Intelligent Selection of Key Generation Methods in Cryptography via Classical Q-Learning

Rajaosolomanantena Haingonirina Ignace[1], Ravaliminoarimalalason Toky Basilide[2], Randimbindrainibe Falimanana[3]

[1] *Student at Ecole Doctorale en Sciences et Techniques de l'Ingénierie et de l'Innovation, Laboratory of Cognitive Sciences and Applications, University of Antananarivo, Madagascar*
[2] *Professor at Ecole Doctorale en Sciences et Techniques de l'Ingénierie et de l'Innovation, Laboratory of Cognitives Sciences and Applications, University of Antananarivo*
[3] *Professor at Ecole Doctorale en Sciences et Techniques de l'Ingénierie et de l'Innovation, Laboratory of Cognitives Sciences and Applications, University of Antananarivo*

ABSTRACT

The study highlights the effectiveness of the Q-learning system in cryptographic key generation against simulated attacks. Its dynamic adaptability, adjusting strategies based on the type of attack, is noteworthy. The analysis shows that the system learns to counter threats by choosing specific key generation methods. The curves of Q-values illustrate the system's constant progression, especially for the "Dynamic Key Adaptation" strategy, emphasizing the proactive ability of Q-learning to adjust to attacks and enhance the security of digital exchanges.

Keyword : - *Q-learning, Key generation, Dynamic adaptability, Cybersecurity, Simulated attacks.*

1. INTRODUCTION

Robust cryptographic keys are essential to ensure the security of exchanges. The integration of Q-Learning into this process represents an adaptable and forward-thinking approach, enabling intelligent adaptation to emerging threats. This methodology meticulously selects key generation methods in response to intrusion attempts or identified patterns.

2. LITERATURE REVIEW ON CONTEMPORARY KEY GENERATION METHODS

Several significant research efforts have contributed to enriching this domain, highlighting the growing importance of innovative methods. A study titled "Machine Learning Based Key Generating for Cryptography," conducted by Hayfaa Abdulzahra Atee, Norliza Noor, Robiah Ahmad, and Abidulkarim K. I. Yasar and published in January 2026 on ResearchGate, explores the use of machine learning to generate cryptographic keys [1]. Another relevant research is that of Satyanarayana Pamarthi and R. Narmadha, published on SpringerLink in November 2021, which examines the key generation process in chaotic maps, using the SA-SFO algorithm to produce optimal key pairs [2]. Additionally, an article published on IEEE in 2022 by J.S. Prasath, Deepa Jose, B. Rammyaa, and R. Pandian, titled "Dynamic Key Generation Mechanism to Strengthen Data Security," presents a new dynamic key generation algorithm, generating unique keys for each encoding to enhance data security [3]. These contributions demonstrate the constant evolution of key generation methods and their adaptation to contemporary challenges in computer security.

3. CONTEXT AND RELEVANCE OF Q-LEARNING

In the context of secure digital exchanges, the robustness of cryptographic keys is vital to ensure the confidentiality and authenticity of data. Threats such as interception or tampering underscore these critical issues. The introduction of Q-Learning, a machine learning algorithm, provides an innovative solution to this challenge. Based on trial-and-error reinforcement, Q-Learning dynamically adapts to the evolving landscape of attacks [4][5]. By identifying patterns or intrusion attempts, it enables intelligent selection of cryptographic key generation methods.

4. METHODOLOGY

4.1 Q-Learning Algorithm

```
Start
  Initialize Q(s, a) to 0 for all a in A and s in S

  Repeat for each episode:
    Choose the initial state s

    Repeat for each step of the episode until reaching the terminal state:
      Select an action a from s using a policy based on Q (e.g., epsilon-greedy)
      Take action a, observe the reward r, and the next state s'
      Update Q(s, a) <- Q(s, a) + α * [r + γ * max(Q(s', a')) - Q(s, a)]
      s <- s'

    End of the episode steps loop

  End of the episodes loop
End
```

The Q-learning algorithm is a reinforcement learning technique where Q-values are updated based on observed rewards and future estimates [6]. The agent explores the environment, chooses actions via an epsilon-greedy policy, observes rewards, and iteratively updates its knowledge. This approach strengthens the agent's ability to make optimal decisions over time [7].

4.2 Initialization of the Q-Learning Process

The initialization of Q-Learning involves initially setting the values of the Q-table, represented as $Q(s, a)$, for each state s and action a, along with learning parameters such as the learning rate α and the discount factor γ. This is expressed as: [6]

$$Q(s, a) \leftarrow 0 \; for \; each \; s \in S, a \in A \tag{1}$$

Where S is the set of states, and A is the set of possible actions in the given context.

