $ed = 1 mod \phi$. In this work, we assume that the user devices and the FNs exchange keys through secure channels and that the encryption routines are pre-fetched in each of them, removing the need for on-the-fly program updates.

5 Problem Formulation

In this section, we present the network and device parameters and formulate the proposed two-level fuzzy decision process. We consider the IoT devices with data, represented as two-tuples $< \mathbb{D}_{sz}, \theta_{cyc} >$, where $\mathbb{D}_{sz}$ is the data size and θ_{cyc} is the number of necessary CPU cycles.

5.1 Energy Consumption

We calculate the energy consumption in the devices (irrespective of the types) as a function of $\mathcal{C}$ (cycles/second). We obtain the number of cycles that each device uses (c_{dev}) to complete each task from the CPU utilization (%) using command line applications. It may be noted that the tasks refer to the decision-making process and necessary encryptions in this work. The energy consumption per cycle κ varies depending on the processor material. Using these parameters, we calculate the energy consumption as [15]:

$$\mathcal{E} = \kappa \times (\mathcal{C})^2 \times c_{dev} \tag{1}$$

5.2 Delay

We consider the delay in this work as the summation of the time for executing CEaaS and that for network communications. We calculate the execution time (t_{exe}^{CEaaS}) as $t_{exe}^{CEaaS} = \theta_{cyc}/\mathcal{C}$. We calculate the network delay as the sum of transmission time (t_{dev}^{trans}) and propagation time (t_{dev}^{prop}). t_{dev}^{trans} is the time necessary for transferring $\mathbb{D}_{sz}$ (bits) of data from the user devices to the fog layer. For a data rate R, $t_{dev}^{trans} = \mathbb{D}_{sz}/R$. We calculate R using the channel capacity formula $R = BWlog(1 + SINR)$, where BW is the bandwidth and $SINR$ is the signal to interference plus noise ratio. We obtain the $SINR$ as:

$$SINR = \frac{p_{dev} h_{dev}}{\sum_{k=\mathcal{D}-d_{dev}}^{|\mathcal{D}|} p_k h_k + \sigma^2} \tag{2}$$

where, p_{dev} is the power of the signal and h_{dev} is the signal gain. The term $\sum_{k=\mathcal{D}-d_{dev}}^{|\mathcal{D}|} p_k h_k$ represents the interference from the neighboring devices and σ^2 is the Additive White Gaussian Noise (AWGN) in the signal. On the other hand, we calculate t_{dev}^{prop} as the time for transferring the data to the FN through the wireless channel. Mathematically, $t_{dev}^{prop} = \mathcal{N}\frac{d}{v}$, where $\mathcal{N}$ is the number of packets, d is the distance between the IoT device and FN, and v is the speed of the packet in the channel (speed of light in this work). Using these parameters, we calculate the total delay (t_{dev}^{total}) as:

$$t_{dev}^{total} = t_{exe}^{CEaaS} + t_{dev}^{trans} + t_{dev}^{prop} \tag{3}$$

5.3 Two-Level Fuzzy Based Decision

We briefly present the basics about FIS related to this work. FIS consists of fuzzy sets and rules. These sets are represented in ordered pairs in the form $F = (a, \mu_z(a))|a \in \mathbb{A}$ where $\mathbb{A}$ is the universe of discourse and $\mu_z(a)$ is the membership function of a for z. As the membership functions are dependent on the nature of the problem, we define z and its respective distributions in the subsequent sections. The FIS makes its inferences based on rules in the form of conditional statements like $R_i = If\ \alpha\ then\ \beta$, where α is an antecedent consisting of a combination of conjunctions and disjunctions, and β is a consequent for the corresponding antecedent for the i^{th} rule.

