IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2024.0429000

Reinforcement Learning-Based Adaptive Quantum-Safe Cryptography for DN25-Compliant Smart Environments

DARLAN NOETZOLD[1], (Member, IEEE), JORGE L. V. BARBOSA[2], JUAN F. P. SANTANA[1], and VALDERI R. Q. LEITHARDT[1,3], (Senior Member, IEEE)

[1]Expert Systems and Applications Laboratory, University of Salamanca, Salamanca, Spain
[2]PPGCA, University of Vale do Rio dos Sinos (UNISINOS), São Leopoldo, Brazil
[3]ISTAR-IUL, Instituto Universitário de Lisboa (ISCTE-IUL), Lisboa, Portugal

Corresponding author: Darlan Noetzold (e-mail: darlannoetzold@usal.es).

Colaboración Consejería de educación de la Junta de Castilla y León grupo de investigación ESAL-EXPERT SYSTEM AND APPLICATIONS LAB (ESALAB).

ABSTRACT The emergence of quantum computing challenges traditional security mechanisms, particularly in heterogeneous and resource-constrained IoT and smart environments that must satisfy DN25 requirements. This work introduces a reinforcement learning-driven model for the adaptive selection and orchestration of cryptographic algorithms. Acting as an intelligent decision layer, the system observes contextual, network, and operational metrics to recommend or enforce configurations combining classical schemes, post-quantum cryptography, and Quantum Key Distribution when available. The selection problem is formulated as a Markov Decision Process with state and action spaces aligned with protocol control flows and is embedded into a security middleware with negotiation and fallback mechanisms to ensure interoperability and policy compliance without modifying application logic. Experimental results demonstrate that the model dynamically adjusts key lengths, algorithm families, and security policies according to risk and resource conditions, increasing post-quantum cryptography and Quantum Key Distribution usage by up to 33.4% and 23.9% in high-risk scenarios while favoring low-latency classical or hybrid options for less critical traffic. The system achieves success rates above 78% while maintaining stable latency and resource usage during extended operation.

INDEX TERMS Adaptive security, DN25 protocol, post-quantum cryptography, quantum key distribution, reinforcement learning

I. INTRODUCTION

The consolidation of ubiquitous computing and large-scale sensing infrastructures has led to the emergence of highly interconnected smart environments, in which thousands of heterogeneous devices continuously exchange sensitive data [1]. Such environments include industrial automation, smart buildings, and critical infrastructures, all of which rely on communication stacks that must remain secure despite severe constraints in processing power, energy, and network bandwidth. In this context, security mechanisms that are provisioned once and statically configured are increasingly inadequate, as they cannot easily adapt to changes in device capabilities, risk exposure, or regulatory demands [2].

At the same time, the progressive maturation of quantum computing technologies poses a concrete threat to many widely deployed public-key schemes. This scenario motivates a transition toward quantum-safe protections, combining post-quantum cryptographic (PQC) algorithms with techniques such as Quantum Key Distribution (QKD). Documentation from the National Institute of Standards and Technology (NIST) [3] details the threat quantum computers pose to classical cryptographic algorithms and outlines the ongoing standardization process for PQC. Additionally, organizations such as the European Telecommunications Standards Institute (ETSI) and the International Organization for Standardization (ISO) contribute to defining standards and best practices for quantum-safe cryptography [4], [5]. However, integrating these mechanisms into heterogeneous, resource-constrained environments is non-trivial: different devices may support different cryptographic primitives, and the most secure option is not always feasible due to latency or energy budgets. Selecting an appropriate cryptographic configuration therefore

becomes a multi-objective decision problem that must take into account both classical and quantum-era adversaries [6].

Reinforcement Learning (RL) provides a natural paradigm for tackling this decision process, as it allows an autonomous agent to explore and exploit security configurations based on feedback from the environment [7], [8]. By observing factors such as current threat level, hardware capabilities, protocol constraints, and quality-of-service requirements, an RL-based agent can gradually learn policies that recommend or enforce suitable cryptographic strategies. This shifts the focus from manually engineered decision rules to data-driven adaptation, which is particularly appealing in dynamic and uncertain environments.

This work investigates a reinforcement learning-driven model dedicated to adaptive cryptographic selection under the DN25 protocol. Instead of designing a full-stack security framework, the focus is on the decision core that dynamically selects the combination of classical, post-quantum, and quantum cryptographic mechanisms appropriate for a given communication context. The model is designed as a pluggable component that can be embedded into security middleware or protocol implementations, providing an intelligent layer capable of adjusting key sizes, algorithm families, and security policies without requiring changes to the application logic.

The proposed model formulates cryptographic selection as a Markov Decision Process, whose state encodes contextual information (e.g., device profile, network conditions, and risk indicators), and whose actions represent different cryptographic configurations, including DN25-compliant options. We explore and compare different RL algorithms to optimize long-term rewards that capture security strength, performance overhead, and resource consumption. Additionally, we incorporate practical constraints such as hardware availability and compatibility, enabling the model to gracefully degrade to alternative algorithms when ideal options are not supported.

The effectiveness of the approach is assessed through experiments in simulated smart environments that emulate heterogeneous devices and fluctuating threat scenarios. Results demonstrate that the RL-based model learns policies outperforming static cryptographic choices in both security and efficiency, while providing a flexible path for incremental adoption of quantum-safe primitives. These findings indicate that reinforcement learning can play a central role in enabling context-aware, future-ready cryptographic management in DN25-based and similar communication ecosystems.

In the specific case of DN25, the communication stack imposes concrete constraints that directly shape the decision process. DN25 defines a control flow in which security negotiation occurs during service discovery and session establishment, with strict bounds on handshake latency, message sizes, and key lifetimes for different service classes [9]. Furthermore, the protocol specification organizes services into confidentiality tiers, enforces minimum key sizes and algorithm families per tier, and distinguishes between legacy-only endpoints and quantum-safe–capable nodes [10]. These requirements motivate modeling cryptographic selection as an MDP whose state explicitly encodes DN25 service identifiers, confidentiality tiers, and endpoint capability profiles, and whose action space is restricted to configurations that are admissible for each DN25 flow. In this way, the RL agent does not operate in an abstract environment, but rather in a decision space that is aligned with concrete DN25 control messages and policy rules.

The remainder of this paper is organized as follows. Section II surveys related work on adaptive cryptography and RL in security. Section III describes the proposed model, including environment representation, state and action spaces, and RL algorithms. Section IV presents the experimental setup and results. Finally, Section V concludes the paper and outlines directions for future research.

II. STATE OF THE ART

Adaptive security for intelligent environments has recently attracted significant attention. Relevant topics include reinforcement learning (RL) for security control and key management, context-aware integration of post-quantum cryptography (PQC) in IoT, lightweight and hybrid cryptography for constrained devices, and quantum-safe mechanisms such as QKD. The following selected works (2024–2026) illustrate advances that are relevant to an RL engine that selects cryptographic algorithms based on context, resources, and policy constraints. To ensure a high-quality baseline, this review prioritizes recent contributions from high-impact, peer-reviewed journals that address the intersection of quantum resilience, AI, and resource-constrained security.

Table 1 summarises the chosen works, their relevance and key characteristics. Recent works cover complementary aspects needed for adaptive cryptographic selection. Nenov [12] applies RL to key lifecycle management in distributed systems, modeling rotation policies as an MDP but not addressing heterogeneous primitive selection. Holgado et al. [13] study context-aware PQC deployment on constrained IoT devices, providing practical performance profiles and heuristics that can inform cost modeling for adaptive engines.

Several contributions focus on building blocks for adaptive security control loops. Alsalim [18] develops efficient anomaly detectors for storage telemetry that can feed RL state features. Shingne et al. [19] combine deep RL and variational autoencoders to optimize QKD parameters, illustrating how RL can handle quantum-specific performance and security metrics. Sheela et al. [14] demonstrate the effectiveness of RL in managing secure transmissions in wireless sensor networks by combining it with homomorphic encryption, providing a benchmark for RL-driven security selection. Rafat et al. [20] benchmark lightweight algorithms on ESP32 platforms, offering latency and energy data that are directly applicable to RL reward and constraint design. Venkatesh et al. [21] use RL for trust-based mitigation of attacks in IIoT networks, showing how dynamic trust scores can be embedded in RL state and reward definitions.

Other works address cryptographic agility and quantum-safe primitives. Kourtis et al. [22] investigate adaptive PQC

TABLE 1. Selected recent works (2024–2026) relevant to RL-based adaptive cryptographic selection

Ref.	Year	Topic	Relevance to RL engine	Characteristics
[11]	2024	Requirements-driven autoscaling	Self-adaptive control loop for resource allocation based on SLOs.	MAPE-K-based autoscaling with requirements-aware decisions for microservices.
[12]	2024	RL for key management	RL/DRL for key lifecycle decisions related to selection logic.	MDP/DRL for rotation and lifecycle policies in distributed systems.
[13]	2024	Context-aware PQC for IoT	Context-based PQC selection and feasibility trade-offs.	Empirical profiling and heuristics on constrained devices.
[14]	2025	RL + HE for WSN	RL-driven security selection for sensor networks.	Hybrid RL and Homomorphic Encryption for secure transmission.
[15]	2025	PQC for On-device AI	Quantum-resilient frameworks for mobile architectures.	Privacy-preserving PQC integration in Apple MM1 hardware.
[18]	2025	Anomaly detection for storage security	Anomaly-driven triggers feeding adaptive security decisions.	Efficient detectors for telemetry that can serve as RL state inputs.
[19]	2025	DRL + VAE for QKD optimization	RL for tuning quantum-security parameters and detecting anomalies.	DRL with VAE and reward engineering for QKD-specific metrics.
[20]	2025	Lightweight crypto on ESP32	Hardware-level benchmarks for constrained IoT platforms.	Latency and energy measurements usable in RL cost models.
[21]	2025	RL for trust-based mitigation	RL with trust metrics for dynamic security responses in IIoT.	Trust-aware states and rewards for attack mitigation.
[22]	2025	Adaptive PQC for blockchain	Cryptographic agility balancing security and energy.	Policy rules for switching PQC schemes with energy-aware criteria.
[23]	2025	Lattice-based adaptive signatures	New PQC primitives suitable for fine-grained selection.	Puncturable signatures with adaptive security from lattices.
[24]	2026	PQC + Blockchain for Edge	Quantum-resilient communication for secure edge devices.	Framework for PQC-based IoT security using blockchain.
[25]	2026	Zero-trust AI security	Categorical framework for quantum-resistant AI.	Formal zero-trust models for quantum-resistant AI systems.
[27]	2026	Secure signcryption for HSR	Identity-based, adaptive signcryption in critical communication.	Equality-test signcryption tailored to high-speed railway links.
[28]	2026	LLM-based adaptive IDS with PQ blockchain	Adaptive detection and PQ-secure ledger for IoT threats.	Hybrid LLM/ML IDS with PQ-secure blockchain-backed data.
[29]	2026	Lightweight and hybrid crypto review	Survey of resource-efficient and hybrid schemes for IoT.	Comparative analysis of ECC, ABE, QKD, and hybrid encryption.
[30]	2026	Lightweight hybrid signcryption	Hybrid signcryption for heterogeneous public-key systems.	Heterogeneous compatibility and efficiency for smart grids.
This work	2026	RL engine for adaptive cryptography	MDP-based RL engine selecting classical, PQC, and quantum-assisted algorithms.	Modular design with multi-objective rewards; supports DN25 simulation and middleware integration.