4.3 Detection of Attack Attempts or Patterns

The process of detecting attack attempts by Q-Learning can be based on specific metrics to assess the system's state. If we consider a generic metric M evaluating state s_t at time t and a detection function f(M), the detection could be formulated as follows: [7]

- M = Metric evaluating the system state
- s_t = System state at time t
- $f(M)$ = Detection function applied to metric M

In the context of attack detection, the function $f(M)$ can be expressed as:

$$f(M) = H(M - S) \tag{2}$$

Where S represents the threshold, and H is the activation function (threshold function). If the metric M surpasses the threshold S, $f(M)$ will be 1, indicating an attack attempt. Otherwise, $f(M)$ will be 0.

Thus, the detection of attack attempts or patterns can utilize $f(M)$ to trigger specific responses when M surpasses or does not meet a predefined threshold, indicating an abnormal state or potentially an attack [8].

4.4 Key Generation Method Selection

The selection of action a_t, which is the key generation method at state s_t, is determined using the policy π, which can be defined as follows :

$$\pi(s_t) = \arg max_a Q(s_t, a) \quad (3)$$

Where s_t represents the current state at time t, a_t is the action (key generation method) selected at time t, $Q(s_t, a)$ represents the value in the Q-table for state s_t and action a, and $\arg max_a$ denotes the action a that maximizes the value in the Q-table for state s_t.

This equation determines the key generation method to be used at each step by choosing the one that maximizes the predicted value in the Q-table for the current state [9].

4.5 Evaluation and update of the Q-Table

The evaluation of generated keys can be represented by a reward function R that assigns a value to each generated key. This function assesses the robustness or quality of the produced key based on security criteria.

$$R(s_t, a_t) = \omega_1 . Criterion_1(s_t, a_t) + \omega_2 . Criterion_2(s_t, a_t) + \omega_3 . Criterion_3(s_t, a_t) \quad (4)$$

In this formula, $Criterion_1(s_t, a_t)$, $Criterion_2(s_t, a_t)$, and $Criterion_3(s_t, a_t)$ are the evaluations of specific security criteria for the key generated at state s_t with action a_t, and ω_1, ω_2, and ω_3 are the weights assigned to each criterion respectively.

This reward is utilized to update the Q-table using the standard Q-learning update algorithm, which involves adjusting the Q-value for state s_t and action a_t based on the obtained reward and the current values in the Q-table [10]. The Q-table update can be performed using the Q-update formula Q [10].

La mise à jour de la table Q peut être réalisée en utilisant la formule de mise à jour Q :

$$Q(s_t, a_t) \leftarrow (1 - \alpha) . Q(s_t, a_t) + \alpha . [R(s_t, a_t) + \gamma . max_a Q(s_{t+1}, a)] \quad (5)$$

Where α is the learning rate controlling the extent to which new information replaces old Q-values, γ is the discount factor determining the importance of long-term rewards compared to immediate rewards, s_{t+1} represents the next state, and a is the possible action at the next state.

This Q-table update is iterative and occurs after each key generation and evaluation, allowing Q-learning to adapt to received rewards and improve the quality of decisions made when selecting key generation methods [11].