Among FIS models, the Sugeno model [16] requires the consequent to be a function of the input antecedents and the Tsukamoto model [17] requires the output function to be monotonic. Since our model does not fit in these two FIS systems, we choose the Mamdani model [18] for interpreting our fuzzy rules. In the Mamdani model, in case there are n inputs for a rule R_i at time instant t, it considers the minimum among all the membership values for the final aggregation with the other rules. Mathematically, $\mu_{R_i,t}(a_1, a_2, ..., a_n) = min[\mu_{z_1,t(a_1)}, \mu_{z_2,t(a_2)}, ..., \mu_{z_n,t(a_n)}]$. For the aggregation of all the rules, the Mamdani model considers the maximum of all of them as the output. Mathematically, $\mu_{R,t}(k) = max[\mu_{R_i,t}(a_1, a_2, ..., a_n)] \forall i \in$ *number of rules* . Unlike the other two FIS systems, the Mamdani model also needs a defuzzifier to get the crisp values for the corresponding output. To achieve this, we use the centroid technique which is calculated as,

$$\psi_{p,t} = \frac{\int_0^1 \mu_{R,t}(k).k\ dk}{\int_0^1 \mu_{R,t}(k)\ dk} \tag{4}$$

where $\psi_{p,t}$ is the final crisp output for parameter p at time instant t.

We propose CEaaS as a two-level FIS. The FIS in level 1 takes inputs to determine the category of the on-field

IoT device. This category acts as input for the second level which finally determines the encryption scheme. We explain each layer in detail in the subsequent sections.

On receiving the requests, the FNs first decide the category of the user devices using the CPU clock cycles and RAM availability. The FNs then check the current network condition. Then, along with the device category and data type, they decide an encryption algorithm along with the corresponding key for the requesting user devices.

5.3.1 Level-1 FIS

In this stage, the FN decides the category of the IoT devices ($\mathcal{P}_{dev}$). We design this system with the IoT devices' $\mathcal{C}$ and $\mathcal{R}$ as inputs and its category as output. We consider {low, medium, high} as the fuzzy set of literals for $\mathcal{C}$ as well as for $\mathcal{R}$. Also, we consider {good, better, best} as the set of literals for $\mathcal{P}_{dev}$. Although the selection of membership functions is an open problem [19], as the device behavior remains unchanged for a range of intervals of the device configurations ($\mathcal{C}$ and $\mathcal{R}$), we use trapezoidal membership functions for fuzzification of the crisp values. The trapezoidal membership function takes the lower (l_{low}) and upper (l_{up}) limits along with supports (l_{low}^{sup} and l_{up}^{sup}), such that $l_{low} < l_{low}^{sup} < l_{up}^{sup} < l_{up}$. We calculate the membership values from the universe of discourse elements x_{ele}^{univ} as:

$$\mu^{trap}(x_{ele}^{univ}) = \begin{cases} 0, & (x_{ele}^{univ} < l_{low}) \text{ or } \\ & (x_{ele}^{univ} > l_{up}) \\ \frac{x_{ele}^{univ} - l_{low}}{l_{low}^{sup} - l_{low}}, & l_{low} \leq x_{ele}^{univ} \leq l_{low}^{sup} \\ 1, & l_{low}^{sup} \leq x_{ele}^{univ} \leq l_{up}^{sup} \\ \frac{l_{up} - x_{ele}^{univ}}{l_{up} - l_{up}^{sup}}, & l_{up}^{sup} \leq x_{ele}^{univ} \leq l_{up} \end{cases} \quad (5)$$

On the other hand, for $\mathcal{P}_{dev}$, we use the triangular membership function. The triangular membership function does not require the support elements l_{low}^{sup} and l_{up}^{sup}. It uses the limits l_{low} and l_{up} along with a middle value v_{mid} to determine the peak value of 1. We calculate the membership values from the universe of discourse elements x_{ele}^{univ} as:

$$\mu^{tri}(x_{ele}^{univ}) = \begin{cases} 0, & x_{ele}^{univ} \leq l_{low} \\ \frac{x_{ele}^{univ} - l_{low}}{v_{mid} - l_{low}}, & l_{low} < x_{ele}^{univ} \leq v_{mid} \\ \frac{l_{up} - x_{ele}^{univ}}{l_{up} - v_{mid}}, & v_{mid} < x_{ele}^{univ} < l_{up} \\ 0, & x_{ele}^{univ} \geq l_{up} \end{cases} \quad (6)$$

For $\mathcal{C}$, we set the values for low as $35 - 70$ MHz, medium as $60 - 100$ MHz, and high for the others (higher than 100 MHz). In the case of RAM, we set low up to 0.07 KB, medium up to 1.2 KB, and high for values above that. Using these $\mathcal{C}$ and $\mathcal{R}$ values, we determine the $\mathbb{P}_{dev}$ in the range [0,1] and set the literals as good for $0 - 0.3$ values, better for values between $0 - 0.6$, and best for $0.4 - 1$, respectively.