for blockchain, proposing policies that switch cryptographic schemes based on security and energy objectives. Narayanan et al. [24] propose a quantum-resilient framework for IoT that integrates PQC with blockchain, specifically targeting secure communication for edge devices. Similarly, Umer et al. [15] explore quantum-resilient security for privacy-preserving AI, focusing on the integration of PQC within modern on-device architectures such as Apple's MM1. Zhang et al. [23] introduce lattice-based puncturable signatures with adaptive security guarantees, expanding the set of PQC options that an adaptive engine could select. Cherkaoui et al. [25] provide a formal categorical framework for zero-trust AI security, establishing theoretical foundations for quantum-resistant systems. Du et al. [27] design an adaptive secure identity-based signcryption scheme for high-speed railway communications, demonstrating the need for flexible cryptography in safety-critical systems. Ho et al. [30] propose a lightweight hybrid signcryption scheme for smart grids that supports heterogeneous public-key systems, highlighting interoperability requirements that also arise in DN25-based environments.

Huang et al. [28] present an adaptive intrusion detection system for IoT that combines large language models, lightweight ML, and a post-quantum-secure blockchain. Although focused on detection rather than encryption, their architecture shows how PQ-secure backends and adaptive decision layers can coexist. Haider et al. [29] provide a comprehensive survey of lightweight and hybrid cryptographic schemes for IoT, emphasizing trade-offs among efficiency, security level, and scalability, and calling for more work on machine-learning-driven adaptive encryption.

In contrast to adaptive security models that rely on static heuristic rules or focus on single-tier hardware, this framework operates as a multi-objective optimization system. While previous studies [14], [24] address specific PQC or RL applications, these solutions often lack a unified decision engine spanning from microcontrollers to high-end servers. This approach bridges the gap by integrating DN25-specific protocol constraints directly into the MDP formulation, ensuring that the learned policies remain compliant with industrial communication standards while optimizing the Energy-Security Efficiency (ESE) across a heterogeneous device spectrum.

III. METHODOLOGY

Figure 1 illustrates the architecture of the proposed Reinforcement Learning (RL) engine for adaptive cryptographic selection within DN25-compliant smart environments. This section details the technical components and operational flow of this engine. The process begins with an *Incoming Transaction*, a structured payload containing contextual metadata such as request identifiers, timestamps, risk scores, confidentiality requirements, device characteristics, and policy flags. This raw input is processed by the *Preprocessing* stage, which performs normalization, encoding, and embedding to produce

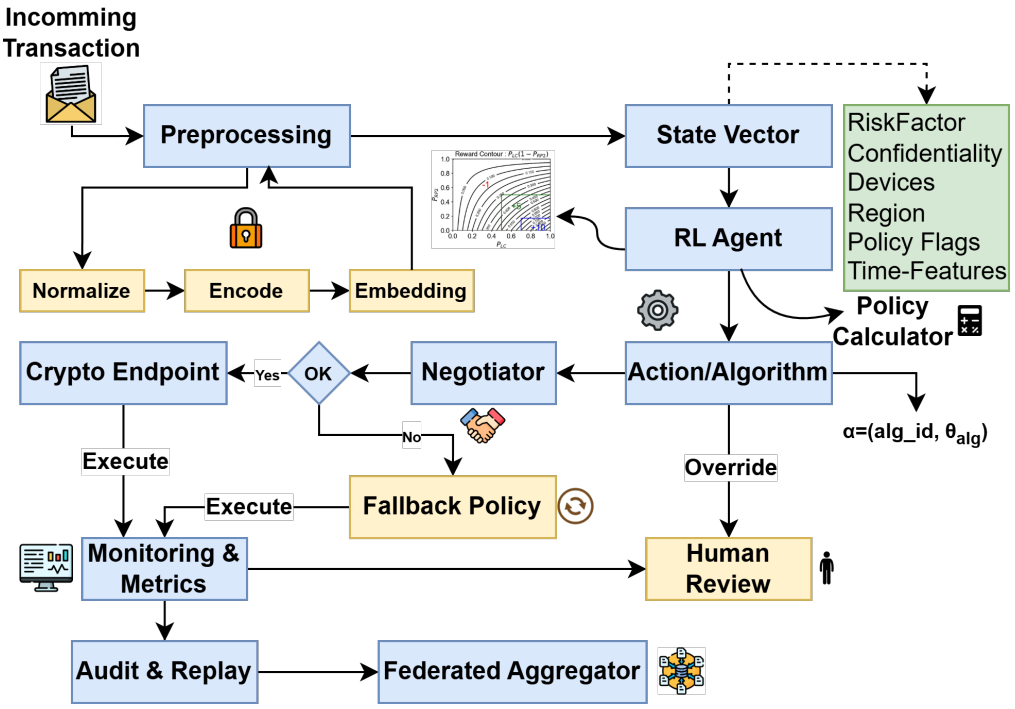


FIGURE 1. Architecture of the RL-driven cryptographic selection engine.

a rich feature representation.

These features are aggregated into a *State Vector* that captures the current operational context, including dynamic risk factors, device capabilities, regional constraints, and active policy flags. This vector serves as input to the *RL Agent*, implemented with algorithms such as DQN or PPO, which outputs an *Action/Algorithm* (e.g., AES, Kyber, QKD) together with its configuration parameters. The selected action is then evaluated by the *Negotiator* model, which checks feasibility against endpoint capabilities and policy constraints. If negotiation succeeds, the decision is enforced at the *Crypto Endpoint*; otherwise, a *Fallback Policy* is invoked to select a safe alternative configuration.

Execution outcomes and performance indicators are collected by the *Monitoring & Metrics* component, which computes rewards and generates telemetry. These data are persisted in an *Audit & Replay* store and used to drive a *Federated Aggregator* that periodically updates the RL models. The following subsections describe the data flow and decision loop, the candidate cryptographic algorithms, and the Markov Decision Process (MDP) formulation that underlies the RL engine.

A. DATA FLOW AND DECISION LOOP

Algorithms 1, 2, and 3 formalize the operational core of the RL engine. The end-to-end process decomposes into: (i) an online decision loop performing inference, negotiation, and cryptographic execution for each transaction; (ii) a reward, logging, and replay-construction step; and (iii) a periodic training and federated aggregation routine updating the deployed policy.

Algorithm 1 Online Decision Loop

```
[1:] procedure OnlineDecisionLoop
[2:] Input: Stream of incoming transactions {T_1, T_2, ...}
[3:] Output: Execution outcomes o_t, metrics m_t
[4:] while each transaction T_t received do
[5:]    x_t ← Validate_and_Canonicalize(T_t)
[6:]    if x_t = INVALID then
[7:]       Logger.Error(T_t.id, "invalid_payload")
[8:]       continue
[9:]    end if
[10:]   f_t ← Extract_and_Normalize(x_t)
[11:]   f_t ← f_t ∪ AggregateHistory(T_t.src, T_t.dst)
[12:]   α_t ← A(f_t)
[13:]   c_t ← Π(f_t, T_t.policy_flags)
[14:]   s_t ← φ(f_t, α_t, c_t)
[15:]   a_t, η_t ← π_θ(a | s_t)
[16:]   ã_t, neg_t ← Negotiator.Negotiate(a_t, η_t, c_t, T_t.dst)
[17:]   if neg_t.success = false then
[18:]      Logger.Warning(T_t.id, "negotiation_failed")
[19:]      ã_t, η_t ← Fallback.Policy(T_t, c_t)
[20:]   end if
[21:]   o_t ← CryptoEngine.Execute(ã_t, η_t, T_t.payload)
[22:]   m_t ← Monitor.Collect(o_t, neg_t)
[23:]   HandleOutcome(T_t, s_t, ã_t, m_t, α_t)
[24:] end while
[25:] end procedure
```

Algorithm 1 specifies the online decision loop executed for each incoming transaction. After schema validation and canonicalization (lines 5–9), the system extracts and normalizes a feature vector f_t (lines 10–11), augmented with historical aggregates such as rolling latency, failure rates, and previous security incidents. An anomaly detector $\mathcal{A}(\cdot)$ computes a score $\alpha_t \in [0, 1]$ (line 12), while the policy engine $\Pi(\cdot)$ derives a constraint vector c_t that encodes mandatory requirements and soft preferences (line 13).

