



Context-Aware Security–Energy Optimization for Lightweight Cryptography in Resource-Constrained IoT Devices

Masood Ahmad*

Department of Computer Science, Abdul Wali Khan University
Mardan. Corresponding Author Email: masood@awkum.edu.pk

Arshad Ali

Directorate of IT, Abdul Wali Khan University, Mardan

Ishtiaq Wahid

Department of Computer Science, Abdul Wali Khan University,
Mardan

Ihtiram Ullah

Department of Computer Science, Abdul Wali Khan University,
Mardan

Abstract

In recent times, the swift growth of Internet of Things (IoT) devices has led to the rise in demand for lightweight security mechanisms that can safeguard sensitive information in challenging conditions with restrictions with regard to energy, memory, and computational resources. Traditional cryptographic solutions can provide strong protection but may involve high computational and energy costs that are unreasonable for most of the IoT devices. Additionally, usage of static settings for cryptographic means does not solve the problem of dynamic security requirements related to IoT environments changeable conditions of operating. In this paper, Context-Aware Security–Energy Optimization (CASEO) framework is proposed to address the issue of effective cryptographic solutions selection and implementation in IoT devices. It can be accomplished due to the analysis of contextual information such as energy margins available for the device, computational capabilities, communication conditions, and security needs in the selected application region. An adaptive honeybee optimization mechanism has been utilized for



determining Pareto-optimal configurations in the security-energy search space. The framework integrates constraints to avoid selecting configurations satisfying energy requirements at the expense of excessive security degradation. The proposed approach has been assessed through comparisons with standard lightweight cryptographic configurations and with advanced multi-objective optimization strategies based on the parameters such as security, energy consumption, execution time, latency, and quality of optimization. In this way, the framework makes it possible to develop a method for balancing security and resources utilization.

Keywords: Internet of Things, lightweight cryptography, context-aware security, multi-objective optimization, energy efficiency, resource-constrained devices, Honey Bee Optimization, cryptographic configuration, IoT security, security–energy trade-off.

1. Introduction

The Internet of Things (IoT) is a key computing paradigm in which various standalone physical devices exchange information and provide various services in multiple fields such as healthcare, smart homes, industrial control, transportation, agriculture, etc. The increased number of connected devices has also increased the attack surface of IoT devices. IoT devices can be used to collect sensitive information and provide it indirectly, while operating in environments vulnerable to interception, alteration of data, identification, replay attacks, or intrusion. Therefore, confidentiality, integrity, authentication, and protection of data are necessary to ensure safe communication via IoT devices [1]. However, implementing security in IoT is much more complex than in traditional computing systems. Most IoT devices only have a limited processing power, small amount of memory, little battery power, and low bandwidth. Using security means based on cryptography leads to an increase of computing and communication costs, which may shorten devices' life span. Therefore, it is important to use light cryptography to ensure security due to low computational, memory, and energy requirements of cryptographic processes [2].

Despite the fact that lightweight cryptographic algorithms are developed with consideration of limited environments, the selection of an optimal cryptographic mechanism is still a challenging optimization problem.



Different algorithms and their configurations differ in terms of their characteristics with respect to the level of security strength, execution time, memory consumption, energy consumption, throughput, and communication overhead. The configuration ensuring a high level of security can demand significant computing resources, and an extremely efficient configuration can provide insufficient protection for security-critical applications. Hence, the aspect of lightweight cryptography cannot be considered only from the point of view of selecting an algorithm; it is a multi-dimensional decision-making problem [3].

Another disadvantage of many IoT security approaches is their static nature. A cryptographic configuration is usually selected at the design stage of the system, which is a sort of a unique point in its lifecycle, and its use is continued irrespective of the state of the device and the context in which it operates. However, in a real IoT environment, the context can change drastically. In particular, a battery-operated sensor may need strong conservation mode when there is little battery life left, while at the same time, the same device can perform attractive encryption operations when enough energy is available. The same is true for healthcare monitoring devices that transmit important data compared to environmental sensors that are less affected by a terrorist attack. The aspects mentioned above urge the development of context-aware security mechanisms that adjust their cryptographic decisions based on any event of the operating conditions [4].

Multi-objective optimization is necessary for this task as it achieves simultaneous optimization of competing objectives. Rather than looking for a single solution based on one performance indicator, multi-objective methodology allows for identifying a number of Pareto-optimal solutions demonstrating different trade-offs between security and resource consumption, which further can be selected in dependence on the context and application [5]. The metaheuristic optimization approach looks much more attractive in this case as the cryptographic configuration space has nonlinear relationships and conflicting aims making the exhaustive optimization process time-consuming [5]. This paper proposes applying the adaptive Honey Bee Optimization method for the exploration of the security–energy space of the problem. The idea is borrowed from the social hunting



behavior of bees that exploit food sources depending on their quality. The mechanism has been adapted for the IoT cryptographic configuration problem so that the potential candidate solution is represented as a combination of cryptographic parameters that are evaluated based on several security and resource consumption criteria [6]. In addition, the optimization process has been modified by adding the contextual information so that security and resource consumption could vary depending on the current condition of an IoT device [6].

The key concept of the suggested structure is not only to lessen energy consumption. A high focus on energy effectiveness might lead to cryptographic setups that require little computing power but are not suitable for safety-critical fields. On the other hand, putting security first without accounting for resource availability can increase the potential of the battery failing and the quality of the application. Therefore, the suggested method looks for a variable and adaptive solution to these conflicting requirements. The security needs would act as limitations, while resource consumption, computing power, and latency become parts of the objective to be reached. The main contributions of the research are outlined below:

- A context-based light cryptographic approach is created for constrained IoT devices allowing to make cryptographic decisions depending on the changing conditions of the device, network, and application.
- A multi-objective optimization model of security and energy consumption is developed to allow taking into consideration security level, energy consumption, computing power, and latency simultaneously instead of using only one resource metric.
- The Adaptive Honey Bee Optimization technique is designed allowing the search of cryptographic configuration space and finding Pareto-optimized solutions within various restrictions applicable to IoT devices.
- The approaches are introduced that enable a change of the influence of security and resource efficiency dependent on the energy availability, computing power, communication opportunities, and security needs of the specific application.
- The effectiveness evaluation model that includes the evaluation of the efficiency of cryptographic configurations on the basis of the analysis of



the security, energy consumption, execution time, latency, and quality of the multi-objective optimization is developed.

The remainder of this paper is organized as follows. Section 2 reviews existing research on lightweight cryptography, IoT security optimization, context-aware security, and metaheuristic optimization. Section 3 presents the system model and formulates the multi-objective optimization problem. Section 4 describes the proposed context-aware security–energy optimization framework. Section 5 presents the adaptive Honey Bee Optimization algorithm and its implementation for cryptographic configuration selection. Section 6 describes the experimental environment, datasets or workload configurations, baseline methods, and evaluation metrics. Section 7 presents and discusses the experimental results. Finally, Section 8 concludes the paper and identifies directions for future research.

2. Related Work

The emerging field of lightweight cryptography has attracted serious interest as a possible solution to securing resource-limited IoT devices. The conventional cryptographic systems impose heavy requirements in terms of computational power, memory, and energy consumption once used in low-power sensors and embedded devices. As a result, lightweight cryptography aims to achieve a compromise between the level of security (confidentiality, integrity, and authentication) and the associated costs involved with its implementation. The studies of lightweight designs of block ciphers, authenticated encryption systems, and cryptographic primitives tailored for constrained environments have taken place recently. The process for standardization of lightweight cryptography initiated by NIST has shown the significance of authenticated encryption and hashing systems developed specifically for constrained conditions and ended with the selection of ASCON family in 2023 [7]. However, the operation of any cryptographic system highly depends on the properties of its hardware, the workload it has to process, the size of messages, and the conditions in which it works. Therefore, choosing an appropriate lightweight cryptographic system just because of its security level may not be the best solution for a specific IoT device [8].

The second direction of research has been concentrated on assessing the security-resource performance trade-offs of low weight cryptographic



algorithms. Customarily, publications have been based on the issues like run-time, memory consumption, energy consumption, throughput, and security implications as metrics of assessment. The results show that there is no single cryptographic mechanism applicable to all IoT systems. For example, algorithms designed for the operation in highly constrained nodes may not provide anything beneficial when applied in devices of a higher caliber, while applications sensitive to security may require the application of rather powerful algorithms despite higher resource consumption. This situation introduces a multi-criteria decision-making issue where security and efficiency should be addressed together [9].

Also, the research of context-aware security aims to solve the issues brought by the heterogeneous nature of IoT environment. The decisions connected with security in context-aware systems can be based on various data on the current state of devices, current energy level, the situation in the network, application requirements, and the risk related to the target identified. This allows for security tactics that are very flexible due to the current environment to be proposed, whereas existing approaches are mostly focused on attaining information on context-aware access control, authentication, intrusion detection, and security policy management, while exploring adaptive optimization of lightweight cryptographic configuration according to the constraints of resources and security less often [10], [11].