5.3.2 Level-2 FIS

At this level, the FN decides the appropriate encryption algorithm. We consider $\mathcal{P}_{dev}$ determined in the first level FIS along with network link bandwidth ($\mathcal{B}$) and the size of impending data transfer ($\mathcal{D}_{sz}$) as inputs into the FIS in the second level. On the same lines as in Section 5.3.1, we consider {low, medium, high} as the set of literals for the $\mathcal{B}$ and $\mathcal{D}_{sz}$. For $\mathcal{B}$, we set $200 - 800$ Kbps as a medium, anything below that as low, and anything above 600 Kbps as high. Since $\mathcal{B}$ is a continuous function, we consider the Gaussian membership function. The Gaussian membership function takes the center (v_{center}) and width (σ_{width}^{fuzzy}) from the elements in universe of discourse and we calculate as:

$$\mu^{gauss}(x_{ele}^{univ}) = exp\Big(- \frac{(v_{center - x_{ele}^{univ}})^2}{(2\sigma_{width}^{fuzzy})^2} \Big) \quad (7)$$

On the other hand, we consider the trapezoidal membership function for the $\mathcal{D}_{sz}$. We consider the $\mathcal{D}_{sz}$ of textual data as low ($0 - 10$ MB), images as medium ($8 - 100$ MB), and that of multimedia as high, which generally ranges above 75 MB. Finally, we select the encryption schemes ($\mathcal{P}_{ec}$) based on these parameters. As mentioned earlier, we only consider the popular AES symmetric encryption scheme with key sizes 128 and 256 respectively, and a heavyweight RSA public-key encryption scheme [1]. Similar to $\mathbb{P}_{dev}$, we set the values for $\mathcal{P}_{ec}$ to [0,1] and set them as AES-128 for $0 - 0.3$, AES-256 for $0 - 0.6$, and RSA for 0.3 and higher, respectively. We have included the visualization for the literals and corresponding membership functions for both Level-1 and Level-2 FIS in the supplementary file.

Proposition 1. *Irrespective of the RC IoT device types, although the security level may vary, CEaaS facilitates the transmission of secured data over the network.*

Proof. The devices use varying key lengths for each of the encryption algorithms. For a n bit key length, a device may use a key from 2^n combinations, implying the probability of guessing the correct key as $1/2^n$, thereby increasing the security proportional to the key bits. Additionally, as mentioned in Section 4.1, the number of execution rounds of AES adds more security. Further, using ECC makes malicious activities more difficult. Taking into account these considerations, CEaaS assures secure data transmission. □

Proposition 2. *CEaaS offers optimal decisions to the IoT devices.*

Proof. We prove the feasibility of CEaaS by proving that its decisions offer minimum delays. While the hardware configurations of the IoT devices remain relatively constant (compared to network configurations), the objective of this work is to minimize Equation 3. Considering the variability in t_{dev}^{trans} (compared to t_{exe}^{CEaaS} and t_{dev}^{prop}), we prove that $R = BWlog(1 + SINR)$ has a maxima. On double differentiation R with respect to the power of the signal, we obtain the following expressions:

$$\frac{\partial R}{\partial p_{dev}} = BW \frac{h_{dev}}{h_{dev}p_{dev} + \sum_{k=\mathcal{D}-d_{dev}}^{|\mathcal{D}|} p_k h_k + \sigma^2}$$
$$\frac{\partial^2 R}{\partial p_{dev}^2} = -BW \frac{h_{dev}^2}{(h_{dev}p_{dev} + \sum_{k=\mathcal{D}-d_{dev}}^{|\mathcal{D}|} p_k h_k + \sigma^2)^2} \quad (8)$$

Owing to the negative values in Equation 8, we conclude that CEaaS offers decisions that take minimal delay. □

Proposition 3. *CEaaS is immune to attacks.*

Proof. From Theorem 1 in [20], the probability of breaking a FIS is a function of time. The two-level FIS system in CEaaS increases the algorithmic complexity. Further, from Proposition 2, CEaaS offers minimal delays, which elevates

Table 1: Encryption time per byte (in seconds) for Type 1 and Type 2 devices while using different schemes

	AES 128	AES 256	RSA
Type 1	6.41×10^{-6}	8.98×10^{-6}	Not Applicable
Type 2	4.64×10^{-7}	5.39×10^{-7}	1.82×10^{-4}