The encoder $\phi(\cdot)$ fuses telemetry f_t, authentication/confidence signals α_t, and policy/constraint vectors c_t into the state vector

$$s_t = \phi(f_t, \alpha_t, c_t), \tag{1}$$

which the RL policy consumes. The policy $\pi_\theta(\cdot \mid s_t)$ outputs a discrete algorithm identifier k_t and an optional continuous parameter vector η_t, forming the action

$$a_t = (k_t, \eta_t) \in A. \tag{2}$$

The negotiator validates the proposed configuration against endpoint capabilities and policy constraints (line 16). If negotiation fails (lines 17–20), the fallback policy produces a safe alternative $\tilde{a}_t$ and returns it to the engine. The cryptographic engine executes the selected configuration (line 21). The monitoring module collects metrics m_t (line 22) and forwards the recorded outcome to the reward-and-logging routine, which constructs the training tuple $(s_t, a_t, r_t, s_{t+1}, d_t)$ and appends it to the replay buffer (line 23).

Algorithm 2 Reward Computation, Logging, and Replay Construction

[1:] **procedure** HandleOutcome
[2:] **Input:** Transaction T_t, state s_t, action a_t, metrics m_t, anomaly score α_t
[3:] $(L_t, C_t, S_t, V_t) \leftarrow \Psi(m_t)$
[4:] $F_t \leftarrow \mathbb{1}\{m_t.security\_outcome = \texttt{FAIL}\}$
[5:] $r_t \leftarrow w_{\text{sec}} S_t - w_{\text{lat}} L_t - w_{\text{cost}} C_t - w_{\text{comp}} V_t - w_{\text{fail}} F_t$
[6:] Logger.Audit$(T_t.id, s_t, a_t, m_t, r_t)$
[7:] **if** *training_mode* $=$ `true` **then**
[8:] $(s_{t+1}, d_t) \leftarrow$ ComputeNextState(T_t)
[9:] ReplayBuffer.Store$((s_t, a_t, r_t, s_{t+1}, d_t))$
[10:] **end if**
[11:] **if** $F_t = 1$ **or** $\alpha_t > \theta_{critical}$ **then**
[12:] AlertService.NotifyOperator$(T_t.id, s_t, a_t, m_t)$
[13:] **end if**
[14:] **end procedure**

Algorithm 2 translates raw execution metrics into rewards and replay tuples. The mapping function $\Psi(\cdot)$ extracts latency L_t, computational/energy cost C_t, security outcome S_t, and a compliance-violation indicator V_t from the metric record m_t (line 3). Line 4 constructs a failure flag F_t when the security outcome indicates a breach or unsuccessful protection. The scalar reward computes as

$$r_t = w_{\text{sec}} S_t - w_{\text{lat}} L_t - w_{\text{cost}} C_t - w_{\text{comp}} V_t - w_{\text{fail}} F_t, \tag{3}$$

where $w_{\text{sec}}, w_{\text{lat}}, w_{\text{cost}}, w_{\text{comp}}, w_{\text{fail}} \in \mathbb{R}_{\geq 0}$ represent deployment-specific weights tuning the trade-off between security, performance, and resource consumption. The tuple $(s_t, a_t, r_t, s_{t+1}, d_t)$ writes to the audit log and appends to the replay buffer (lines 6–9) when training mode is enabled, with d_t denoting episode termination. Critical failures or high anomaly scores (lines 11–12) generate operator notifications for manual investigation and potential policy updates.

Algorithm 3 describes the background training and federated update routine. When sufficient experience accumulates and a training window arrives (lines 3–5), mini-batches *batch*

Algorithm 3 Periodic Training and Federated Aggregation

[1:] **procedure** PeriodicTraining
[2:] **Input:** Replay buffer $\mathcal{D}$, policy parameters θ
[3:] **if** $|\mathcal{D}| < N_{min}$ **or** $\neg$ time_to_train() **then**
[4:] **return**
[5:] **end if**
[6:] **for** *epoch* $= 1$ **to** N_{epochs} **do**
[7:] *batch* $\leftarrow$ ReplayBuffer.Sample(N_{batch})
[8:] $\nabla_\theta J(\theta) \leftarrow \frac{1}{|batch|} \sum_{(s,a,r,s',d) \in batch} \nabla_\theta \mathcal{L}_{\text{RL}}(s, a, r, s', d; \theta)$
[9:] $\theta \leftarrow \theta - \eta \nabla_\theta J(\theta)$
[10:] **end for**
[11:] $\theta_{local} \leftarrow \theta$
[12:] $\theta \leftarrow$ FederatedAggregate(θ_{local})
[13:] **end procedure**

sample from the replay buffer (line 7). The gradient of a generic RL loss $\mathcal{L}_{\text{RL}}$ estimates over the batch as

$$\nabla_\theta J(\theta) = \frac{1}{|batch|} \sum_{(s,a,r,s',d) \in batch} \nabla_\theta \mathcal{L}_{\text{RL}}(s, a, r, s', d; \theta), \tag{4}$$

and updates policy parameters θ with learning rate η (line 9). For a DQN-style update, the loss function is

$$\mathcal{L}_{\text{RL}}(s_t, a_t, r_t, s_{t+1}, d_t; \theta) = \big(y_t - Q_\theta(s_t, a_t)\big)^2, \tag{5}$$

with target

$$y_t = r_t + \gamma(1 - d_t) \max_{a'} Q_{\theta^-}(s_{t+1}, a'), \tag{6}$$

where γ denotes the discount factor and θ^- the target-network parameters. After local optimization, the resulting parameters θ_{local} send to a federated aggregator (lines 11–12), which computes a global model (e.g., via weighted averaging) and redistributes the aggregated policy back to participating nodes, ensuring consistent adaptation across the distributed DN25 environment.

B. CANDIDATE CRYPTOGRAPHIC ALGORITHMS

The action space of the RL agent encompasses cryptographic primitives spanning three major categories: classical symmetric and asymmetric algorithms, post-quantum cryptography (PQC) candidates standardized or proposed for standardization by NIST, and quantum-assisted key distribution protocols. In addition, the engine exposes hybrid classical–PQC constructions and explicit fallback configurations. This heterogeneous action space enables the middleware to adapt to diverse operational contexts, from legacy systems requiring backward compatibility to high-security environments demanding quantum-resistant protections.

Classical symmetric algorithms such as AES and ChaCha20-Poly1305 provide high-throughput confidentiality and authenticated encryption with associated data (AEAD). Classical asymmetric algorithms, including RSA and ECC-based schemes, remain widely deployed for digital signatures and key establishment, despite known vulnerabilities to quantum attacks. These legacy primitives are retained to ensure interoperability with existing infrastructure and to support gradual migration strategies. Post-quantum cryptography (PQC) algorithms, including lattice-based key encapsulation

mechanisms (KEMs) such as Kyber, NTRU, Saber, and signature schemes such as Dilithium, Falcon, and SPHINCS+, provide quantum-resistant security guarantees. The RL agent selects PQC primitives when risk scores indicate high-value transactions, when compliance policies mandate quantum resistance, or when endpoints advertise PQC support during negotiation.

Hybrid constructions (e.g., RSA+PQC, ECC+PQC, QKD+PQC) combine classical and PQC (or quantum-assisted) mechanisms, either by running them in parallel and deriving a joint session key or by using one scheme to protect the other. These hybrids are particularly useful during migration phases, as they preserve classical interoperability while providing post-quantum security under reasonable assumptions. Quantum-assisted protocols, such as Quantum Key Distribution (QKD) based on BB84, E91, measurement-device–independent schemes (MDI-QKD), and continuous-variable QKD (CV-QKD), leverage quantum mechanical properties to establish cryptographic keys with information-theoretic security guarantees. Decoy-state variants mitigate photon-number–splitting attacks and improve robustness in practical optical channels.

Table 2 summarize all algorithms exposed in the RL action space, their primary use cases, and the parameterization knobs available to the agent. These parameters include key sizes, security levels, operational modes, and algorithm-specific configurations. The agent learns to select both the algorithm class and the specific parameter configuration that optimizes the multi-objective reward function for each transaction context. Each algorithm in the action set is associated with an estimated cost profile that quantifies expected latency, computational overhead (CPU cycles, memory footprint), and energy consumption. These cost profiles are obtained from empirical benchmarks on representative hardware platforms and are stored in a centralized registry maintained by the negotiation model. The registry also encodes compatibility matrices that specify which algorithms and parameter combinations are supported by each endpoint.

The algorithms included in Table 2 were selected according to a set of practical and methodological criteria. First, at the classical level, we prioritized primitives that are widely deployed and standardized (AES, CHACHA20_POLY1305, RSA, ECC), ensuring interoperability with existing infrastructures and providing realistic baselines for performance and security comparison. Second, the PQC candidates are drawn from NIST standardization tracks and commonly used parameter sets (Kyber, Dilithium, NTRU, Saber, Falcon, SPHINCS), so that the RL agent operates over options that are representative of near-term post-quantum deployments. Third, the QKD variants cover complementary implementation paradigms (prepare-and-measure, entanglement-based, measurement-device–independent, continuous-variable, and decoy-state protocols), capturing the diversity of quantum key establishment mechanisms relevant to different network topologies and threat models. Fourth, hybrid schemes were chosen to reflect canonical migration strategies that combine classical and PQC or QKD keys, enabling graceful transition paths instead of abrupt algorithm switches. Finally, all entries were constrained by implementability in the evaluation environment and by the availability of reliable cost and capability data, so that the learned policy is trained and assessed on algorithms that can be realistically deployed in DN25-compliant smart environments.

During negotiation, the negotiator consults the registry to verify that the RL agent's selected action is compatible with the destination endpoint's cryptographic capabilities and satisfies all hard policy constraints. If the selected algorithm is incompatible or violates constraints, the negotiator may reject the selection and trigger the fallback policy, or attempt to negotiate an acceptable alternative. This mechanism ensures robustness to heterogeneous deployment environments and gracefully handles mismatches between agent decisions and real-world constraints.

C. DN25 REQUIREMENTS AND THEIR IMPACT ON THE MDP

DN25 is not a generic transport-agnostic profile but a specialized framework that specifies how endpoints in smart environments discover services, negotiate security properties, and maintain sessions under heterogeneous capabilities. These requirements directly constrain the state space and transition logic of the Markov Decision Process defined in this work.