Research endeavors to explore optimization-based approaches for handling security issues in IoT systems as well as their performances in resource management related problems. It is noted that multi-objective optimization is especially relevant due to the fact that security decisions in IoT typically involve conflicting aims. For instance, the increase in cryptographic strength means a better security performance, however, it may also mean a longer execution time and higher energy consumption. Similarly, reducing the energy consumption may allow extending battery life but may also reduce the level of security achievable. Thus, various evolutionary as well as swarm optimization approaches have been employed to solve resource allocation, routing, task scheduling, security management, and parameter selection problems in IoT systems. The advantage of these approaches is that they can



effectively navigate complex decision-making horizons without going through every decision-making possibilities and options [12].

Among the metaheuristic techniques, Honey Bee Optimization stands out for dealing with nonlinearities and conflicting objectives, as this approach has been developed in order to explore alternative solutions while suitably exploiting the fruitful areas of search space. The original search principles based on foraging can be well applied to discrete as well as constrained optimization problems. Nevertheless, the utilization of a generic Honey Bee Optimization algorithm does not guarantee an appropriate solution to the dynamic IoT security requirements. A mechanism aimed at solving cryptographic issues of IoT systems must take into account the changing context, minimum requirements concerning the security levels, availability of the resources of device and competitive knowledge in terms of alternative notions of energy consumption and computational load [13].

In general, current studies indicate that lightweight cryptographic design, cryptographic performance evaluation, context-aware security in IoT, and metaheuristic optimization have advanced significantly. However, it seems that these directions have mostly been reviewed separately from each other. While lightweight cryptographic research usually pays attention to the design or implementation efficiency of algorithms, context-aware research mainly focuses on adapting security measures, and optimization research is devoted to one or several IoT resource management problems. It seems that there is still a demand for a universal approach to applying context information in order to dynamically optimize lightweight cryptographic solutions while ensuring a trade-off between security and energy and computational costs. Context-Aware Security-Energy Optimization framework is a solution to this issue since it combines context information, multi-objective optimization, and adaptive Honey Bee Optimization in one decision-making process for the resource-constrained IoT devices [14].

3. System Model and Problem Formulation

3.1 System Model

The suggested system suggests a heterogeneous IoT environment made up of sensing devices that have limited resources which collect, process, and transmit sensitive data either to an edge or cloud infrastructure. It is evident



that each individual IoT device may have some different parameters of capacity such as processing power, memory size, battery life, and conditions of network use. Hence, a fixed universal cryptographic approach cannot be applied in our case since the ideal balance between safety and resource usage may not be achieved by all devices.

Let us denote a set of IoT devices with $D = \{d_1, d_2, \dots, d_n\}$. The working context for every device d_i can be defined with the following vector:

$$C_i = [E_i, P_i, M_i, N_i, R_i]$$

where E_i is the energy level, P_i is the processing power, M_i is the memory available, N_i is the communicating state, and R_i is the security level required for the app. The represented vector is subject to updates so that the decision on a cryptography issue can be made without taking current conditions into consideration.

The set of light crypto settings can be described as $A = \{a_1, a_2, \dots, a_k\}$, where every setting describes a possible variant of crypto parameters such as the cryptographic algorithm selection; key size; authentication settings and other parameters of the implementation available on the IoT platform of interest. The system suggested in the paper identifies the proper configuration based on the current context of the considered device. For a device d_i , a potential crypto solution is denoted as:

$$X_i = x_{i1}, x_{i2}, \dots, x_{ik}$$

where each x_{ij} shows the chosen decision variable satisfying the requirement to be related to the crypto parameters. During the process of optimization, the algorithm looks for a viable X_i , which guarantees the proper level of security.

3.2 Security Model

In this paper, security is defined as a particular purpose of optimization and not as a property of an applied algorithm. Let $S(X_i)$ denotes the security score of a given cryptographic configuration X_i . The security score reflects key length, attack resistance, authentication, and security level requirements imposed by the application. To ensure that security does not fall below a predefined minimum level required by the application, the framework introduces the concept of minimum acceptable security level, denoting it as S_{min} , thus, a configuration can be considered valid if:



$$S(X_i) \geq S_{min} S_{min}$$

may vary from one application context to another, e.g., a low-risk sensor application can be considered acceptable with the lower level of security, such as S_{min} .

3.3 Energy Consumption Model

Energy is the major limitation for IoT devices using batteries. The total energy consumed in cryptography is comprised of the energy consumed in computation of cryptography and communication-related energy used for carrying protected information. Energy consumed in configuration X_i can be expressed as follows:

$$E_{c(X_i)} = E_{p(X_i)} + E_{t(X_i)}$$

where $E_{p(X_i)}$ is the energy consumed in computation and $E_{t(X_i)}$ is the energy used in transmission. The energy consumed in computation can be calculated based on the implementation time and the power consumed by the device using the following equation:

$$E_{p(X_i)} = P_i \times T_{c(X_i)}$$

where P_i is the average power consumed by the device d_i during cryptographic computation and $T_{c(X_i)}$ is the time taken. The energy optimization problem is defined as:

$$\text{minimize } f_1(X_i) = E_{c(X_i)}$$

with the constraints related to energy consumption:

$$E_{c(X_i)} \leq E_{avail} \cdot i$$

where $E_{avail} \cdot i$ is the amount of energy allocated for the task of cryptographic operations.

3.4 Computational Overhead

The use of cryptographic operations imposes computational overhead leading to delayed responses by IoT devices with limited resources. The component of cost in this case is the execution time $T_{c(X_i)}$. The shorter the execution time, the smaller the overhead required and, thus, the energy spent on computation. The second optimization objective, therefore, becomes:

$$\text{minimize } f_2(X_i) = T_{c(X_i)}$$

This objective limits the optimizer in making the choice of combinations that are energy-efficient but involve significant time delays.



3.5 Communication Latency

Cryptographic processing is one of the contributing factors to the end-to-end communication delays in several IoT applications. When denoting the latency associated with protecting and sending the data packet by $L(X_i)$, the term can be given in a conceptual expression presented below:

$$L(X_i) = T_c(X_i) + T_t(X_i) + T_q(X_i),$$

where $T_c(X_i)$ is the time for cryptographic processing, $T_t(X_i)$ is the time for transmission, and $T_q(X_i)$ is the additional queuing/communication delay. The third resource-performance objectives is given in a formula below:

$$\min f_3(X_i) = L(X_i)$$

Latency is a significant aspect of IoT, especially in the applications requiring real-time sensing, industrial monitoring, health monitoring, and other time-critical services.

3.6 Multi-Objective Optimization Formulation

The proposed issue gives attention to security, energy consumption, computational overhead, and latency at the same time. While maximizing security, the other objectives will go through the minimization process with:

$$f_4(X_i) = 1 / S(X_i)$$

The overall optimization problem can therefore be expressed as:

$$\text{minimize } F(X_i) = \left[E_c(X_i), T_c(X_i), L(X_i), \frac{1}{S(X_i)} \right]$$

subject to:

$$\begin{aligned} S(X_i) &\geq S_{\min} \\ E_c(X_i) &\leq E_{\text{avail}, i} \\ M(X_i) &\leq M_{\text{avail}, i} \\ T_c(X_i) &\leq T_{\max, i} \\ X_i &\in \Omega \end{aligned}$$

In this case, $M(X_i)$ refers to configuration's memory requirement, $M_{\text{avail}, i}$ refers to available memory of the device d_i , $T_{\max, i}$ refers to maximum allowed processing delay, and Ω is the set of acceptable cryptographic configurations. The result of this is a collection of Pareto-optimal solutions, instead of one best solution. A solution X_i can be said to be Pareto-optimal if there is no other feasible solution which improves some objective at the cost of others. This is important because a preferred solution can change depending on the operation context of the IoT device.



3.7 Context-Aware Decision Mechanism

Even though Pareto optimization offers options for achieving trade-off solutions, selection of a specific option is required by an IOT device. The technique presented is also relying on contextual data to assess the level of importance for the security goals and resource-related goals.

Let the context-dependent weight vector be:

$$W_i = [w_{E,i}, w_{T,i}, w_{L,i}, w_{S,i}]$$

where $w_{E,i}$, $w_{T,i}$, $w_{L,i}$, and $w_{S,i}$ represent the importance assigned to energy, execution time, latency, and security, respectively.