5. RESULTS

5.1 Progression of Q-Learning Q-Table Adaptation in Response to Cryptographic menaces

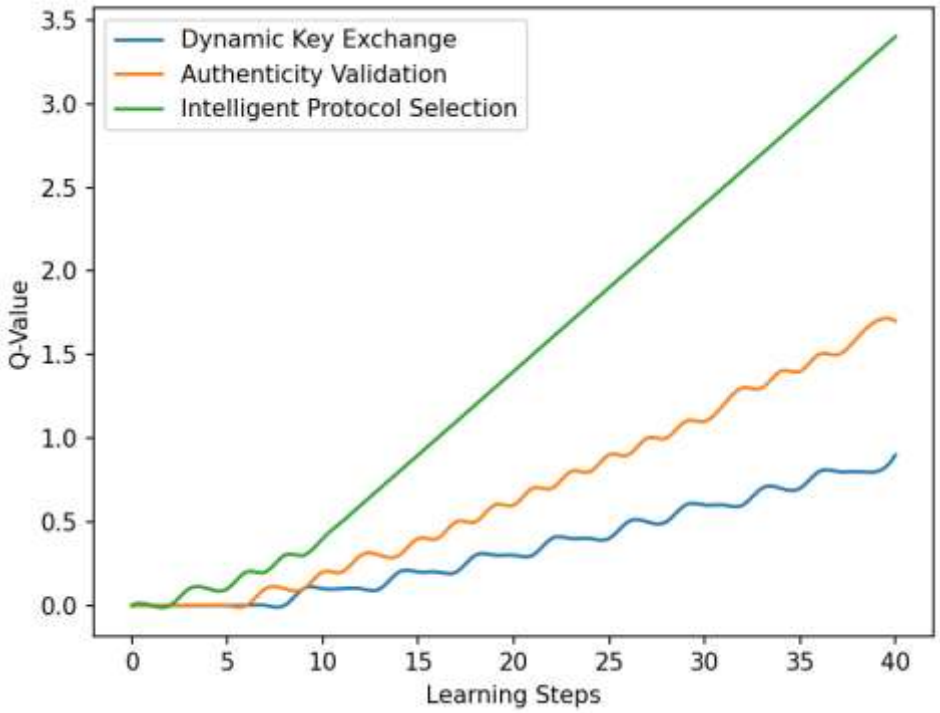

Fig -1: Evolution of Q-Learning Q-Table Adaptation Against Cryptographic menaces

The dynamic key exchange scheme refers to the periodic and proactive modification of cryptographic keys, thereby reducing risks associated with potential compromise. Authenticity validation involves verifying the identity of the parties, ensuring that only legitimate entities are involved in communication. Finally, intelligent protocol selection dynamically adapts the choice of communication protocol based on context, optimizing security and real-time performance. The upward trends in Q-table values for strategic actions such as "Dynamic Key Adaptation," "Authenticity Validation," and "Protocol Selection" highlight the continuous adaptability of Q-Learning to enhance the security of digital exchanges. This consistent improvement denotes its ability to proactively adjust to potential attacks.

5.2 Performance Comparison Against Attacks

Step 1: No attack. Normal key generation: generate_random_seq
Generated key: a@12KLpqw
Step 2: Key weakness attack detected. Parameter adjusted: 2
Key generation method: generate_weighted_random
Generated key: ACCCAACBCC
Step 3: No attack. Normal key generation: generate_random_seq
Generated key: b?34hjx !eV
Step 4: Fault injection attack detected.
Key generation method: generate_modulated_random
Generated key: rBV{E9W34:
Step 5: Brute force attack detected.
Key generation method: generate_random_seq
Generated key: j7HFsDKdFW
Step 6: Differential cryptanalysis attack detected.
Key generation method: generate_modulated_random
Generated key: vxgID3cM8k

Step 7: No attack. Normal key generation: generate_random_seq
Generated key: gL14PT*#tyP*
Step 8: Covert channel attack detected. Parameter adjusted: 2
Key generation method: generate_weighted_random
Generated key: ACCCACCCAB
Step 9: No attack. Normal key generation: generate_random_seq
Generated key: PO98%fge)t
Step 10: No attack. Normal key generation: generate_random_seq
Generated key: XO78%ghai%
Step 11: Brute force attack detected.
Key generation method: generate_random_seq
Generated key: a3N4l878Q7
Step 12: Attack detected. Parameter adjusted: 2
Key generation method: generate_weighted_random
Generated key: ACCBCCCACA
Step 13: No attack. Normal key generation: generate_random_seq
Generated key: UOtyw12Ly
Step 14: Private key compromise attack detected. Parameter adjusted: 0
Key generation method: generate_random_seq
Generated key: DyCYPls20

The results convincingly demonstrate the system's dynamic adaptive capability in the face of attacks. Detection of an attack at step 2 leads to an adjustment of the system parameter, which then selects the key generation method based on a weighted random sequence. This strategy aims to counter a random attack by generating complex keys. When a fault injection attack is detected at step 4, the system adjusts again by choosing the random modulation method.

5.3 Adaptation of Key Generation Methods in Response to Attacks

The stacked bar chart illustrates the reactivity rate of key generation methods at each stage of attack detection.