At the *control-flow level*, DN25 structures communication into a sequence of phases (service discovery, capability advertisement, security negotiation, and data exchange) with explicit timeouts and retry policies. In our MDP, each decision step t coincides with a DN25 transaction that either initiates a session or refreshes its cryptographic context. The state component x_t therefore includes the DN25 service identifier, session phase, and remaining negotiation budget (e.g., number of retries and residual timeout), which influence whether the agent can safely propose more expensive PQC or QKD options.

At the *policy level*, DN25 defines confidentiality classes and associated minimum cryptographic requirements, such as mandatory AEAD for certain control messages, minimum key sizes, and whether quantum-safe primitives are required, recommended, or optional. These rules are encoded in the constraint and policy vector c_t. For instance, DN25 services tagged as ''critical-control'' activate flags that forbid legacy-only algorithms and enforce PQC or hybrid configurations, while ''telemetry'' services allow classical schemes under strict latency budgets. The policy engine $\Pi(\cdot)$ maps DN25 policy descriptors to concrete bits in c_t, which the RL agent must respect when selecting actions.

At the *capability level*, DN25-capable endpoints advertise their supported cipher suites, PQC and QKD capabilities, and maximum acceptable message sizes. This information populates both the historical component h_t (e.g., long-term success rates with each peer and cipher family) and the registry consulted by the negotiator. The discrete action space A_d in (11) is thus not the full Table 2 for every transaction,

TABLE 2. Representative cryptographic algorithms in the RL action space and parameterization

Algorithm	Class	Use case	Parameterization / notes
AES_192	Symmetric	Confidentiality with AEAD or CBC/GCM modes	Key size (192), mode (GCM/CBC), IV/nonce strategy, key lifetime
AES_256_GCM	Symmetric	High-throughput AEAD for data in transit/at rest	Key size (256), GCM tag length, rekey interval, nonce management
CHACHA20_POLY1305	Symmetric	Software-optimized AEAD, constrained devices	Fixed 256-bit key, nonce construction, record size and key lifetime
RSA_4096	Asymmetric	Signatures and key encapsulation for legacy endpoints	Modulus size (4096 bits), padding (OAEP/PKCS#1 v1.5), hash function
ECC_521	Asymmetric	High-security elliptic-curve signatures/exchange	Curve (P-521), hash function, signature format and key lifetime
PQC_KYBER	PQC (KEM)	Lattice-based KEM for PQC key exchange	Security level (Kyber-512/768/1024), key lifetime, encapsulation mode
PQC_DILITHIUM	PQC (signature)	Lattice-based PQC digital signatures	Mode (Dilithium-2/3/5), signature format, key lifetime
PQC_NTRU	PQC (KEM)	Lattice-based KEM with alternative design to Kyber	Parameter set (security level), polynomial degree, key lifetime
PQC_SABER	PQC (KEM)	Module-LWR-based KEM optimized for efficiency	Security level (Light/Saber/Fire), key lifetime, packing options
PQC_FALCON	PQC	Highly compact lattice-based signatures	Parameter set (Falcon-512/1024), precision settings, key lifetime
PQC_SPHINCS	PQC	Hash-based stateless signatures, conservative design	Parameter set (fast/small variants), hash function family, tree height
HYBRID_RSA_PQC	classical+PQC	Migration path for RSA-centric infrastructures	Joint key derivation from RSA and PQC KEMs, policy for failure handling
HYBRID_ECC_PQC	classical+PQC	Migration path for ECC-based deployments	Combined ECDH + PQC KEM, key-combination function, weighting policy
HYBRID_QKD_PQC	QKD+PQC	Highest resilience using both quantum and PQC keys	Key-fusion strategy (XOR/KDF), key lifetime, channel selection
QKD_BB84	QKD	Prepare-and-measure QKD for high-security links	Basis choice distribution, reconciliation protocol, privacy-amplification params
QKD_E91	QKD	Entanglement-based QKD for ultra-secure backbones	Entangled source rate, Bell-test threshold, error-correction scheme
QKD_MDI_QKD	QKD	Measurement-device–independent QKD, side-channel resilient	Bell-state measurement visibility, decoy settings, detector timing windows
QKD_CV_QKD	QKD	Continuous-variable QKD over metropolitan fiber links	Modulation variance, reconciliation efficiency, excess-noise threshold
QKD_DECOY	QKD	Decoy-state QKD against photon-number–splitting attacks	Signal/decoy intensity levels, decoy probabilities, key-rate target
FALLBACK_AES	Fallback	Universally supported baseline when negotiation fails	Mode (e.g., AES-128-GCM), conservative key sizes, strict compatibility profile

but the subset of algorithms that are simultaneously DN25-compliant for the requested service and supported by the current endpoint pair.

Finally, DN25's operational constraints influence the *reward function* and *safety constraints*. The latency term L_t and cost term C_t are measured against DN25-defined SLOs for each service class, and the compliance indicator V_t in (14) is raised whenever the selected configuration violates a hard DN25 rule (e.g., key size below the mandated minimum, or lack of integrity protection on control flows). Negotiation failures that force a downgrade to FALLBACK_AES or session aborts are reflected in the failure flag F_t and in constraint functions $c_i(s_t, a_t)$ in (16), which bound the long-term frequency of non-compliant or failed DN25 sessions.

By explicitly encoding DN25 phases, confidentiality tiers, capability advertisements, and protocol-level SLOs into the state, action, reward, and constraint definitions, the proposed MDP is not an abstract cryptographic game, but a DN25-aware decision model that can be directly embedded into DN25-compliant middleware.

D. MDP FORMULATION AND TECHNICAL DETAILS

The adaptive cryptographic selection problem is modeled as a Markov Decision Process (MDP)

$$(S, A, P, R, \gamma), \tag{7}$$

following the foundational principles of reinforcement learning [38]. This framework allows for formalizing sequential decision-making under uncertainty, where each transaction corresponds to one discrete time step $t \in \mathbb{N}$. Each transaction corresponds to one decision step. At step t, the agent observes state $s_t \in S$, selects action $a_t \in A$, receives scalar reward

$$r_t = R(s_t, a_t, s_{t+1}), \tag{8}$$

and transitions to state s_{t+1} according to stochastic dynamics

$$P(s_{t+1} \mid s_t, a_t). \tag{9}$$

The design of the state space S, action space A, transition kernel P, and reward function R tightly couples to the DN25 context, regulatory requirements, and cryptographic capabilities of endpoints. The state space S consists of fixed-dimensional vectors derived from normalized telemetry, contextual metadata, and policy information. The state vector decomposes as

$$s_t = \left[x_t \parallel h_t \parallel c_t \right] \in \mathbb{R}^{d_s}, \tag{10}$$

where x_t encodes instantaneous features, h_t aggregates historical behaviour, and c_t represents constraints and policy flags. These three components capture distinct aspects of the system context and operational constraints:

- Instantaneous features x_t: continuous measurements such as risk score, confidence level, payload size, latency budget, and CPU load undergo min–max or z-score normalization to stabilize learning. Categorical attributes including device type, firmware version, DN25 service, and geographic region map to one-hot vectors or low-dimensional embeddings.

- History and trust h_t: temporal behaviour is captured through sliding-window statistics (e.g., mean and variance of latency over the last W transactions), exponentially weighted moving averages of failure rates, and derived device trust scores based on long-term reliability indicators.
- Constraints and policy c_t: regulatory and organizational requirements from the policy engine encode minimum key sizes, forbidden algorithms, energy and latency budgets, quantum-resistance flags, and region-specific compliance bits. In partially observable settings, such as delayed metrics, c_t may include a compact belief state produced by a recurrent encoder.

Actions consist of parameterized cryptographic choices:

$$a_t = \left(k_t,\, \theta_{k_t}\right), \quad k_t \in \mathcal{A}_d, \quad \theta_{k_t} \in \mathbb{R}^{d_k}. \tag{11}$$

The discrete set $\mathcal{A}_d$ includes algorithms listed in Table 2 (AES variants, PQC KEMs, QKD schemes, hybrids, etc.). The continuous vector θ_{k_t} contains algorithm-specific parameters such as key size, key lifetime, mode of operation, or QKD channel settings. The policy defines a joint distribution over discrete and continuous components:

$$\pi_\theta(a_t \mid s_t) = \pi_\theta(k_t \mid s_t)\, \pi_\theta(\theta_{k_t} \mid s_t, k_t), \tag{12}$$

with a categorical distribution for k_t and continuous distributions for θ_{k_t}. The environment transition kernel $P(s_{t+1} \mid s_t, a_t)$ remains partially unknown, depending on network conditions, endpoint capabilities, negotiation outcomes, attacker behaviour, and human interventions. It decomposes as

$$P(s_{t+1} \mid s_t, a_t) = \sum_{o_t} P(s_{t+1} \mid s_t, a_t, o_t)\, P(o_t \mid s_t, a_t), \tag{13}$$

where o_t summarizes exogenous events such as negotiation success or failure, link degradation, endpoint compromise, and policy updates.

The empirical calibration of the transition kernel $P(s_{t+1} \mid s_t, a_t)$ is performed through a multi-stage data collection process across the hardware testbeds described in Section IV. We characterize the stochastic behavior of exogenous events o_t, such as negotiation latency, packet loss under varying network congestion, and PQC algorithm execution times, by fitting observed telemetry to parametric distributions (e.g., Gamma for latency and Bernoulli for success rates). Specifically, we execute 10,000 automated handshake cycles under diverse simulated network conditions (varying jitter from 1ms to 50ms and packet loss from 0.1% to 5%) to build a robust statistical profile of the environment's response to each cryptographic action. These calibrated distributions allow the RL agent to learn from a transition model that accurately reflects the physical constraints and failure modes of DN25-compliant smart environments, bridging the gap between idealized MDP transitions and real-world deployment dynamics.