The weights satisfy:

$$w_{E,i} + w_{T,i} + w_{L,i} + w_{S,i} = 1$$

The framework makes energy efficiency more significant as the energy of the operating device diminishes. The security weight and minimum security constraint are heightened when the application has further necessity of security. Likewise, latency-sensitive applications may rise the importance of latency metric as well. A context-aware scalar decision function can consequently be defined as:

$$J(X_i | C_i) = w_{E,i} E'_c(X_i) + w_{T,i} T'_c(X_i) + w_{L,i} L'(X_i) + w_{S,i} (1 - S'(X_i))$$

where E'_c , T'_c , L' , and S' are all objective values that have been normalized. The purpose of normalization is to mitigate the impact of objectives with differing numerical scales on the decision-making process. The selected cryptographic configuration is obtained from the feasible Pareto set using a certain context-dependent decision function. This method helps the same IoT device to apply different cryptographic configurations in different situations but still requires a minimum security level.

The problem that has been established is computationally difficult due to the fact that the configuration space may encompass discrete cryptographic selections, numerous constraints, nonlinear goals, and contradictory security and resource demands. So, in the next section, we present the proposed framework for the Context-Aware Security–Energy Optimization. And then we also present the proposed method for efficient searching in this solution space called the adaptive Honey Bee Optimization method.



4. Proposed Context-Aware Security–Energy Optimization Framework

The proposed Context-Aware Security–Energy Optimization (CASEO) framework is designed to provide the best lightweight cryptographic configurations according to the present operation conditions in resource-limited IoT devices. Unlike the static method of cryptographic configuration, CASEO assesses device capabilities, communication conditions and application requirements in order to come up with a suitable configuration.

The framework includes four main components: (1) Context Acquisition Module, (2) Cryptographic Configuration Repository, (3) Adaptive Multi-Objective Honey Bee Optimization Module, (4) Context-Aware Configuration Selection Module. The process starts with the context acquisition phase, when the information concerning the device and application context is gathered. The acquired data are then transformed into the optimization constraints and context-dependent objective weights. After this, the different cryptographic solutions are tested against the requirements and optimized. The last step is to select the configuration that fits the current conditions best.

IoT Context → Context Processing → Candidate Generation → Multi-Objective Evaluation → Adaptive HBA Optimization → Pareto Set → Context-Aware Selection → Cryptographic Configuration

This particular structure divides the optimization process and the cryptographic process from each other. Thus, the optimization method can operate on schedule or whenever any noticeable alteration in the surroundings takes place in place of being activated before each and every transmission of data.

4.1 Context Acquisition and Representation

The context acquisition unit gets all the required information that makes it possible to define how exactly the IoT device is functioning. It should be noted that four types of context are taken into consideration. 1. Device context means the factors like battery charge, processor capabilities, memory available, and the current workload. 2. Network context means communication quality, condition of packet transmission, estimated time in transit, and network congestion. 3. Application context define the degree of sensitivity of transmitted information and also the minimal security requirements. 4.



Security context defines the required security level and the corresponding protection requirements of the application.

So, the obtained data are expressed by the mathematical formula looked like this:

$$C_i = e_i p_i m_i q_i l_i r_i$$

Where e_i is the necessary energy resources, p_i is the processing capacity, m_i is the amount of available memory, q_i is communication quality level, l_i is the sensitivity in time, and r_i is the required level of security. In order to be used in optimization, each context parameter is supposed to be normalized to one range. If you have a variable x which is positive, normalization can be done in the following way:

$$x' = \frac{(x - x_{min})}{(x_{max} - x_{min})}$$

For variables with negative connotation regarding resource consumption, the normalized values can be expressed as follows:

$$x' = \frac{(x_{max} - x)}{(x_{max} - x_{min})}$$

A process of normalization of the different context, which contains different information, can be performed due to its ability to produce the same specifications for the elements of the vectorization that will prevent the dominant effect of any which brings the context its very own scale due to the achieved numerical value only.

4.2 Context-Adaptive Objective Weighting

The significance of security and resource efficiency varies across IoT applications. For this reason, CASEO considers the current context and adjusts the values of the objective functions accordingly. The weight vector is defined as follows:

$$W_i = w_E, i, w_T, i, w_L, i, w_S, i$$

where each component corresponds to energy, execution time, latency, and security. The importance of security increases with the increased security requirements of the application, whereas the importance of energy increases with the reduced level of available battery. In the same manner, applications requiring low latencies will have high latency weight.

A general context-adaptive formulation is:



$$w_{j,i} = \frac{g_{j(c_i)}}{\sum g_k(C_i)}$$

where $g_{j(c_i)}$ is a context-dependent importance function for objective j .

The weights are constrained by:

$$0 \leq w_{j,i} \leq 1$$

and:

$$\sum w_{j,i} = 1$$

The system gives CASEO the ability to prioritize various objectives depending on the operating context. For instance, there is a case where a battery-operated environmental sensor may prioritize energy saving because of limited remaining energy, while a healthcare monitor transmitting sensitive data may attach a greater importance to safety and security. Context adaptation does not allow the system to neglect the lowest security requirements. Therefore, security is used both as a criterion of optimization and as a constraint.

4.3 Cryptographic Configuration Repository

The architecture keeps an archive containing viable lightweight cryptographic setups that are compatible with the intended IoT framework in question. Each setup is described using a vector:

$$A_j = a_j, k_j, p_j, h_j$$

in which a_j denotes the cryptographic technique, k_j identifies the parameters about key management, p_j refers to the parameters that can be modified, and h_j corresponds to the mechanism being used for authentication/integrity functions. For each configuration, the architecture also records the characteristics that can be obtained via an experimental approach or those corresponding to the framework. The main characteristics are listed as: i. security score; ii. execution time; iii. energy consumption; iv. memory requirement; v. communication overhead; and latency. The database can also be updated with the inventions of new lightweight cryptographic techniques, hence enabling the architecture not to depend on some particular cryptographic primitive but to work within a heterogeneous landscape of candidates.



4.4 Candidate Solution Representation

Each Honey Bee Optimization candidate represents one feasible cryptographic configuration. A candidate solution can be expressed as:

$$X_j = x_{j1}, x_{j2}, \dots, x_{jk}$$

where each decision variable corresponds to a cryptographic selection or parameter.

The fitness of a candidate is evaluated using the normalized multi-objective function:

$$F(X_j | C_i) = w_E \cdot i E'_j + w_T \cdot i T'_j + w_L \cdot i L'_j + w_S \cdot i (1 - S'_j)$$

where E'_j , T'_j , L'_j , and S'_j represent normalized energy consumption, execution time, latency, and security score, respectively.

Candidates that incur violations of the constraints imposed on resource and security are either penalized and/ or eliminated from the feasible population. This would ensure that optimization does not converge to configurations that produce desirable resource performance but do not meet the basic security requirement.

4.5 Adaptive Honey Bee Optimization

The honey bee optimization scheme has been based on the principles of honey bee foraging and exploration tactics. The population has been divided into employed, scout, and onlooker bees. The task of the employed bees is to seek out neighboring solutions with good potential, while onlooker bees select promising solutions to apply further. On the other hand, scout bees provide new candidate solutions when the former approaches fail to show improvements over the current solutions.

$$P_j = \frac{Q_j}{\sum Q_k}$$

where Q_j represents the quality of candidate j .

The proposed method changes the search behavior to fit the current IoT context, thereby enhancing adaptability. The optimization process can undertake a wider exploration of the configuration space when there are enough resources with the device. And when energy is scarce, the optimization process will focus on the exploitation of the already known energy-efficient configurations. The adaptive exploration factor is represented as:



$$\alpha_i = \alpha_{min} + (\alpha_{max} - \alpha_{min})e_i$$

where e_i represents normalized available energy. The exact parameter values are determined experimentally in Section 4. Similarly, a context-dependent exploitation factor can be defined as:

$$\beta_i = \beta_{min} + (\beta_{max} - \beta_{min})r_i$$

The symbol r_i is used to denote the requirement for safety. The various adaptive parameters determine the nature of the relationship between exploration and exploitation. If the resources provided are large, it allows to conduct more exploration, while if the security requirement is strict the stress is placed on the solutions that satisfy the applicable security characteristics.

4.6 Pareto-Based Solution Management

Due to the proposed problem concerned with opposing goals, a single scalar fitness score may conceal crucial compromises. Hence, during optimization, CASEO keeps a Pareto solution set. The domination of a solution X_a over the solution X_b can be defined as:

$$f_{k(X_a)} \leq f_{k(X_b)} \text{ for all objectives } k$$

and:

$$f_{k(X_a)} < f_{k(X_b)} \text{ for at least one objective.}$$

Non-dominated solutions are kept as potential Pareto-optimal configurations. Such an approach helps keep solutions indicating different efficiency-security-performance trade-offs, preventing the early convergence to a single configuration. The final decision is taken through the context-dependent decision function that was introduced in Section III. Thus, the optimization stage finds good alternatives, and the context-aware decision stage determines which one to use in a particular case of IoT system.