Table 2: Encryption time per byte (in μ-seconds) for Type 1 devices using different lightweight AES schemes

	Scheme \ Key bit	128	256
Type 1	AES	6.41	8.98
	AES Small	8.93	11.65
	AES Tiny	8.19	10.92

its immunity. Additionally, Proposition 1 assures secured transmissions. We account for these features in CEaaS and conclude that it is immune to attacks. □

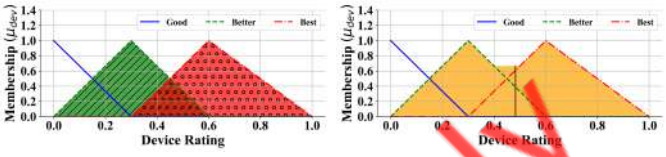
6 Result and Discussions

In this work, to demonstrate the feasibility of CEaaS, we first implement the proposed scheme using a Dell Inspiron i5 laptop (Type 3) as the FN while using RPis and NodeMCUs as the IoT devices. Additionally, we also demonstrate the performance of CEaaS by using Type 2 devices as FNs. The Type 1 devices in our experiment connect to the FN via wired as well as wireless links (WiFi). Figure 2 depicts our hardware setup. We first present our observations from the two-level FIS and then the operation, network, and energy overheads. Tables 1 and 2 present the per byte delays in Type 1 and Type 2 devices for different encryption schemes.

6.1 CEaaS Output for Two-Level FIS

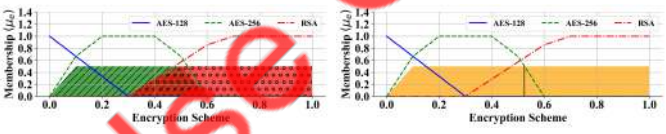
In this section, we describe an instance of our experiment to demonstrate the working of the CEaaS two-level FIS. Initially, the first level FIS determines the $\mathcal{P}_{dev}$ of the IoT device. Upon receiving the request from the user devices, the fuzzy input parameters ($\mathcal{C}$ and $\mathcal{R}$) influence the output. In Fig. 3a, the affinity of the device category to be better and best are represented as dashes and circles in the triangles, respectively. We show the aggregation of these influences in Fig. 3b. The vertical line in between 0.4 and 0.6 is the $\mathcal{P}_{dev}$ determined by the FIS in level-1. We speculate that the request to the FN came from a Type 2 device at that instant. However, since it is already performing other operations together with maintaining communication with other devices, its $\mathcal{C}$ and $\mathcal{R}$ might have been lower than usual.

As illustrated in Fig. 1, the output from the FIS in level-1 ($\mathcal{P}_{dev}$) is sent as an input into the FIS in level-2. Along with $\mathcal{P}_{dev}$, the FN considers the $\mathcal{D}_{sz}$ as well as $\mathcal{B}$ by sensing the network link. Here, for simplicity, we assume that the network condition will remain pseudo-static while making the encryption scheme decision. The influence of these inputs is as shown in Fig. 4a. Fig. 4b shows the aggregation of these influences from the parameters and the vertical line in between 0.5 and 0.6 represents $\mathcal{P}_{ec}$. We select the encryption scheme which has a closer Euclidean distance from the curve to the vertical line. In this case, the FIS in the second level finalizes the use of the RSA algorithm for the user device. We believe that although $\mathcal{P}_{dev}$ output was corresponding to the literal 'better', the $\mathcal{B}$ and $\mathcal{D}_{sz}$ influenced the decision for a high level of security. From this, we conclude that CEaaS selects the appropriate encryption scheme for the IoT devices. Since the routine for CEaaS remains the same irrespective of the FN used, the output ($\mathcal{P}_{ec}$) also remains the same.