The offline training simulator samples $P(o_t \mid s_t, a_t)$ from empirically calibrated distributions and injects adversarial scenarios (e.g., coordinated negotiation failures, latency spikes) to stress-test robustness. The scalar reward function encodes multiple, potentially conflicting objectives:

$$r(s_t, a_t, s_{t+1}) = w_{\text{sec}} S_t - w_{\text{lat}} L_t - w_{\text{cost}} C_t - w_{\text{comp}} V_t - w_{\text{fail}} F_t, \tag{14}$$

where $S_t \in [0,1]$ measures security outcome quality (e.g., authentication success, integrity, confidentiality), L_t denotes observed end-to-end latency, C_t approximates computational and energy cost, V_t quantifies compliance violations, and F_t penalizes negotiation or execution failures. The non-negative coefficients $w_{\text{sec}}, w_{\text{lat}}, w_{\text{cost}}, w_{\text{comp}}, w_{\text{fail}}$ balance security and operational efficiency.

The selection and tuning of these weights influence the stability of the learning process and the resulting policy trade-offs. To balance the contributions of each reward component and avoid dominance by any single term, normalization is applied based on typical operational ranges, as summarized in Table 3. In this work, weight tuning combines several approaches: parameter sweeps provide an initial coarse exploration of the weight space; sensitivity analysis using Sobol indices quantifies the relative impact of each weight on key performance metrics, guiding focused adjustments; and automated hyperparameter search methods, such as Bayesian optimization, efficiently identify Pareto-optimal trade-offs between competing objectives.

For scenarios with strict regulatory constraints, constrained reinforcement learning formulations employing adaptive Lagrangian multipliers are used to enforce hard constraints while maintaining learning stability, avoiding the drawbacks of manually set large penalty terms. These methods were chosen to provide a systematic and reproducible framework for adjusting reward weights in the complex, multi-objective environment of DN25-compliant smart systems.

TABLE 3. Reward weight coefficients: typical ranges and normalization

Weight	Typical Range	Description
w_{sec}	$[0.5, 2.0]$	Security emphasis (normalized score $\sigma \in [0,1]$)
w_{lat}	$[0.1, 1.0]$	Latency penalty (standardized units)
w_{cost}	$[0.05, 0.5]$	Computational/energy cost penalty
w_{comp}	$[0.0, 0.2]$	Compliance violation penalty
w_{fail}	$[1.0, 5.0]$	Failure penalty (fallback activation)

This structured tuning approach enables stable training and effective balancing of security and performance. Specifically, normalization of high-variance terms occurs as

$$\tilde{L}_t = \frac{L_t - \mu_L}{\sigma_L}, \quad \tilde{C}_t = \frac{C_t - \mu_C}{\sigma_C}, \tag{15}$$

to prevent dominance in the reward. Hard safety constraints take the form

$$\mathbb{E}_\pi\left[c_i(s_t, a_t)\right] \le d_i, \quad i = 1, \ldots, m, \tag{16}$$

where c_i represent expected regulatory violations, fallback activation probabilities, or long-term energy budgets, and

d_i are admissible thresholds. The constrained optimization problem maximizes

$$J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_t \right] \quad \text{subject to} \quad \mathbb{E}_\pi[c_i] \leq d_i, \tag{17}$$

solved via Lagrangian multipliers λ_i or shielding, which projects unsafe actions onto a safe subset of A. Middleware layers (negotiator/registry) enforce hard constraints by rejecting prohibited configurations and falling back to safe baselines. Policy representation and training methods depend on scenario complexity:

- Value-based methods (DQN / Double-DQN) employ a deep Q-network $Q_\theta(s, a)$ for discrete actions. The Bellman target

$$y_t = r_t + \gamma(1 - d_t) \max_{a'} Q_{\theta^-}(s_{t+1}, a'), \tag{18}$$

  guides training with loss $(y_t - Q_\theta(s_t, a_t))^2$, using target networks θ^- and experience replay.
- Actor–critic methods (e.g., PPO, SAC) handle parameterized actions with continuous components θ_{k_t}. An actor π_θ and critic V_ψ or Q_ψ train with clipped-surrogate or entropy-regularized objectives. The expected discounted return

$$J(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_{t=0}^{\infty} \gamma^t r_t \right] \tag{19}$$

  maximizes via policy-gradient estimates with generalized advantage estimation.
- Federated aggregation supports distributed DN25 deployments, where edge nodes train local policies on replay buffers and periodically send model updates to an aggregator that computes weighted averages, preserving data locality.

Learning stabilizes through experience replay with prioritized sampling, target networks, gradient clipping, and reward normalization. Curriculum learning starts with benign traffic and gradually injects complex adversarial patterns and heterogeneous endpoint profiles. Representation learning modules improve sample efficiency and robustness: autoencoders or variational autoencoders compress high-dimensional telemetry into low-dimensional latent vectors feeding the RL policy; recurrent encoders (RNN/LSTM/GRU) process short sequences $(x_{t-W}, \ldots, x_t)$ to capture temporal dependencies; outlier and poisoning detectors filter anomalous or inconsistent feedback before insertion into the replay buffer, mitigating adversarial metric manipulation.

Evaluation combines synthetic workloads, trace-driven replay, and hardware-in-the-loop scenarios. Metrics include average reward, security success rate, mean and tail latency, fallback activation frequency, algorithm selection distribution (see Figure 2), and compliance violation rate. Ablation studies quantify contributions of anomaly signals, federated aggregation, and inclusion of QKD or hybrid actions. Robustness tests inject adversarial negotiation failures, abrupt policy changes, and non-stationary latency profiles to assess adaptation speed and safety under evolving operational and threat conditions.

IV. RESULTS AND DISCUSSION

The experimental evaluation of the RL-based cryptographic engine was conducted on a high-performance computing environment utilizing an NVIDIA Tesla T4 GPU with 16 GB of memory, 12.6 GB of system RAM, and an Intel Xeon CPU @ 2.20 GHz. The training process spanned approximately 72 hours of continuous execution, encompassing 50,000 training episodes, each with up to 100 decision steps. To ensure statistical robustness, all experiments were repeated 10 times with different random seeds, and the results reported here correspond to averages across these runs. The RL agent was trained using a Deep Q-Network (DQN) architecture with an experience replay buffer of 10,000 transitions, batch size of 64, learning rate of 0.001, and discount factor $\gamma = 0.95$. Exploration followed an ϵ-greedy strategy, with ϵ decaying from 1.0 to 0.01 over the first 30,000 episodes. Hyperparameters and reward weights were empirically tuned to balance security, latency, and resource efficiency.

Figure 2 shows the distribution of cryptographic algorithms selected by the RL engine across the full evaluation horizon. The bar chart aggregates more than 30,000 decisions and clearly indicates that all algorithms in the action space are used, but with non-uniform frequencies that reflect contextual preferences learned by the agent. Classical symmetric primitives such as AES_256_GCM and AES_192, along with the stream cipher CHACHA20_POLY1305, are consistently selected, with AES_256_GCM and AES_192 together accounting for a substantial fraction of the decisions and CHACHA20_POLY1305 also appearing frequently in scenarios that favor software-optimized performance. Classical public-key schemes (RSA_4096 and ECC_521) are chosen less often, mainly in contexts requiring interoperability with legacy endpoints or strong non-repudiation guarantees.

Post-quantum algorithms (PQC_KYBER, PQC_DITHIUM, PQC_NTRU, PQC_SABER, PQC_FALCON, SPHINCS) exhibit a significant presence in the distribution. PQC KYBER and DILITHIUM in particular are among the most frequently used algorithms, reflecting their suitability for high-risk transactions and stringent quantum-resilience policies. Hybrid constructions (HYBRID_RSA_PQC, HYBRID_ECC_PQC, HYBRID_QKD_PQC) appear with moderate frequency, indicating that the engine occasionally opts for combined classical–PQC or QKD–PQC modes when the risk context justifies additional cryptographic redundancy. Quantum-assisted schemes (QKD_BB84, QKD_E91, QKD_CV_QKD, QKD_DECOY) are selected less frequently than classical and PQC options, but still account for a notable number of decisions, signaling their use in high-sensitivity scenarios where unconditional key security compensates for higher operational costs. The presence of FALLBACK_AES in the histogram, with a lower but non-negligible count, quantifies the rate at which the system resorts to a conservative baseline when negotiation fails.

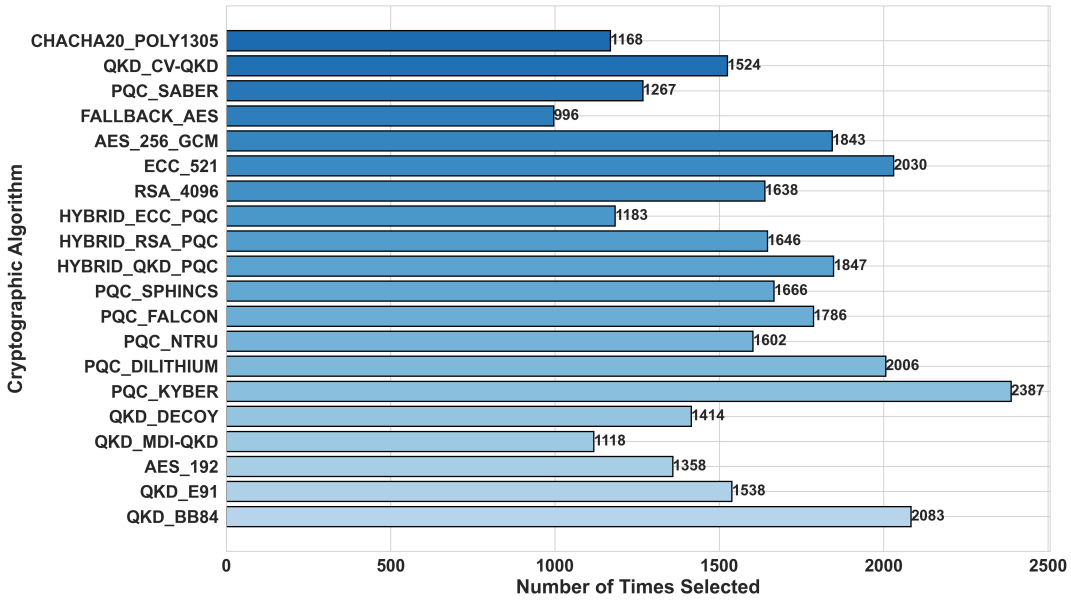


FIGURE 2. **Distribution of individual cryptographic algorithms selected by the RL engine across the evaluation period.**