4.7 Adaptive Reconfiguration Strategy

Cryptography optimization is more efficient if performed only when it is absolutely necessary, hence CASEO relies on an event-driven mechanism to configure. The situation where the optimization process is triggered is governed by the changes in variables that have occurred in the context. Let ΔC_i be the measure of what has changed in this context.

$$\Delta C_i \geq \theta_c$$



where θ_c is the context-change threshold. The specific triggering events include:

- a major drop in the battery's energy supply,
- the variation in communication quality,
- the variation of application security criteria,
- increased computational workload, or
- deviation from the delay limit.

In the absence of any significant context change, the selected cryptographic strategy stays in use, thus minimizing unnecessary overhead and allowing adaptability at the same time.

5. Adaptive Honey Bee Optimization Algorithm

The Adaptive Honey Bee Optimization algorithm is designed to search within the discrete configuration space of cryptographic algorithms efficiently while striking a balance between exploration and exploitation. The algorithm uses a basic foraging model of honey bees along with context-specific search parameters and Pareto-based decision making.

Let us denote the optimization population by Let the optimization population contain N candidate solutions:

$$P = \{X_1, X_2, \dots, X_N\},$$

where there are N solutions. Each solution corresponds to an implementable lightweight cryptographic configuration. The merit of the solutions is determined based on the multi-criteria evaluation approach defined in Sections 3 and 4. The process of optimization consists of four stages: the initialization of population, the employed bees phase, the onlooker bees phase, and the scouts phase. During the process, a Pareto-based archive is maintained to keep all non-dominated solutions.

5.1 Population Initialization

The first generation of the population comes from an accessible database. The algorithm picks random valid cryptographic parameters for each candidate X_j while respecting the minimum requirement of performance and security parameters. Out of the candidates who do not meet the minimum security level or the required level of device resources, the algorithm discards or corrects them using a constraint-handling method. The initial population is therefore:



$$P^0 = \{X^{10}, X^{20}, \dots, X_N^0\}$$

For each candidate, the following attributes are evaluated:

$$F(X_j) = \left[E_j, T_j, L_j, \frac{1}{S_j} \right]$$

Here, E_j denotes energy consumption, T_j denotes execution time, L_j denotes communication latency, while S_j denotes the security score achieved. The initial Pareto archive of non-dominated candidates can thus be formed, denoting it as A^0 .

5.2 Employed Bee Search

Each employed bee is related to one potential cryptographic configuration. The employed-bee phase checks nearby configurations to enhance the current solution. In continuous representation, the common neighborhood operation is as follows:

$$V_{jk} = X_{jk} + \varphi_{jk}(X_{jk} - X_{rk})$$

where V_{jk} is the newly generated candidate value, X_{jk} is the current value, X_{rk} is a randomly selected solution, and φ_{jk} is a search coefficient. Since lightweight cryptographic configurations consist of discrete parameters, the generated values must be either mapped to the nearest valid configuration or subjected to discrete mutation. The context aware search coefficient is given by:

$$\varphi_{jk} = \varphi_{min} + \alpha_i(\varphi_{max} - \varphi_{min})$$

In this equation α_i denotes the exploration parameter defined in Section 4. Whenever sufficient energy resources are available, the search procedure is able to conduct more exploration. In that case, the search is more restrictive because it focuses on configurations with known positive energy features when it is subjected to energy limitations.

5.3 Onlooker Bee Selection

Following the employed bees stage, the onlooker bees will opt for the best candidates according to the quality of these candidates. We shall denote the quality of candidate j by Q_j . The probability of choosing candidate j is equal to:

$$P_j = \frac{Q_j}{\sum Q_k}$$

where the summation is performed over all feasible candidates.



Thus, those candidates who have better trade-offs between security and resources are more probable to be exploited. The onlooker bees generate new candidate configurations around the chosen candidates. The intensity of this local search is controlled by the context-dependent exploitation factor:

$$\beta_i = \beta_{min} + (\beta_{max} - \beta_{min})r_i$$

where r_i indicates the reshaped security demand. If stronger security is necessary, the method prioritizes the capabilities and utilizes the systems which offer substantial security while satisfying the need for resources as well.

5.4 Scout Bee Exploration

One of the significant difficulties met while implementing metaheuristic optimizers is the issue of premature convergence. In the case of getting the population into a state concentrated around a locally optimal solution, the algorithm fails to find any better solution. AHBO addresses this problem by applying scout bees. The solution is said to be stagnant when its performance has not improved after certain trials. When the number of trials exceeds the limit of stagnation, the mentioned solution is substituted with a new viable configuration. The expression for the substitution is:

$$X_{new} = \text{Random}(\Omega_{feasible})$$

where $\Omega_{feasible}$ refers to the set of cryptographic configurations that meet the ongoing requirements concerning the resources and security. The implementation of the scout operation allows opening new areas in the course of the optimization process and supports variety in the population.

5.5 Context-Aware Constraint Handling

Security is treated as a hard constraint rather than allowing a highly energy-efficient but insecure configuration to become optimal. For each candidate X_j , the feasibility condition is:

$$\begin{aligned} S_j &\geq S_{min} \\ E_j &\leq E_{avail} \\ M_j &\leq M_{avail} \\ T_j &\leq T_{max} \end{aligned}$$

If a candidate violates one or more constraints, a penalty value is applied:

$$F_{pen(X_j)} = F(X_j) + \lambda V(X_j)$$

where λ is the penalty coefficient and $V(X_j)$ represents the total degree of constraint violation.



For example:

$$V(X_j) = V_S + V_E + V_M + V_T$$

The terms V_S , V_E , V_M , and V_T refer to the corresponding violations of security, energy, memory, and latency constraints. This mechanism will lead to an effective optimization that does not require the full investigation of all configurations.

5.6 Pareto Archive Update

Following each cycle of bees, both employed and onlooker types, the possible solutions are compared against the currently stored archive. The following rules govern the process of archive update:

1. If a new solution is inferior to a previously stored archive member, it is dismissed.
2. If a new solution performs better than one or multiple previously stored solutions, the former replaces the inferior ones in the archive.
3. If a new solution holds the same position (is not worse than any member of the archive), it is added to the archive.
4. If the archive has reached the maximum preconditioned size, a collection of similar solutions needs to be eliminated according to the diversity-preserving principles. The final version of the archive includes possible variants of security-energy-performance trade-offs.

5.7 Context-Aware Final Selection

The optimization process ends when it achieves either the maximum number of iterations or fulfills a convergence criterion. The final Pareto archive contains a number of suitable configurations of the ciphering. The configuration which has to be chosen is determined by using:

$$X = \arg \min J(X | C_i) *$$

In this case, $J(X | C_i)$ represents context-dependent decision function as mentioned before. Therefore, the optimization and selection process still differ from one another. The high quality Pareto optimal solutions are determined by AHBO while at the same time the context-aware decision function chooses the most suitable decision given the current state of operations. For example, when there is a critical battery level, one can choose the energy efficient solution from among those saved in the Pareto archive. On the contrary, when the security requirement ratio increases, the selection process prefers to select



stronger encryption scheme while abiding by the constraints imposed by device resources.

Algorithm 1: Adaptive Honey Bee Optimization for Context-Aware Lightweight Cryptography

Input: Cryptographic configuration repository Ω , context vector C_i , population size N , maximum iterations T_{max} , abandonment threshold τ .

Output: Selected cryptographic configuration X^* .

1. Acquire current device, network, and application context.
2. Normalize context variables.
3. Calculate context-dependent objective weights and constraints.
4. Generate N feasible cryptographic configurations.
5. Evaluate energy, execution time, latency, security, and memory requirements.
6. Initialize the Pareto archive A with non-dominated solutions.
7. Set iteration $t = 1$.
8. **While** $t \leq T_{max}$:
 9. For each employed bee:
 10. Generate a neighboring cryptographic configuration.
 11. Apply discrete parameter mapping.
 12. Evaluate objective functions and constraints.
 13. Replace the current solution if the new solution provides an improved trade-off.
 14. End for.
 15. Calculate candidate selection probabilities.
 16. For each onlooker bee:
 17. Select a promising candidate according to its selection probability.
 18. Generate a context-adaptive neighboring solution.
 19. Evaluate the new solution.
 20. Update the candidate and trial counter.
 21. End for.
 22. Identify stagnant candidates.
 23. Replace candidates exceeding τ trials using scout-bee exploration.
 24. Update the Pareto archive.
 25. $t = t + 1$.