(a) Influence of incoming request on first-level FIS (b) Prediction of user device category on first-level FIS

Figure 3: Output of FIS in level-1



(a) Influence of input parameters on second-level FIS (b) Encryption scheme decision on second-level FIS

Figure 4: Output of FIS in level-2

6.2 CPU and Memory Usage

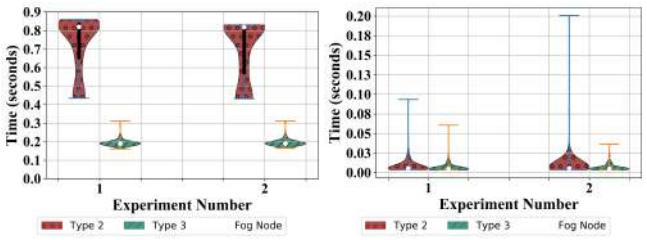
We record the CPU and memory usage in each of the devices under the CEaaS scheme and tabulate them in Table 3. We observe that the CPU usage in the Type 2 devices is minuscule (in the range of 0.24%) for all encryption algorithms, and further less in Type 3 devices (0.15%). We observe such low values due to higher CPU clock cycles in each of these devices, specifically 1.2 and 2.4 GHz, respectively. As the Type 1 devices have a limited CPU capacity of only 80 MHz, we observe a higher usage value of 47%. It may be noted that we assigned Not Available (NA) under RSA in the case of Type 1 devices as RSA requires a minimum of 64 bit processors, and the Type 1 devices have only 32 bit processors, which makes implementation challenging. We observe a similar trend in memory consumption, and we make the same comment as that of CPU usage for the observations. From the values in Table 3, we infer that CEaaS correctly assesses the devices and makes efficient use of the available resources in the IoT devices.

6.3 Operational Delays

In this section, we present the delays endured while implementing CEaaS. To capture the events, we send requests from arbitrary IoT devices to the FNs for 100 times and present the results from two such experiments. To get a better insight into the delays, we present a breakdown of the delays and present in terms of CEaaS execution and network transmission, respectively. Due to the difference in $\mathcal{C}$, we observe a higher delay in the two-level FIS system of CEaaS while using Type 1 devices as FNs in Fig. 5a. However, the

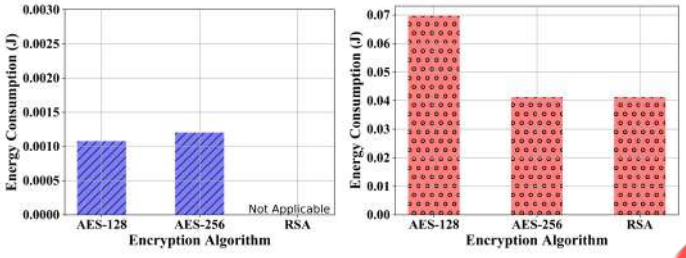
Table 3: CPU and memory consumptions on different devices under CEaaS scheme

Device type	CPU usage			Memory usage		
	AES-128	AES-256	RSA	AES-128	AES-256	RSA
Type 1	42%	47%	NA	36.3%	35%	NA
Type 2	0.24%	0.144%	0.144%	0.89%	0.93%	0.89%
Type 3	0.05%	0.1%	0.15%	0.0017%	0.0018%	0.0015%



(a) CEaaS delay (b) Network delay

Figure 5: Comparison of delays on using Raspberry Pi and i5 core processor as fog node



(a) Type 1 (b) Type 2

Figure 6: Comparison on energy consumption while using different encryption schemes in Type 1 and Type 2 devices

delay is in the range of milliseconds, which is a minuscule difference ratio as compared to configurations, which makes our proposed service suitable for fog computing implementations with RC FNs. On the other hand, since the packet size of the request and $\mathcal{P}_{ec}$ decision remains the same, we observe similar delays in network transmission in Fig. 5b. However, we observe a sharp peak in the 2^{nd} experiment on using Type 2 devices as FNs. Intuitively, the network congestion may be the cause for such a delay as the delays mostly concentrate on the lower values.

6.4 Energy Consumption

We compute the energy consumption by the Type 1 and 2 devices under the CEaaS scheme using equation 1 and the CPU consumptions in Table 3. As shown in Fig. 6a, we observe a maximum energy consumption of 0.0013 Joules in Type 1 devices while performing the AES-256 encryption routine and almost 15% less energy consumption while executing AES-128. Such observations are because of the difference in the shuffling rounds in the AES schemes. The Type 1 devices do not perform RSA encryptions due to the reason mentioned in Section 6.2. On the other hand, we observe a higher consumption of 0.7 Joules in Type 2 devices on performing AES-128 and lower consumptions for AES-256 and RSA in Fig. 6b. While we should have