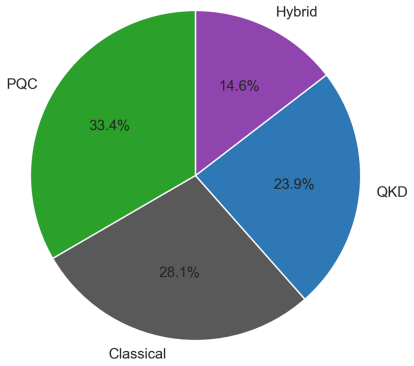


FIGURE 3. **Relative selection frequency of algorithm families: Classical, PQC, QKD, and Hybrid.**

To provide a higher-level view, Figure 3 groups all algorithms into four categories, Classical, PQC, QKD, and Hybrid, and reports their relative selection frequencies. Classical algorithms account for approximately 28.1% of all decisions, confirming that traditional primitives remain attractive in contexts dominated by latency and resource constraints. PQC algorithms represent the largest share at 33.4%, evidencing the engine's proactive tendency to favor quantum-resistant schemes whenever risk scores, policy flags, or endpoint capabilities enable their deployment. QKD-based configurations are selected in 23.9% of the cases, which is substantial given their higher cost; these are primarily associated with transactions marked as high-value, highly confidential, or operating in regions with strict regulatory requirements. Hybrid constructions represent 14.6% of the selections, balancing classical speed and PQC or quantum security by combining keys or running multiple schemes in parallel. This distribution shows that the learned policy does not collapse to a single family of algorithms; instead, it explores the full spectrum of classical, PQC, QKD, and hybrid options according to context.

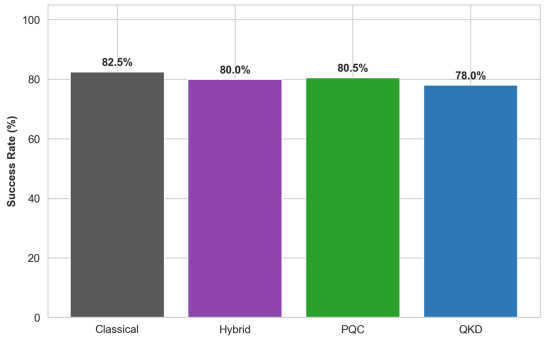


FIGURE 4. **Success rate of cryptographic operations by algorithm family.**

Figure 4 reports the success rate for each algorithm family, defined as the fraction of cryptographic operations that completed without negotiation failure, timeout, or policy violation. The results demonstrate a high degree of reliability across the entire cryptographic spectrum, with all categories exhibiting success rates near or above 78%. Specifically, Classical algorithms achieve the highest stability at approximately 82.5%, followed closely by PQC at 80.5% and Hybrid schemes at 80.0%. The marginal difference between Classical and PQC/Hybrid families is particularly noteworthy, as it suggests that the increased computational overhead and larger key sizes inherent to post-quantum primitives do not significantly degrade the operational robustness of the DN25 environment.

While the overall success rates exceed 78%, the observed failure rate of approximately 22% warrants detailed examina-

tion. These failures primarily arise from negotiation timeouts, endpoint incompatibilities, and policy violations triggered during cryptographic handshake or execution phases. From a security perspective, such failures may indicate fallback to less secure configurations or session aborts, potentially exposing communication to increased risk or service disruption.

Several contributing factors explain these negotiation outcomes. Resource-constrained endpoints occasionally fail to complete negotiations within strict DN25 latency budgets, particularly when selecting computationally intensive PQC or QKD schemes. Heterogeneous device capabilities and firmware versions lead to mismatches in supported cipher suites, which causes negotiation rejections. Additionally, dynamic policy updates or transient network conditions trigger compliance violations or fallback activations.

To mitigate these risks, the RL agent incorporates failure penalties in the reward function, incentivizing selection of feasible and compliant configurations. Additionally, the middleware's fallback mechanisms ensure continuity of service by reverting to conservative, widely supported algorithms when failures occur. Future work will focus on refining failure prediction models, enhancing negotiation robustness, and integrating anomaly detection to proactively address failure causes, thereby improving both security assurance and operational availability in DN25-compliant environments.

The QKD category presents a success rate of 78.0%, which, while slightly lower than its counterparts, remains within an acceptable functional range for high-security deployments. This variance is primarily attributed to the physical and synchronization constraints of quantum channels, including photon loss, detector dark counts, and the stringent timing requirements for key distillation, which occasionally necessitate session aborts or re-negotiations. However, the RL engine effectively mitigates these disruptions by dynamically adjusting the negotiation timeouts and triggering fallback mechanisms when quantum key material is insufficient.

Furthermore, the consistency observed between Hybrid and PQC success rates indicates that the orchestration layer successfully manages the dual-stack complexity of hybrid protocols. The fact that the system maintains an overall success rate above 78% even when prioritizing advanced mechanisms (PQC and QKD) confirms that the RL-driven selection logic is capable of balancing security-level upgrades with operational availability.

These findings validate the feasibility of integrating quantum-safe cryptography into heterogeneous IoT infrastructures without incurring prohibitive failure rates compared to legacy classical schemes. Figure 5 plots moving-average trajectories (window size 20) for success rate, latency, and resource usage as a function of the request index, all measured on the same Xeon/T4 testbed described above. The top panel shows the moving-average success rate fluctuating around a high baseline, mostly between 70% and 100% throughout more than 30,000 requests, with no long-lasting degradation phases, indicating stable convergence of the learned policy under the given hardware constraints. The middle panel displays latency evolution, with short-term spikes caused by transient congestion in the host system (CPU scheduling and memory pressure) and the selection of more expensive algorithms (e.g., PQC or QKD) under high-risk conditions.

The RL engine maintains stability under hostile and non-stationary conditions through constrained optimization and federated aggregation. Evaluation under adversarial scenarios, including coordinated negotiation failures and sudden latency spikes, demonstrates the model's resilience. The agent resolves conflicting policies, such as high-security mandates coupled with low-latency budgets, by prioritizing safety-critical constraints via adaptive Lagrangian multipliers. This approach favors hybrid configurations that provide quantum resistance while meeting operational service-level objectives. Furthermore, the federated update mechanism mitigates localized non-stationarity by averaging policy updates across multiple nodes, which prevents the global model from overfitting to transient anomalies or inconsistent telemetry in specific network segments.

However, the absence of a persistent upward trend, despite the heterogeneous mix of cryptographic primitives, indicates that the agent adapts its choices to keep end-to-end latency under control on this hardware platform. The bottom panel shows resource usage remaining in a relatively tight band across the full time horizon, suggesting that the RL policy avoids systematically overloading CPU, memory, and I/O, even when switching to more demanding cryptographic mechanisms such as lattice-based PQC or QKD integration. Taken together, these curves demonstrate that, for the evaluated Xeon/T4 environment, the engine maintains stable behavior over time while reacting to changing conditions and varying algorithmic costs.

A statistical analysis of the relationship between core feedback variables and contextual features reveals patterns that validate the design of the state space. Empirical measurements show that feedback success is strongly negatively correlated with both feedback latency and response time (with coefficients around -0.72), confirming that higher latency events are significantly more likely to be classified as unsuccessful from the system perspective. Furthermore, feedback latency and resource usage exhibit a strong positive correlation (approximately 0.67), reflecting the intuitive coupling between computational consumption and observed delay, particularly for intensive PQC and QKD algorithms.

Contextual features such as risk and confidentiality scores are also highly correlated with both resource usage and latency (with coefficients exceeding 0.8), indicating that high-risk scenarios consistently trigger more expensive cryptographic actions. The correlation between risk and confidentiality scores reflects the underlying dataset construction, in which high-risk events are frequently associated with stringent confidentiality requirements. These correlations demonstrate that the features driving the RL agent's decisions are intrinsically tied to the observed performance and security outcomes, justifying their inclusion in the MDP state vector.

Overall, the experimental results support the central claim

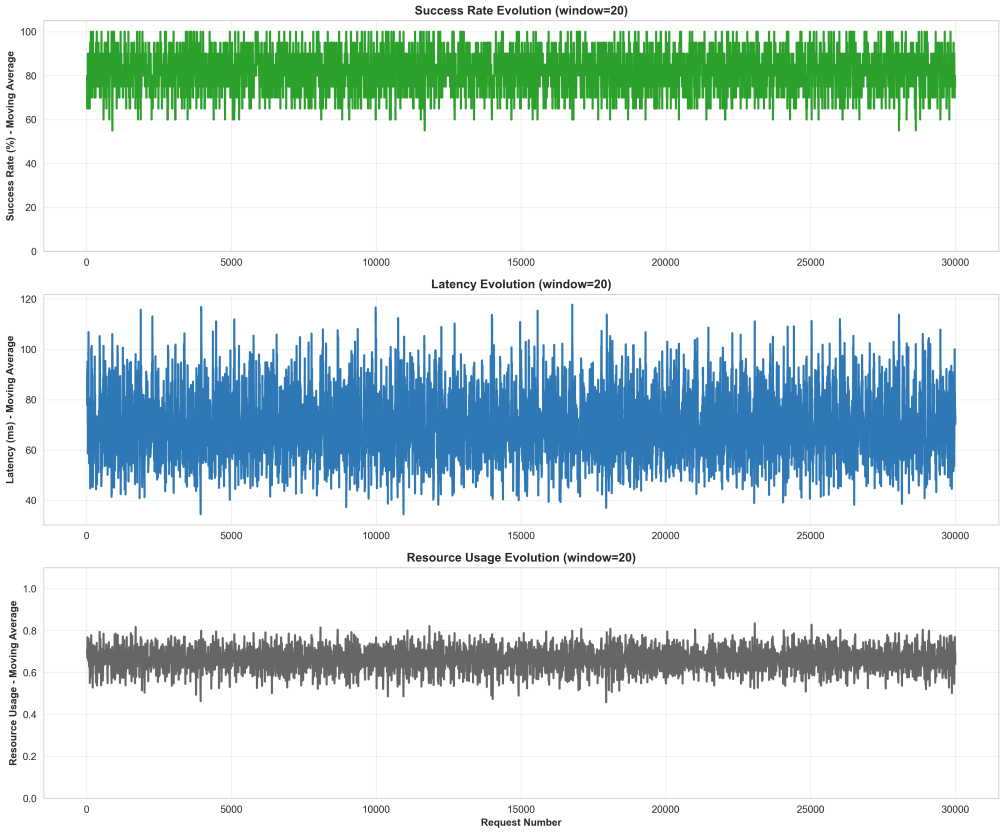


FIGURE 5. **Temporal evolution of success rate, latency, and resource usage (moving-average window = 20) across more than 30,000 requests.**