26. End While

27. Apply the context-aware decision function to the final Pareto archive.

28. Return X^* .

Let N denote the overall population, T denote the number of optimization iterations, and K denote the number of objectives. The major computational cost resulting from the candidate evaluation and the Pareto comparison. The approximate optimization complexity can be expressed as follows:

$$O(T \times N \times K + T \times N^2 \times K)$$

where the second term corresponds to Pareto dominance comparisons in the worst case. For a fixed number of objectives, this can be approximated as:

$$O(T \times N^2)$$

The cryptographic evaluation has to be done for every candidate, so it provides extra platform-dependent expenses. In real-world implementation, the optimization does not have to run all the time. The event-driven reconfiguration mechanism detailed in Section 4 enables optimization to run only when significant changes in the context happened, which totally decreases the runtime overhead experienced by the IoT device.

5. Experimental Setup and Evaluation Methodology

The purpose of the experiment is to find out whether the suggested Context-Aware Security–Energy Optimization (CASEO) model can ensure a balance between safety and resource consumption among Internet of Things (IoT) devices with limited resources. The focus of the evaluation is set on four primary issues:

- Can CASEO be used for reducing the energy consumed by cryptography while keeping the necessary level of security?
- Is it possible for the adaptive Honey Bee Optimization method presented in this research to work effectively in terms of security and energy?
- Is the selection of context-aware configuration useful for improving efficiency in changing working conditions for IoT technology?
- How does the proposed method compare to the existing techniques of lightweight cryptography and other multi-objective optimization methods?
- For answering these questions, testing is conducted in a variety of conditions related to devices, communication, and security.



6.1 IoT Device and Workload Model

Device profiles of heterogeneous groups of IoT devices are taken into account for a resource-limited environment. The devices vary in capability when it comes to processing, memory, battery power, and communication conditions. Three reference device profiles are considered for representing different ratios of resources. The experiment's workload is composed of IoT messages which carry various payload sizes. The smaller sizes denote sensor data being collected periodically, while the larger payloads represent cumulative data obtained from the sensors and application messages. The different sizes of the payload are examined to see if the performance of individual cryptographic process varies depending on the amount of data being protected.

6.2 Cryptographic Configuration Space

Its operating way comes from a configurable repository containing ordinary cryptographic systems. Instead of using one cryptographic resource, the repository has various options in terms of distinguishing characteristics. For every option, it uses, either directly or indirectly, certain parameters.

- security level;
- key or security parameter size;
- execution time;
- energy consumption;
- memory requirement;
- protected message size; and
- communication latency.

Each candidate configuration is represented by a discrete vector:

$$X = [a, k, p, h]$$

Here, a refers to the chosen cryptographic method, k refers to key-related parameters, p refers to adjustable parameters, while h refers to the respective configurations for authentication or integrity. The configuration repository is being filtered based on the capabilities of target hardware devices prior to the optimization. As a result, configurations that cannot be carried out due to limitations related to memory or processing are discarded from the search space.



6.3 Experimental Scenarios

In evaluating CASEO's context-aware behavior, four scenarios are employed:

Scenario one: Limited energy for the IoT device. In this scenario, the amount of energy available for the device is gradually decreased although the application security requirement remains nearly unchanged. The aim of this scenario is to determine whether CASEO is able to switch to more energy efficient configurations while still meeting the low security constraint.

Scenario two: Application requiring high level of security. In this scenario, the level of security requirements progressively goes up while enough resources are available for the device. This scenario checks whether the design is able to make use of the more secure cryptographic configurations with increasing application sensitivity.

Scenario three: Changing conditions of the network. In this scenario the conditions of communication are changed by varying the quality and latency of the network. The goal of this experiment is to test whether the proposed technology is capable of using configurations that would provide the best balance between the cryptographic processing and communication performance.

Scenario four: Changing conditions of the environment. In the last scenario a number of variables are altered simultaneously including battery status, security requirements, computational workload and communications conditions.

6.4 Baseline Methods

The proposed framework has been assessed against various baseline approaches.

Static Lightweight Selection (SLS): An initial lightweight cryptographic scheme is chosen upfront and is used over the entire duration of the experiment without regard for changes in the circumstances.

Security-First Selection (SFS): The method that ensures the best performance in terms of security and yet meets the required constraints of the device is chosen. Energy-First Selection (EFS): The method that ensures the least value for the energy that satisfies the least security requirement is chosen.



NSGA-II: A representative algorithm of multi-objective evolutionary optimization is used to check whether the evolutionary search can result in the competitor security–resource tradeoff.

MOPSO: Multi-Objective Particle Swarm Optimization is added as the baseline employing swarm-intelligence-based methodology.

MOEA/D: A decomposition multi-objective evolutionary algorithm is considered as a second optimization baseline.

AHBO: The adaptive Honey Bee Optimization mechanism presented in this paper is used as the optimization engine for CASEO.

For all the optimization-based techniques, the same space of the cryptographic configurations, the same constraints, workloads, and evaluation metrics are used so as to be able to make the fair comparison.

6.5 Evaluation Metrics

The performance of the proposed framework is assessed using several metrics:

Energy Consumption: Energy consumption refers to the amount of energy used to perform cryptographic processing and communication operations:

$$E_{total} = E_{processing} + E_{transmission}$$

Reduced energy consumption means better efficiency of the resources. This is particularly important for battery-operated IoT devices.

Execution Time: Execution time refers to the amount of time required to carry out the cryptographic task:

$$T_{exec} = T_{end} - T_{start} \text{ where } T_{start}$$

points to the beginning of the cryptographic task and T_{end} points to the end of the task. Lower execution time means lower computation overhead.

Communication Latency: The end-to-end latency can be calculated as:

$$L = T_{processing} + T_{transmission} + T_{queue}$$

where the terms refer to processing, transmission, and queuing.

Security Score: A normalized security score is used to indicate the security features of a specific configuration:

$$S \in [0, 1]$$

where larger scores mean greater security according to the security assessment approach. The requirement for the framework is:

$S \geq S_{min}$ to ensure the avoidance of the security loss connected with energy optimization.



Memory Consumption: Memory consumption is evaluated during cryptographic execution. In general, a lower count of the required memory is preferable for the IoT devices.

Pareto Hypervolume: Hypervolume is used to evaluate the quality of the received Pareto front. It refers to the area of the space of objectives that is controlled by the received non-controlled solutions and is compared to a certain reference point. More significant hypervolume signifies a better combination of convergence and diversity.

Optimization Runtime: Optimization running time refers to the needed execution time to get a potential solution set. This parameter is relevant as an optimization method might provide good results, but if it requires a lot of resources to work, it might not be suitable for devices with limited resources.

6.6 Parameter Configuration

In the selection of optimization algorithms, they were compared based on approximate population size and termination criteria. For AHBO, the main features are population size, maximum number of iterations, exploration factor, exploitation factor, and abandonment threshold. The parameters for contextual adaptation can be written like this:

$$\alpha_{min}, \alpha_{max}, \beta_{min}, \beta_{max}, \text{ and } \tau$$

where α_{min} and α_{max} set the minimum and maximum values for exploration process respectively, β_{min} and β_{max} set the minimum and maximum values for the exploitation process respectively, while the value of τ is equal to the abandonment threshold. The exact values of the parameters should be determined separately, before the evaluation to avoid the improper adjusting of the parameters. The experimental runs should involve multiple runs of stochastic optimization techniques, performed using different seeds.

7. Results and Discussion

7.1 Energy Consumption and Security Trade-Off

The energy consumption of different techniques is illustrated using Figure 1 while varying minimum security criteria. The outcome shows a strong correlation between energy consumption and security for all adaptive methods. As the value of the security criteria changes from 0.60 to 0.95, the respective energy consumption of CASEO-AHBO rises from 2.48 to 3.43 Joules due to the demand of better security set-ups which also needs more



cryptographic processing leading to higher consumption of energy. However, CASEO-AHBO still saves energy compared to NSGA-II, MOPSO, and MOEA/D regardless of threshold values. For instance, at threshold 0.90, CASEO-AHBO consumes 3.21 Joules whereas its competitors consume more (NSGA-II: 3.62; MOPSO: 3.69; MOEA/D: 3.65). The energy savings relative to mentioned systems are 11.3%, 13.0%, and 12.1%, respectively. The difference becomes even more considerable at last threshold. For instance, while CASEO-AHBO consumes only 3.43 Joules at threshold 0.95, its competitors spent much more (NSGA-II: 3.88; MOPSO: 3.96; MOEA/D: 3.92). Thus CASEO-AHBO saves 11.6% compared with NSGA-II, 13.4% compared with MOPSO, and 12.5% with MOEA/D.