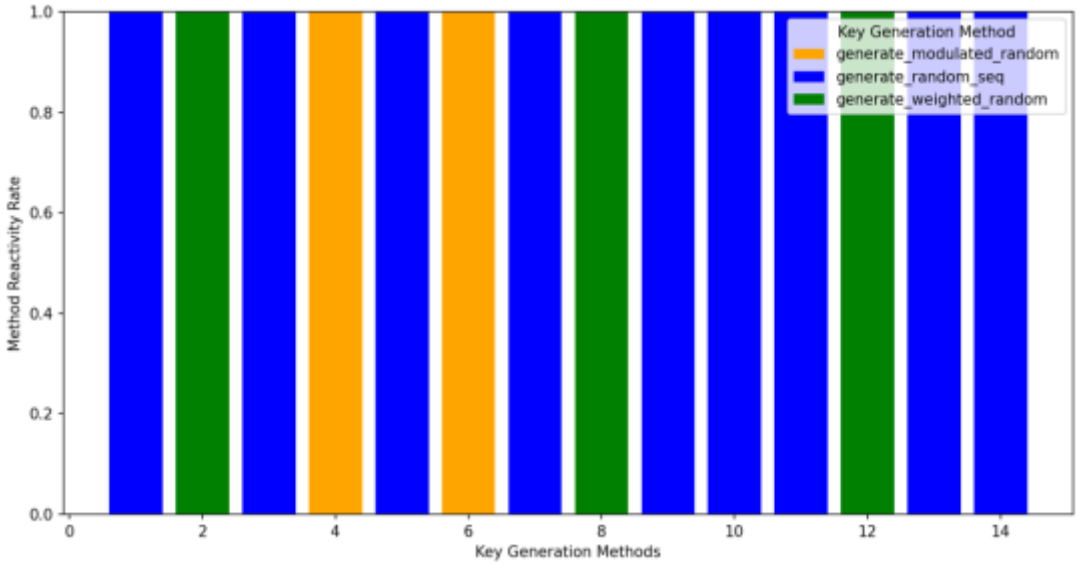

Fig -2: Key Generation Method Change in Response to Detected Attacks

Each method, represented by a distinct color (blue for generate_random_seq, green for generate_weighted_random, and orange for generate_modulated_random), shows a 100% response to each attack, corresponding to a value of 1

on the graph. This result suggests that each method systematically changes in response to an attack, demonstrating maximum responsiveness upon detection. Custom labels on the x-axis indicate different stages of the process, while the y-axis is labeled as the 'Method Reactivity Rate,' emphasizing the methods' ability to quickly adjust their key generation strategy in the presence of threats.