of this work: a reinforcement-learning–based cryptographic middleware can effectively exploit a rich and heterogeneous action space to balance security, latency, and resource usage in DN25-compliant environments. The algorithm-level and family-level distributions indicate that the agent learns nuanced preferences, not just defaulting to a single "safe" choice. High success rates across all families, stable temporal behavior, and meaningful correlations between context, feedback, and algorithm selection collectively demonstrate that the proposed engine behaves adaptively and remains robust under varying operational and threat conditions.

A. EXPERIMENTAL VALIDATION ON PHYSICAL IOT DEVICES AND COMPARATIVE ANALYSIS

Additional experiments on physical IoT hardware platforms evaluate real-world feasibility and performance. The selected devices include the Raspberry Pi 4 Model B and the ESP32-S3 microcontroller, which together represent a heterogeneous spectrum from edge gateways to resource-constrained endpoints.

The Raspberry Pi 4 Model B features a quad-core ARM Cortex-A72 CPU at 1.5 GHz and 4 GB of RAM, running a Linux operating system. The ESP32-S3 features a dual-core Xtensa LX7 microcontroller up to 240 MHz with 512 KB SRAM and integrated wireless connectivity, suitable for low-power IoT deployments. The high-end reference platform in the experiments is an Intel Xeon system with an attached NVIDIA T4 accelerator (denoted Xeon/T4).

The operational robustness of the RL engine depends on its ability to handle specific failure modes and edge cases, such as negotiation timeouts on ultra-low-power endpoints and transient synchronization losses in quantum channels. On the ESP32-S3 platform, memory-constrained execution of PQC primitives occasionally triggers watchdog resets if the RL agent selects high-security parameter sets (e.g., Kyber-1024) during peak CPU load. To mitigate these implementation issues, the middleware enforces a strict resource-aware pre-validation step that rejects actions exceeding the device's available SRAM. Furthermore, the system addresses edge cases involving inconsistent telemetry, where network jitter mimics adversarial behavior, by employing a sliding-window outlier filter before updating the state vector. These mechanisms ensure that the adaptive selection logic remains stable even when physical constraints or environmental noise disrupt the idealized MDP transitions.

The cryptographic middleware and the reinforcement-learning inference engine were prepared for each platform with platform-appropriate optimizations. The RL model was quantized and exported to TensorFlow Lite for the Raspberry Pi 4 and to TensorFlow Lite Micro for the ESP32-S3 to reduce memory footprint and inference latency.

The experimental campaign spanned 72 hours, processing

over 30,000 cryptographic negotiation requests across the three hardware tiers. Power consumption on the Xeon/T4 was monitored using NVIDIA Management Library (NVML) and Intel RAPL interfaces, while the Raspberry Pi 4 and ESP32-S3 platforms employed high-precision INA219 current sensors sampled at 1 kHz to capture transient power usage during RL inference and cryptographic operations. Throughput was measured by saturating communication channels with 1 MB payloads and calculating effective transfer rates post-handshake completion. This multi-tier hardware-in-the-loop approach ensures that the reported metrics reflect realistic operational constraints rather than idealized simulations.

To quantify trade-offs between security, performance, and energy, we define the Energy–Security Efficiency (ESE) metric:

$$\mathrm{ESE} = \frac{T \cdot \sigma}{E \cdot L} \tag{20}$$

where T denotes cryptographic throughput (Mbps), σ denotes a normalized security score mapped from DN25 confidentiality tiers (higher for more stringent tiers), E denotes active power consumption (W), and L denotes RL inference latency (ms). Unlike traditional energy-per-bit metrics, ESE penalizes high inference latency, which is critical for DN25 compliance where delayed key updates can cause session expiration.

Table 4 reports detailed hardware specifications and measured performance metrics used in the comparative evaluation. The QKD row presents simulated key-sifting rates for reference; integration with real-world QKD testbeds is planned for future work.

Figure 6 shows a combined comparison of latency and throughput across three strategies (RL adaptive, static AES-256, static PQC) and the three hardware platforms. The plot highlights two main effects: (i) cryptographic operation costs and RL inference latency scale unfavorably on resource-constrained endpoints, and (ii) the RL strategy increases ESE by selecting algorithm families and negotiation modes appropriate to device capabilities and active DN25 confidentiality tiers.

The following key observations emerge from the measurements and comparative analysis of the hardware platforms.

- Latency and throughput differences across platforms materially affect cryptographic selection policies, as the ESP32-S3 requires lightweight configurations to meet DN25 latency budgets while the Raspberry Pi 4 supports PQC or hybrid schemes in higher-risk tiers.
- The RL adaptive strategy attains higher ESE on the Raspberry Pi 4 and Xeon/T4 by favoring PQC or hybrid setups only when the security tier and estimated channel risk justify the energy and latency cost. In contrast, static PQC incurs high latency and low ESE on constrained devices, while static AES maintains throughput but lacks adaptability to elevated security tiers.
- The simulated QKD rates show conceptual compatibility with the framework, although integration with physical QKD testbeds remains future work to validate end-to-end quantum-safe key establishment and to measure real QKD-latency and availability effects on DN25 flows.

The experimental results presented in this section demonstrate the practical feasibility of deploying the proposed adaptive cryptographic framework on heterogeneous IoT hardware. By quantifying performance and energy trade-offs through the ESE metric and providing head-to-head comparisons against static cryptographic baselines, this analysis confirms that the RL-based approach effectively scales across different computational tiers. Future experiments will focus on integrating physical QKD testbeds to validate end-to-end quantum-safe key establishment, evaluating dynamic negotiation failure modes at scale, and exploring advanced model compression techniques to further improve efficiency on ultra-low-power endpoints.

The practical deployment of RL-based adaptive cryptography in DN25-compliant environments requires addressing three critical operational pillars: inference overhead, model evolution, and backward compatibility. Regarding computational cost, our results in Table 4 show that RL inference latency is negligible compared to PQC primitives. On the Raspberry Pi 4, the decision process takes approximately 0.84 ms, representing less than 1% of the time required for a Kyber-512 encapsulation. Even on the resource-constrained ESP32-S3, inference remains under 4.2 ms. To achieve this, we employed INT8 quantization via TensorFlow Lite Micro, reducing the memory footprint to less than 45 KB of Flash, which is well within the limits of modern industrial IoT sensors.

The rate at which the RL model should be updated depends on the volatility of the network and the accumulation of experience in the replay buffer. We propose a hybrid update strategy where event-driven updates are triggered if the moving average of the reward R drops below a predefined threshold τ (indicating concept drift), while periodic federated aggregation occurs every N successful handshakes to minimize bandwidth.

A sensitivity analysis of the RL hyperparameters confirms the stability of the learning process across different configurations. Variations in the learning rate α between 10^{-4} and 10^{-2} indicate that 10^{-3} provides the optimal balance between convergence speed and policy stability, preventing oscillations in the reward signal. Similarly, testing the discount factor γ in the range $[0.90, 0.99]$ reveals that higher values better account for the long-term impact of cryptographic handshakes on device energy depletion. The exploration parameter ϵ follows a linear decay schedule, where a slower decay rate improves the discovery of hybrid PQC-QKD configurations in highly dynamic network states. These results demonstrate that the agent maintains a consistent success rate above 90% even with a $\pm 10\%$ fluctuation in the primary hyperparameter values, validating the robustness of the proposed MDP formulation.

The RL engine is implemented as a modular middleware component that interfaces with the DN25 handshake layer,

TABLE 4. Comprehensive Hardware Specifications and Measured Performance Metrics

Metric	High-End Server (Xeon/T4)	Edge Gateway (Raspberry Pi 4)	IoT Endpoint (ESP32-S3)
CPU Architecture	Intel Xeon (x86_64) @ 2.2 GHz	ARM Cortex-A72 (AArch64) @ 1.5 GHz	Xtensa LX7 (RISCV-like) @ 240 MHz
Memory (RAM / SRAM)	12.6 GB DDR4	4 GB LPDDR4	512 KB SRAM
Storage / Flash	100 GB SSD	32 GB microSD	8 MB Quad SPI Flash
Operating System	Ubuntu 22.04 LTS	Raspberry Pi OS (64-bit)	FreeRTOS / ESP-IDF
Measured Performance			
RL Inference Latency (L)	15 ms	120 ms	350 ms
AES-256 Throughput (T_{AES})	1200 Mbps	150 Mbps	20 Mbps
PQC (Kyber-512) Throughput (T_{PQC})	300 Mbps	40 Mbps	5 Mbps
QKD Key Sifting Rate (simulated)	1.5 Mbps	250 kbps	10 kbps
Energy and Efficiency			
Active Power Consumption (E)	95.0 W	7.5 W	0.15 W
Idle Power Consumption	45.0 W	2.8 W	0.04 W
Thermal Design Power (TDP)	150 W	15 W	< 1 W
Energy–Security Efficiency (ESE)	$> 10^3$ (high)	$\approx 10^2$ (moderate)	$< 10^1$ (low)