Table 4: Comparison of ECaaS with existing encryption schemes based on static (S)/ dynamic (D), centralized (C)/ decentralized (DC), number of devices, channel condition, data size, and device type

	S/ D	C/ DC	No. of Dev	Ch. Cond.	Data Size	Dev Type
Kung and Hsiao [8]	D	C	✓	✗	✗	✗
Boateng *et al.* [14]	D	D	✗	✗	✓	✓
Choi [13]	D	D	✗	✓	✗	✗
CEaaS	D	D	✓	✓	✓	✓

observed minimal energy consumption in the case of AES-128, we attribute this observation to the reason that the FNs may be simultaneously serving other user requests. We comment that in either case, we observe a maximum energy consumption of only 0.07 Joules under the CEaaS scheme, further strengthening its feasibility on low-powered devices.

6.5 Comparison of CEaaS with Existing Schemes

As mentioned earlier, CEaaS is an encryption scheme for IoT devices operating in a dynamic environment. CEaaS takes into account the varying channel conditions, device type, the data size for transmission, and others. Several other schemes exist in the literature explicitly designed for RC IoT devices or for optimizing channel consumption. In Table 4, we illustrate how CEaaS overcomes limiting factors as compared to the other existing schemes. The existing solutions focus on a few sets of parameters and offer security. Such targetted solutions open the scope for underperformance due to the factors not considered. We also observe that CEaaS makes encryption decisions as a fusion of both device and network-level parameters, making it suitable for IoT environments.

7 Conclusion

In this work, we proposed CEaaS — a service that dynamically selects an encryption scheme for IoT devices based on its current configuration, network state, and the type of data to be transferred. To the best of our knowledge, this is the first attempt at considering the heterogeneity of the devices and attempting diversity in encryption algorithms using a single platform. We presented CEaaS as a two-level FIS to accommodate devices of any type and to remove binary assessment in the selection of encryption schemes. Additionally, we demonstrated the feasibility and advantages of CEaaS by deploying it in the fog layer using both powerful and RC devices as FNs.

In this work, for simplicity, we assumed that the devices have the encryption routines pre-fetched inherently and only limited to the selection of the same. In the future, we

plan to pass the corresponding routines from the FNs to the devices using over-the-air (OTA) updates.

REFERENCES

[1] S. Singh, P. K. Sharma, S. Y. Moon, and J. H. Park, "Advanced Lightweight Encryption Algorithms for IoT Devices: Survey, Challenges and Solutions," *Journal of Ambient Intelligence and Humanized Computing*, May 2017.

[2] J. M. Mcginthy and A. J. Michaels, "Secure Industrial Internet of Things Critical Infrastructure Node Design," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8021–8037, Oct. 2019.

[3] T. Ara, P. G. Shah, and P. M, "Dynamic key Dependent S-Box for Symmetric Encryption for IoT Devices," in *Proceedings of Second International Conference on Advances in Electronics, Computers and Communications (ICAECC)*, Feb. 2018, pp. 1–5.

[4] H. Cohen, G. Frey, R. Avanzi, C. Doche, and T. Lange, *Handbook of Elliptic and Hyperelliptic Curve Cryptography*. New York: Chapman and Hall: CRC Press, 2005.

[5] A. J. Bidgoly and H. J. Bidgoly, "A Novel Chaining Encryption Algorithm for LPWAN IoT Network," *IEEE Sensors Journal*, vol. 19, no. 16, pp. 7027–7034, Aug. 2019.

[6] A. Esfahani, G. Mantas, J. Ribeiro, J. Bastos, S. Mumtaz, M. A. Violas, A. M. De Oliveira Duarte, and J. Rodriguez, "An Efficient Web Authentication Mechanism Preventing Man-In-The-Middle Attacks in Industry 4.0 Supply Chain," *IEEE Access*, vol. 7, pp. 58 981–58 989, May 2019.

[7] G. S. Aujla and A. Jindal, "A Decoupled Blockchain Approach for Edge-Envisioned IoT-Based Healthcare Monitoring," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 2, pp. 491–499, Feb. 2021.

[8] Y. Kung and H. Hsiao, "GroupIt: Lightweight Group Key Management for Dynamic IoT Environments," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 5155–5165, Dec. 2018.