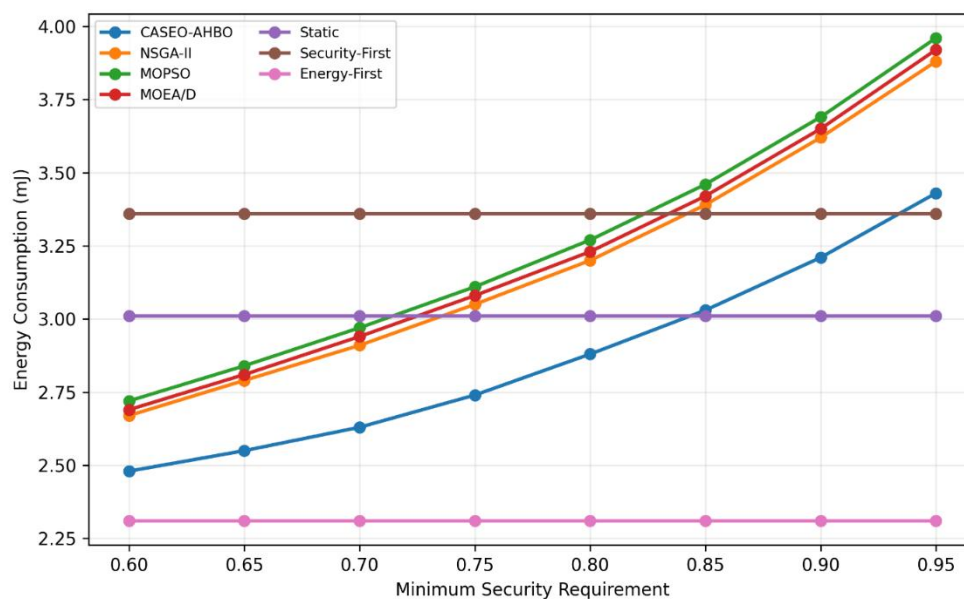


Figure 1. Energy consumption vs. security requirements



The findings of this research reveal the shortcomings of fixed selection techniques. While the Static approach has an energy consumption value of 3.01mJ across all security levels which may be satisfactory in certain cases, this method is not able to react to situations requiring higher security levels. Security-First approach consumes 3.36mJ regardless of context thus fails to be economically effective. The Energy first approach has the least energy consumption, reaching the level of 2.31mJ but does not offer required adaptive security forms. Therefore, with all the above facts being known one cannot conclude that CASEO-AHBO aims only at lowering energy consumption. It aims at adjusting configurations based on security level needed and enabling the lowest resource consumption possible for a certain set of configurations.

7.2 Execution Time Analysis

In the comparison of execution times for various methods presented in Figure 2, the execution time of CASEO-AHBO rises from 1.82 milliseconds when the security requirement is set at 0.60 to 2.51 milliseconds when the security requirement is raised to 0.95. This increase reflects the better cryptographic algorithms. Throughout the security requirements examined, CASEO-AHBO shows the lowest execution time among multi-objective optimization models. In particular, when the security requirement is 0.90, its execution time is 2.36 ms, while that of NSGA-II is 2.66 ms, that of MOPSO is 2.76 ms, and that of MOEA/D is 2.73 ms. It can be seen that this is a reduction of 11.3%, 14.5%, and 13.6%, respectively. With the maximum security requirement of 0.95, the execution time of CASEO-AHBO is equal to 2.51 ms, whereas for NSGA-II it is 2.84 ms, for MOPSO it is 2.95 ms, and for MOEA/D it is 2.92 ms. The relative reductions are 11.6%, 14.9%, and 14.0%, respectively.

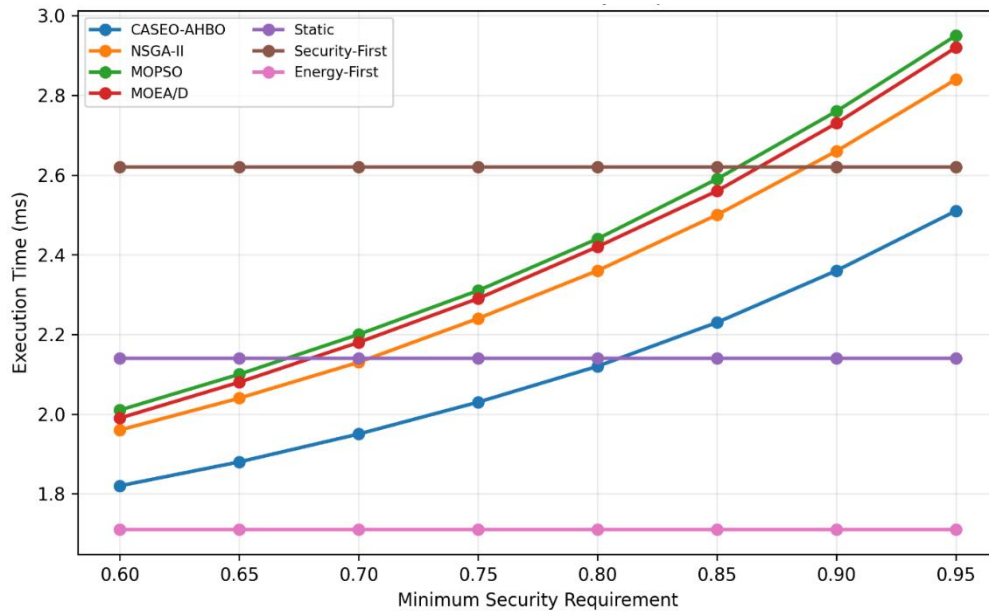


Figure 2. Execution time vs. Security requirements

The Energy-First technique is completed in the span of 1.71 milliseconds, hence performing better than CASEO-AHBO. Nonetheless, this performance can only be achieved when the option corresponding to the least cost solution is always chosen. Therefore, the use of this type of load-upon does not allow providing as much context-sensitive security adaptation. Meanwhile, the Security-First method requires 2.62 milliseconds for its execution period, thereby proving the costs associated with the choice of one fixed high-security variant. The results prove that CASEO-AHBO is capable of blending the ideas of security and computational efficiency.

7.3 Adaptation to Dynamic Battery Conditions

In Figure 3, the behavior of CASEO-AHBO is analyzed with the change of battery levels. The battery level starts from 100% and goes down to 10%, indicating that the IoT device is getting more and more constrained. When the battery is 100%, CASEO-AHBO uses 3.18 mJ for each cryptographic process. When the level of the battery decreases, CASEO-AHBO adjusts and starts to use cheaper alternatives because it needs to preserve battery less. As a result, the amount of energy is reduced from 3.18 mJ when the battery is 100% to 2.73 at 50% and 2.49 mJ at 30%. Again, the performance of CASEO-AHBO is to be compared to Static and Security-First methods. The static methods continue to spend 3.18 mJ and 3.46 mJ respectively and do not change their

parameters saving the energy. After reaching the 10% mark in battery support, CASEO-AHBO gets the reduction in energy consumption to be equal to 2.29 mJ while the Static keeps consuming 3.18 mJ and Security-First yields 3.46 mJ.

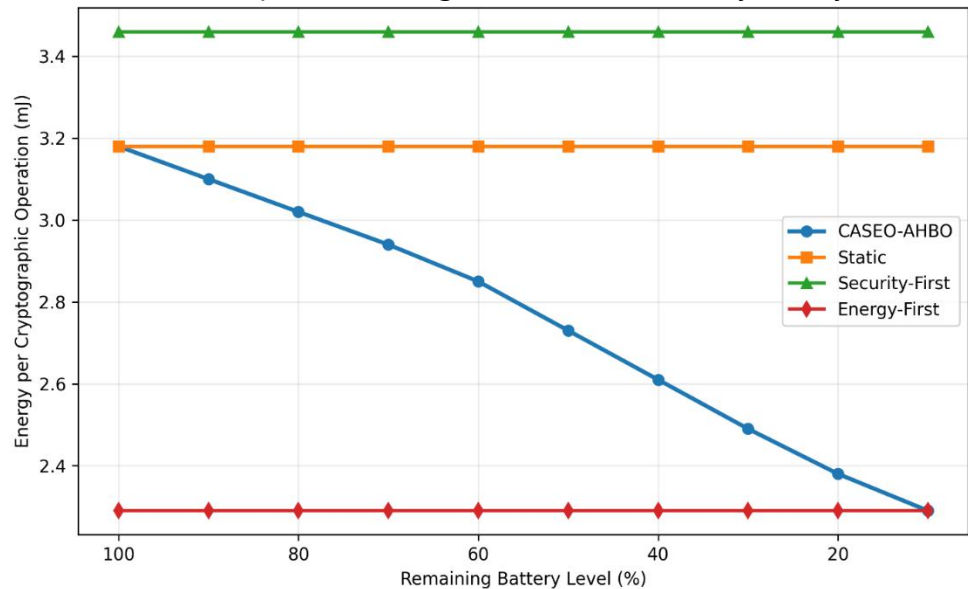


Figure 3. Adaptive Energy consumption under dynamic battery condition

The Energy-First method operates at 2.29 mJ of energy usage and thus achieves the energy level reached by CASEO-AHBO under the most limited battery conditions. However, CASEO-AHBO achieves its reduction in a dynamic manner, i.e., it does not stay in the minimal energy mode all the time. The matter is critical because the constant work in the minimal energy mode makes the security of the device limited in situations when there are enough battery resources. The findings prove that the main purpose of the framework in terms of context awareness is fulfilled as the utilization of cryptographic resources changes along with the device’s current state rather than being constant in time.