5.4 Performance Analysis in the Face of Attack Frequency Variations

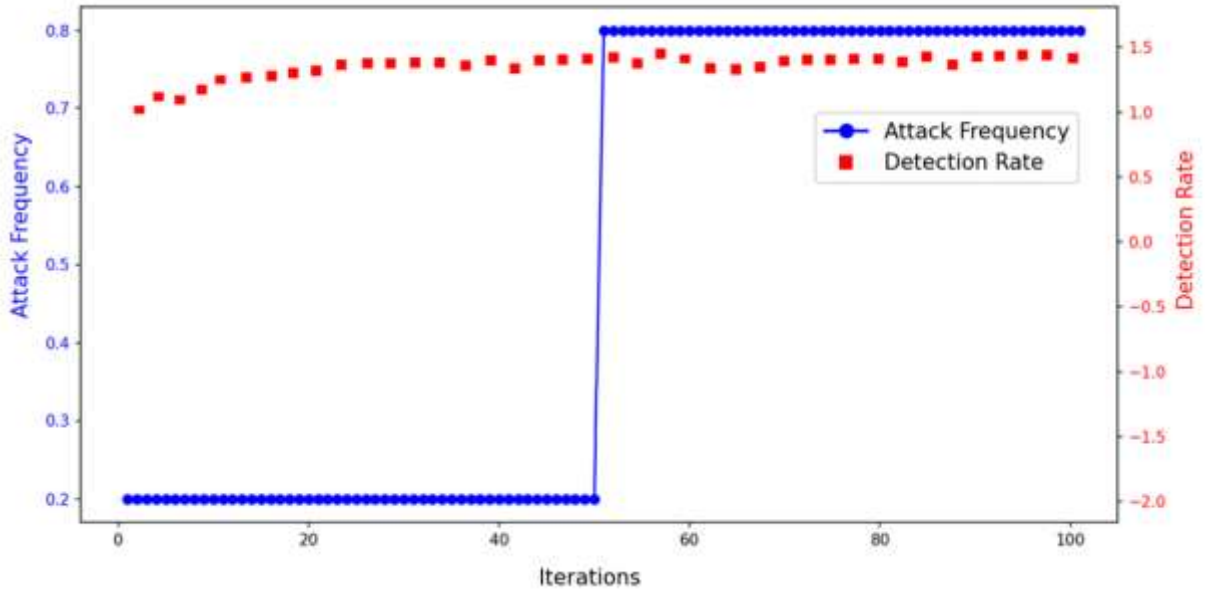

Fig -3: Variation in Attack Frequency and Detection Rate

The results of the attack detection system can be interpreted by considering variations in attack frequency over multiple iterations. When the attack frequency is relatively low (20%) during the initial iterations (iterations 1 to 50), the system demonstrates a consistent ability to identify suspicious patterns or behaviors, even in periods of low attack activity. In a later phase with a considerably higher attack frequency (80%) in subsequent iterations (iterations 51 to 100), the system maintains its robustness by effectively detecting attacks despite a significant increase in frequency. Notably, the attack detection rate remains nearly constant, oscillating between 1 and 1.5, emphasizing the stability and effectiveness of the system. This constancy in the attack detection rate, even in the face of significant variations in attack frequency, reflects the system's ability to dynamically adjust its key generation methods in response to detected attacks, demonstrating its resilience.

6. OBSERVATIONS

The results demonstrate the remarkable adaptability of the system to attacks, efficiently adjusting its parameters and key generation methods in response to the detection of specific attacks. Frequent changes in key generation methods indicate a dynamic adaptation strategy, strengthening the system's security. Even with a significant increase in the attack frequency to 80%, the system maintains its detection capability, highlighting its robustness against substantial increases in the number of attacks. In summary, the system exhibits effective strategic flexibility to counter various cryptographic threats, enhancing its security in environments where attack frequencies may vary.

7. INTERPRETATIONS

The results highlight the remarkable ability of the system to strategically adjust to cryptographic attacks. Detecting attacks early, the system adapts its key generation methods specifically, illustrating a proactive response to each

threat. The frequent changes in key generation methods reflect dynamic adaptation, strengthening system security. Despite a significant increase in the frequency of attacks, the system maintains its detection capability, demonstrating its robustness. These results confirm the reliability of the system in realistic scenarios, establishing a solid foundation for its application in critical environments requiring effective protection against cryptographic threats.

8. CONCLUSION AND PERSPECTIVES

The conclusion underscores the effectiveness of the adaptive system based on Q-learning in cryptographic key generation, with successful strategic adjustments in response to simulated attacks. The results highlight the importance of adaptability and diversification of methods to enhance the security of digital exchanges. Future perspectives could involve exploring more advanced strategies, integrating anomaly detection mechanisms, and optimizing learning parameters for finer decision-making.

9. REFERENCES

[1]. H. A. Atee, N. Noor, R. Ahmad, A. K. I. Yasar (janvier 2026). "Machine Learning Based Key Generating for Cryptography". Middle Technical University, Universiti Teknologo Malaysia, Al Muthanna University. Article publié sur ResearchGate.

[2]. S. Pamarthi, R. Narmadha (18 novembre 2021). "Adaptive Key Management-Based Cryptographic Algorithm for Privacy Preservation in Wireless Mobile Adhoc Networks for IoT Applications". SpringerLink.

[3]. J.S. Prasath, D. Jose, B. Rammyaa, R. Pandian (2022). "Dynamic Key Generation Mechanism to Strengthen the Data Security". Article publié sur IEEE.

[4]. R. S. Sutton, A. G. Barto (2018). *« Reinforcement Learning: An Introduction »*. MIT Press.

[5]. I. Goodfellow, Y. Bengio, A. Courville (2016). *«Deep Learning »*. MIT Press.

[6]. D. R. Stinson (2006). *« Cryptography: Theory and Practice »*. CRC Press.

[7]. S. J. Russell, P. Norvig (2010). *« Artificial Intelligence: A Modern Approach »*. Pearson.

[8]. M. Abadi (2016). *« Deep Learning with Differential Privacy »*. ACM Transactions on Machine Learning (TOMM).

[9]. D. Boneh, V. Shoup (1997). *« Fast Variants of RSA »*. International Conference on the Theory and Application of Cryptology and Information Security.

[10]. G. A. Rummery, M. Niranjan (1994). *« On-line Q-learning using Connectionist Systems »*. Cambridge University Engineering Department, Internal Report.

[11]. H. Zhang (2019). *«A Survey on Reinforcement Learning in Blockchain: Toward Blockchain 3.0 »*. IEEE Access.