[9] S. Roy, A. K. Das, S. Chatterjee, N. Kumar, S. Chattopadhyay, and J. J. P. C. Rodrigues, "Provably Secure Fine-Grained Data Access Control Over Multiple Cloud Servers in Mobile Cloud Computing Based Healthcare Applications," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 1, pp. 457–468, Jan. 2019.

[10] K. Gai, K. R. Choo, M. Qiu, and L. Zhu, "Privacy-Preserving Content-Oriented Wireless Communication in Internet-of-Things," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 3059–3067, Aug. 2018.

[11] A. Saleem, A. Khan, S. U. R. Malik, H. Pervaiz, H. Malik, M. Alam, and A. Jindal, "FESDA: Fog-Enabled Secure Data Aggregation in Smart Grid IoT Network," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6132–6142, Jul. 2020.

[12] R. Melki, H. N. Noura, M. M. Mansour, and A. Chehab, "An Efficient OFDM-Based Encryption Scheme Using a Dynamic Key Approach," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 361–378, Feb. 2019.

[13] J. Choi, "Channel-Aware Randomized Encryption and Channel Estimation Attack," *IEEE Access*, vol. 5, pp. 25 046–25 054, Nov. 2017.

[14] K. Boakye-Boateng, E. Kuada, E. Antwi-Boasiako, and E. Djaba, "Encryption Protocol for Resource-Constrained Devices in Fog-Based IoT Using One-Time Pads," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3925–3933, Apr. 2019.

[15] T. X. Tran and D. Pompili, "Joint Task Offloading and Resource Allocation for Multi-Server Mobile-Edge Computing Networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 856–868, Jan. 2019.

[16] M. Sugeno and G. Kang, "Structure Identification of Fuzzy Model," *Fuzzy Sets and Systems*, vol. 28, no. 1, pp. 15 – 33, 1988.

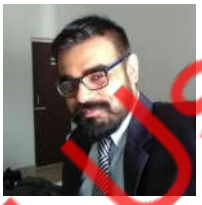
[17] J. R. Jang and C. Sun, *Neuro-fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1997.

[18] E. Mamdani and S. Assilian, "An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller," *International Journal of Man-Machine Studies*, vol. 7, no. 1, pp. 1 – 13, 1975.

[19] A. Sadollah, "Introductory Chapter: Which Membership Function is Appropriate in Fuzzy System?" in *Fuzzy Logic Based in Optimization Methods and Control Systems and Its Applications*, A. Sadollah, Ed. Rijeka: IntechOpen, 2018, ch. 1.

[20] A. Juels and M. Wattenberg, "A Fuzzy Commitment Scheme," in *Proceedings of the 6^{th} ACM conference on Computer and communications security*, Nov. 1999, pp. 28–36.

Pallav Kr. Deb is a Ph.D. Research Scholar in the Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, India. He received his M.Tech degree in Information Technology from Tezpur University, India in 2017. Prior to that, he has completed the B. Tech degree in Computer Science from the Gauhati University, India in 2014. The current research interests of Mr. Deb include UAV swarms, THz Communications, Internet of Things, Cloud Computing, Fog Computing, and Wireless Body Area Networks. Further details are available in https://pallvdeb.github.io/



Anandarup Mukherjee is currently a Senior Research Fellow and Ph.D. Scholar in Engineering at the Department of Computer Science and Engineering at Indian Institute of Technology, Kharagpur. He finished his M.Tech and B.Tech from West Bengal University of Technology in the years 2012 and 2010, respectively. His research interests include, but are not limited to, networked robots, unmanned aerial vehicle swarms, Internet of Things, Industry 4.0, 6G and THz Networks, and enabling deep learning for these platforms for controls and communications. His detailed profile can be accessed at http://www.anandarup.in

Sudip Misra (M'09–SM'11) is a Professor with the Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur. He received his Ph.D. degree in Computer Science from Carleton University, in Ottawa, Canada, and the masters and bachelor's degrees, respectively, from the University of New Brunswick, Fredericton, Canada, and the Indian Institute of Technology, Kharagpur, India. Dr. Misra is the Associate Editor of the IEEE Transactions Mobile Computing and IEEE Systems Journal, IEEE Transactions on Sustainable Computing, IEEE Network, and Editor of the IEEE Transactions on Vehicular Computing. His current research interests include algorithm design for emerging communication networks and Internet of Things. Further details about him are available at http://cse.iitkgp.ac.in/ smisra/.