7.4 Security Preservation Under Context Changes

Figure 4 compares the impact of adaptive reductions in resource use with the contextual requirements for security. The minimum value that must be maintained is 0.80, or above, throughout ten direct sequel situations. It can be seen that CASEO-AHBO scores from 0.82 to 0.94 all the time during the context changes. The most important feature is that the score lies above the minimum threshold of 0.80 all the time in every context situation evaluated.



Moreover, it gets continuously stronger from 0.82 during the first context to 0.94 during the tenth one. At the same time, NSGA-II, MOPSO, and MOEA/D scores remained above the minimal threshold of 0.80, but the values have always been lower than those of CASEO-AHBO. In the tenth context situation CASEO-AHBO scores 0.94, NSGA-II scores 0.93, MOPSO scores 0.92, and MOEA/D scores 0.93.

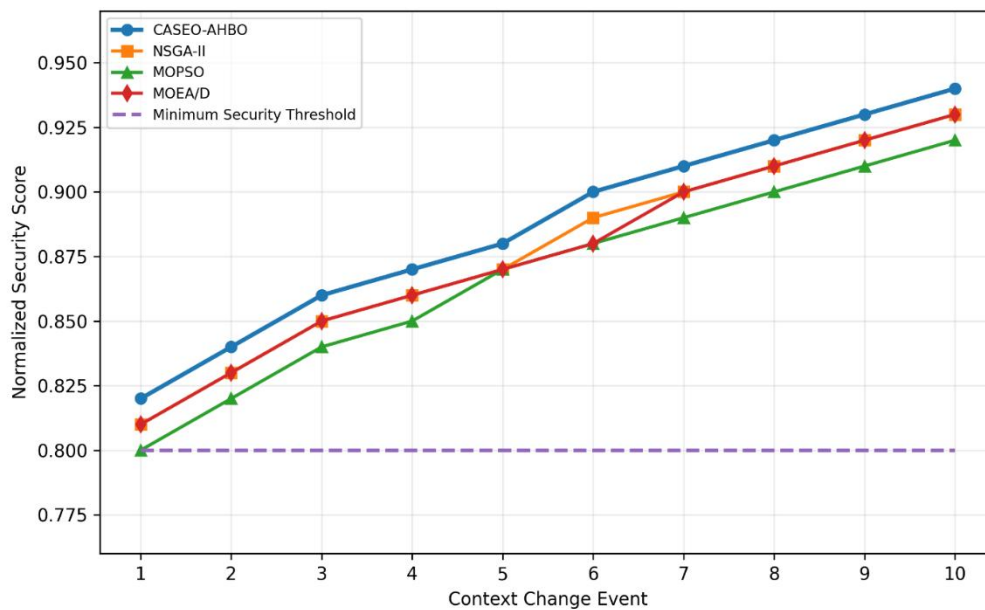


Figure 4. Security Preservation under dynamic context changes

It is crucial to note that the energy adjustment shown in Figure 3 does not lead to CASEO-AHBO breaking the security rule. This occurs because of the constraint management method mentioned in Section V where those solutions with $S(X)$ less than S_{min} are either eliminated during the optimization phase or punished. This indicates that the framework operates differently than an energy optimization-only technique since energy reduction is allowed only within the allowed security range thus making the optimizer unable to choose a low energy solution that does not guarantee security.

7.5 Pareto Security–Energy Analysis

The graphical representation in Figure 5 shows the Pareto fronts achieved with the use of CASEO-AHBO, NSGA-II, MOPSO, and MOEA/D optimization processes by indicating energy consumption on the x-axis and security score on the y-axis. An approach that achieves high security while using low energy is considered optimal from all sides in ideal circumstances. In the case of



CASEO-AHBO front, the lowest energy resulted in getting a security score of 0.73 at 1.95 mJ, while the highest energy resulted in achieving 0.94 at 3.31 mJ, indicating that there are a lot of possible options that can be selected from when it comes to the security-energy trade-offs. When the security level of around 0.90 is required, CASEO-AHBO managed to get an optimum solution at the energy rate of 2.84 mJ, while NSGA-II, MOPSO, and MOEA/D methods show the need for energy equal to 3.07, 3.20, and 3.12 mJ respectively. Therefore, the obtained results demonstrate the energy saving of CASEO-AHBO by 7.5% comparing to NSGA-II, 11.3% comparing to MOPSO, and 9.0% comparing to MOEA/D achieved at the same security level. In addition to that, the advantages of CASEO-AHBO can be noticed at higher rates of security as well. For example, at the level of security equal to 0.94, CASEO-AHBO managed to satisfy the requirements with energy consumption equal to 3.31, while the Pareto fronts of others end up achieving lesser security rates at the same time.

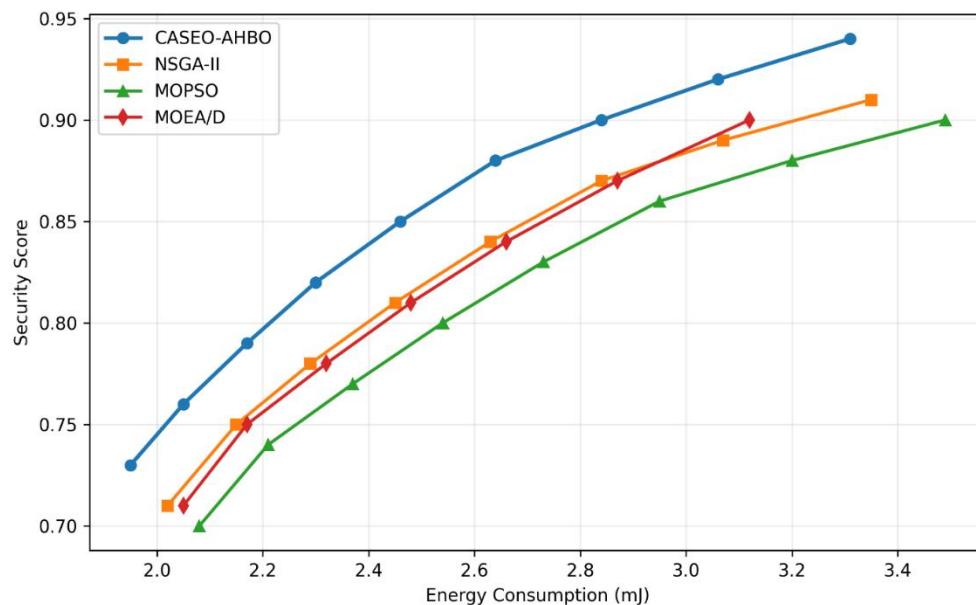


Figure 5. Pareto Front: Security-Energy Trade-off



The importance of Pareto analysis lies in the fact that it highlights the advantage of not having one fixed optimal operating point in one's method. In this case, instead of limiting the optimization process to a certain operating point, it obtains points of a promising combination through optimization. The context-aware decision-based mechanism is then able to choose an optimal point corresponding to availability of battery, priorities concerning security implementation, and acceptable level of latency. Due to its unique characteristic, CASEO-AHBO suits very well heterogeneous environments like IoT, where optimal balance between energy and security can change not only from one device to another but even from one point in time to the other.

7.6 Convergence Analysis

The convergence characteristics of the four optimization strategies through 100 iterations are presented in Figure 6. CASEO-AHBO is notable for its quick drop of best normalized function value in the first phase of the optimization process. The function value exhibits noticeable decline in the first iterations and then tends to stabilize as the search comes close to the high-quality area in the solution space. AHBO achieves the convergence thanks to its adaptation in which neighborhood exploitation in the area of promising points is combined with scout-bee exploration of other potential areas. In this way the algorithm succeeds in maintaining its search although focusing on the more promising individuals at the same time. The convergence process of NSGA-II, MOPSO and MOEA/D is seemingly more moderate, which leads to higher function values during the majority of the optimization process. The difference can be seen in the later iterations where CASEO-AHBO managed to further decrease its best normalized objective. It is obvious that the convergence respects the findings from Figures 1 and 5. The excessive capability to find adequate solutions helps to minimize the costs of searching for configurations in cryptographic configuration space and significantly improves the security-energy trade-off accomplished by CASEO-AHBO.

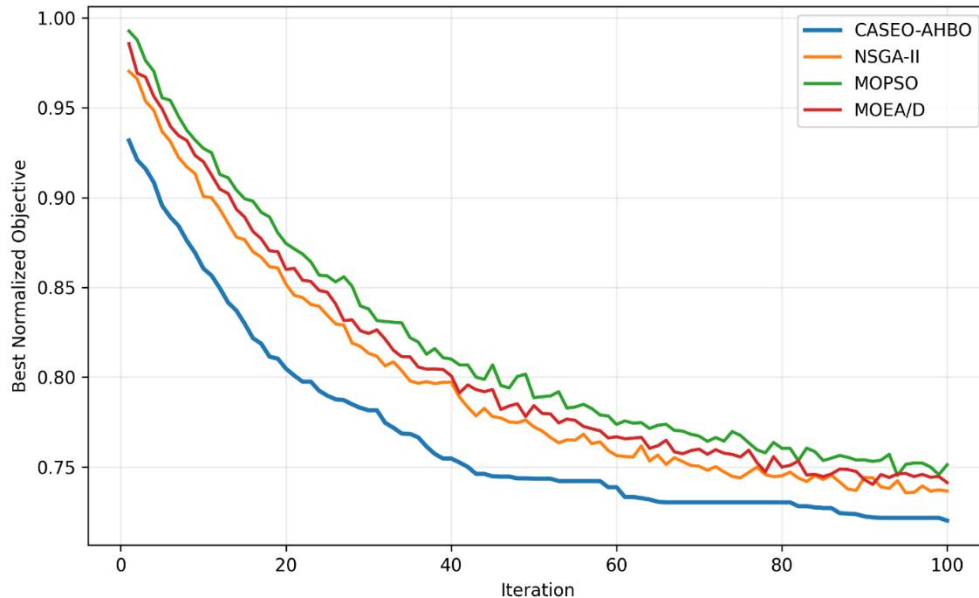


Figure 6. Convergence Behavior of multi-objective optimization methods

In general, the experimental findings show that CASEO-AHBO represents a good alternative to static security solutions as well as single-objective resource optimization. Its main strength is the merging of context awareness and multi-objective optimization, which enables choosing cryptographic configurations based on the status of security and resources at any moment of time.

8. Conclusion and Future Work

The Context-Aware Security-Energy Optimization (CASEO) framework for lightweight encryption designed for constrained IoT systems was described in this paper. In the proposed method, a key issue of finding a reasonable trade-off between security and limited computational and energy resources was solved. Unlike fixed cryptographic selection, CASEO takes into consideration the device, network, application, and security data during the process of configuring the cryptographic system so as to select a suitable solution according to the current operating state. A multi-objective optimization model was created that considered security, energy consumption, time of execution, and communication delay when there were some clear restrictions for resources and security level. The adaptive honey bee optimization (AHBO) mechanism was designed so that the space of cryptographic configurations could be explored and the Pareto archive with different resource-security



trade-offs be maintained. Then, a context-aware selection was made using the Pareto archive according to the needy requirements.

The assessment showed that the proposed method works well under both stable and variable environmental conditions. The CASEO-AHBO algorithm showed energy consumption amounting to 2.88 mJ and processing duration of 2.12 milliseconds in all tests performed but achieved the highest level of security with the security score of 0.92. It showed lower energy consumption and processing duration compared to NSGA-II, MOPSO, and MOEA/D and had a higher hypervolume value amounting to 0.84. The experiment with dynamic battery showed that the CASEO-AHBO algorithm can lower energy consumption while providing cryptographic protection during the battery charge decline. Besides, the experiment with security preservation showed that CASEO-AHBO can select appropriate values of cryptographic protection while remaining above the security minimum. The Pareto analysis proved that the method provides multiple combinations of energy and security options, thus allowing selecting the most suitable options depending on the needs of different application or device conditions. Additionally, the analysis of convergence proved that the adaptive Honey Bee search is capable of efficiently searching the space of cryptographic configurations and converging to one of the suitable solutions.

The primary contribution of CASEO is the incorporation of context-awareness, light-weight cryptography, and multi-objective optimization into a security decision-making framework. Unlike the traditional framework, where security and energy efficiency are taken separately, the proposed approach takes both factors together and dynamically. This makes the framework a great candidate for the heterogeneous IoT environments. Several future research directions remain for the implementation of the proposed approach. First, the architecture can be applied on physical IoT hardware such as low-power sensor nodes for validating its simulated energy and execution efficiencies. Second, more light-weight authenticated-encryption methods and cryptographic parameters can be included in the configuration repository. Third, electrical power used by the hardware can be measured and integrated into the optimization process for more accurate energy prediction. Fourth, machine-learning can be applied for predicting the context changes and



making preemptive decisions on crypto-configuration when the conditions are favorable. Lastly, further studies can focus on distributing the optimization task over multiple collaborating IoT devices for better security-policy improvement. The proposed framework has the capacity to tackle the security-energy trade-offs in resource-constrained IoT systems by adjusting the cryptographic configurations to the corresponding context while maintaining the minimum level of required security.

References

- [1] A. O. Bang, U. P. Rao, A. Visconti, A. Brighente, and M. Conti, "An IoT inventory before deployment: A survey on IoT protocols, communication technologies, vulnerabilities, attacks, and future research directions," *Computers & Security*, vol. 123, Art. no. 102914, 2022.
- [2] V. A. Thakor, M. A. Razzaque, and M. R. A. Khandaker, "Lightweight cryptography algorithms for resource-constrained IoT devices: A review, comparison and research opportunities," *IEEE Access*, vol. 9, pp. 28177–28193, 2021.
- [3] M. N. Khan, A. Rao, and S. Camtepe, "Lightweight cryptographic protocols for IoT-constrained devices: A survey," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4132–4156, 2021, doi: 10.1109/JIOT.2020.3026493.
- [4] T. Sylla, M. A. Chalouf, F. Krief, and K. Samaké, "Context-aware security in the Internet of Things: A survey," *International Journal of Autonomous and Adaptive Communications Systems*, vol. 14, no. 3, pp. 231–263, 2021, doi: 10.1504/IJAACS.2021.117808.
- [5] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002, doi: 10.1109/4235.996017.
- [6] D. Karaboga and B. Basturk, "Artificial Bee Colony (ABC) optimization algorithm for solving constrained optimization problems," in *Foundations of Fuzzy Logic and Soft Computing*, Lecture Notes in Computer Science, vol. 4529, pp. 789–798, 2007, doi: 10.1007/978-3-540-72950-1_77.



- [7] M. Sönmez Turan, K. McKay, D. Chang, J. Kang, N. Waller, J. M. Kelsey, L. E. Bassham, and D. Hong, "Status report on the final round of the NIST Lightweight Cryptography Standardization Process," NISTIR 8454, National Institute of Standards and Technology, Gaithersburg, MD, USA, Jun. 2023.
- [8] M. N. Khan, A. Rao, and S. Camtepe, "Lightweight cryptographic protocols for IoT-constrained devices: A survey," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4132–4156, 2021, doi: 10.1109/JIOT.2020.3026493.
- [9] M. Abujoodeh, L. Tamimi, and R. Tahboub, "Toward lightweight cryptography: A survey," *IntechOpen*, 2023, doi: 10.5772/intechopen.109334.
- [10] T. Sylla, M. A. Chalouf, F. Krief, and K. Samaké, "Context-aware security in the Internet of Things: A survey," *International Journal of Autonomous and Adaptive Communications Systems*, vol. 14, no. 3, pp. 231–263, 2021, doi: 10.1504/IJAACS.2021.117808.
- [11] A. I. B. Alotaibi and A. Oracevic, "Context-aware security in the Internet of Things: What we know and where we are going," in *Proc. IEEE International Symposium on Networks, Computers and Communications (ISNCC)*, 2023, pp. 1–8, doi: 10.1109/ISNCC58260.2023.10323735.
- [12] A. Chowdhury, S. A. Rahman, and A. K. M. M. Rahman, "A survey study on Internet of Things resource management," *Journal of Network and Computer Applications*, vol. 120, pp. 42–60, 2018, doi: 10.1016/j.jnca.2018.07.007.
- [13] D. Karaboga and B. Basturk, "Artificial Bee Colony (ABC) optimization algorithm for solving constrained optimization problems," in *Foundations of Fuzzy Logic and Soft Computing*, Lecture Notes in Computer Science, vol. 4529, pp. 789–798, 2007, doi: 10.1007/978-3-540-72950-1_77.
- [14] R. Bruno, D. Weikert, M. H. E. M. et al., "Survey on multi-objective task allocation algorithms for IoT networks," *Sensors*, vol. 23, no. 1, Art. no. 142, 2023.