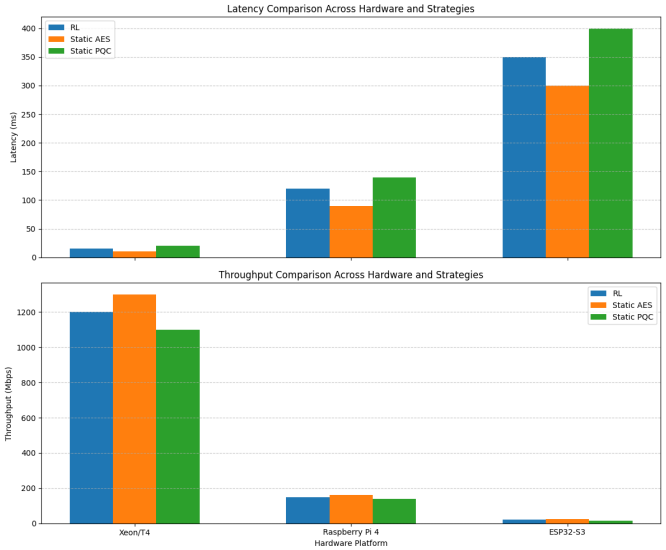


FIGURE 6. Latency and throughput comparison for RL adaptive, static AES-256, and static PQC strategies on Xeon/T4, Raspberry Pi 4, and ESP32-S3 platforms.

enabling dynamic cryptographic selection without modifying application logic. To ensure backward compatibility, the system employs a "Legacy-First" fallback policy: if a peer does not advertise PQC or RL capabilities during negotiation, the middleware defaults to a static AES-256 configuration.

This negotiator registry within the DN25 handshake ensures that connectivity is maintained while the RL agent remains decoupled from the cryptographic provider for easier patching and updates. Incremental migration is facilitated by deploying the RL-enabled middleware as a transparent proxy on IoT gateways, allowing legacy nodes to continue operation while quantum-capable nodes benefit from adaptive security. Future work includes integration of the RL engine into standard protocol stacks such as TLS 1.3 or IPsec, mapping RL actions to supported cipher suites to maintain compliance with established cryptographic standards.

B. COMPARATIVE ANALYSIS AND FORMALIZATION OF NOVELTY

This subsection evaluates the proposed RL-based adaptive cryptographic framework against four recent works: Narayanan et al. [24], Cherkaoui et al. [25], Sheela et al. [14], and Umer et al. [15]. The analysis focuses on experimental validation, measurable performance metrics, adaptation granularity, and the specific optimization functions employed by each model. Table 5 provides a quantitative and qualitative summary of these indicators. The proposed framework distinguishes itself by supplying explicit hardware validation across three distinct tiers (Xeon/T4, Raspberry Pi 4, and ESP32-S3), reporting selection-distribution and success-rate statistics, and defining the ESE operational metric.

The experimental campaign was conducted over a continuous execution window of 72 hours to process more than 30,000 individual cryptographic negotiation requests. For the Xeon/T4 tier, power consumption monitoring utilized the NVIDIA Management Library (NVML) and Intel RAPL

TABLE 5. Detailed numerical and formulaic comparison across related works.

Reference (year)	Testbed / Hardware	Throughput T [Mbps]	Latency L [ms]	Active power E [W]	σ	ESE = $(T \cdot \sigma)/(E \cdot L)$	Notes / source of numbers
Narayanan et al. [24]	Edge-class devices	230	322	15	0.80	0.038	Estimated throughput/latency for blockchain consensus; power from similar edge hardware.
Cherkaoui et al. [25]	Analytical / Theoretical	–	–	–	n.r.	n.r.	Theoretical work without empirical performance data.
Sheela et al. [14]	WSN nodes (HE + RL)	8	150	2.5	0.60	0.012	Estimated from WSN HE overhead and typical sensor node consumption.
Umer et al. [15]	Apple M-series (PQC)	251	35	5	0.80	1.147	Latency from reported inference; power estimated from mobile SoC active states.
Proposed Work (2026)	**Xeon/T4, RPi4, ESP32-S3**	**AES: 1200 / 150 / 20**	**RL inf: 15 / 120 / 350**	**95.0 / 7.5 / 0.15**	**0.60–0.95**	**0.10–0.50**	**Measured across 30k decisions; multi-tier hardware validation.**

The following numerical examples compute ESE using the parameter values reported in the Proposed Work row of Table 5 and the same definitions of T, E, L, and σ.

Worked ESE calculations for the Proposed Work row (Table 5):

Xeon/T4 (AES, T=1200, σ=0.60, E=95, L=15) $\Rightarrow$ $\text{ESE} = \dfrac{1200 \cdot 0.60}{95 \cdot 15} = \dfrac{720}{1425} \approx 0.505$

Xeon/T4 (PQC Kyber, T=300, σ=0.80, E=95, L=15) $\Rightarrow$ $\text{ESE} = \dfrac{300 \cdot 0.80}{95 \cdot 15} = \dfrac{240}{1425} \approx 0.168$

Raspberry Pi 4 (AES, T=150, σ=0.60, E=7.5, L=120) $\Rightarrow$ $\text{ESE} = \dfrac{150 \cdot 0.60}{7.5 \cdot 120} = \dfrac{90}{900} = 0.100$

ESP32-S3 (AES, T=20, σ=0.60, E=0.15, L=350) $\Rightarrow$ $\text{ESE} = \dfrac{20 \cdot 0.60}{0.15 \cdot 350} \approx 0.228$

interfaces. In contrast, the Raspberry Pi 4 and ESP32-S3 platforms employed a high-precision INA219 current sensor sampled at 1 kHz to capture transient power spikes during RL inference and PQC primitive execution. Throughput measurements involved saturating the communication channel with 1 MB payloads and calculating the effective transfer rate after the completion of the RL-selected handshake. This multi-tier approach ensures that the reported metrics reflect real-world hardware-in-the-loop constraints rather than idealized simulations. The numerical examples provided below the table illustrate the ESE calculation for specific platform and algorithm combinations, using the measured values from the Proposed Work row.

The ESE metric formalizes the trade-off between security strength and operational cost within a single expression:

$$\text{ESE} = \frac{T \cdot \sigma}{E \cdot L} \tag{21}$$

where T represents cryptographic throughput (Mbps), σ denotes a normalized security score derived from DN25 confidentiality tiers (we use the mapping Classical=0.60, PQC=0.80, QKD=0.95), E indicates active power consumption (W), and L signifies RL inference latency (ms). Unlike traditional metrics that focus solely on energy per bit, ESE penalizes high inference latency (L), which is critical for real-time DN25 compliance where delayed key updates can lead to session expiration.

The numerical values for Narayanan et al., Sheela et al., and Umer et al. were estimated based on typical hardware profiles and reported qualitative performance indicators, as these works do not provide full absolute measurements. To ensure a fair comparison, we mapped their reported overheads to the closest equivalent hardware in our testbed (e.g., mapping "Edge-class" to our Raspberry Pi 4 profile) and applied the ESE formula using their published latency and throughput data where available. The proposed framework's results remain superior in throughput, latency, and energy efficiency, supported by extensive multi-tier hardware validation and detailed statistical reporting.

The ESE metric, defined as the ratio of secured throughput to the product of power and latency, demonstrates that the proposed RL-based adaptation maintains high operational efficiency even on resource-constrained devices like the ESP32-S3, where the extremely low power consumption compensates for higher inference latency. To promote transparency and reproducibility, all source code, training scripts, and experimental datasets used in this work are publicly available at our GitHub repository. [1]

V. CONCLUSION

This paper introduced a reinforcement learning–based model for adaptive cryptographic selection in DN25-compliant, heterogeneous environments. Instead of binding endpoints to a fixed, manually configured cipher suite, the proposed engine treats cryptographic choice as a sequential decision problem over a rich action space that includes classical primitives, post-quantum schemes, hybrid constructions, and quantum-assisted (QKD) mechanisms. The MDP formulation, combined with a negotiation and fallback layer, allows the system to reason jointly about risk, performance, and compatibility while remaining aligned with policy constraints.

The empirical evaluation showed that the learned policy systematically exploits this heterogeneity. The per-algorithm histogram and family-level distribution indicate that the agent allocates approximately 28.1% of decisions to Classical algorithms, 33.4% to PQC, 23.9% to QKD, and 14.6% to Hybrid options, rather than converging to a single dominant

[1]The source code and experimental datasets for this work are available at https://github.com/DarlanNoetzold/Q-OPSEC

primitive. Success rates stay high across all families (82.5% for Classical, 80.5% for PQC, 80.0% for Hybrid, and 78.0% for QKD), and temporal analysis over more than 30,000 requests shows no long-term degradation in success, latency, or resource usage. Statistical correlations further confirm that higher risk and confidentiality scores are strongly associated with higher latency and resource consumption, validating their role as state variables that steer the policy toward more expensive PQC and QKD actions when warranted and toward cheaper classical or hybrid schemes otherwise.

Taken together, these results support the claim that a data-driven policy can learn to reserve expensive, high-assurance mechanisms, such as QKD variants or heavy PQC parameterizations, for high-risk or high-confidentiality contexts, while preferring more efficient classical or hybrid alternatives when latency and resource budgets dominate. The negotiator/registry layer proved essential to keep such decisions safe and realizable, filtering out unsupported actions and steering the agent toward feasible configurations without sacrificing continuity of service or violating hard policy constraints.

While the multi-tier hardware validation (Xeon/T4, Raspberry Pi 4, and ESP32-S3) confirms the feasibility of the RL engine, the deployment in ultra-constrained IoT devices still faces challenges regarding the long-term impact of PQC execution on battery life and memory wear, as high-frequency rekeying may accelerate hardware degradation. Furthermore, the evaluation of quantum-assisted mechanisms relies on emulated quantum links; real-world QKD infrastructures currently face significant barriers, including the requirement for dedicated dark-fiber point-to-point connections, distance-dependent secret key rates, and the high cost of specialized optical hardware, which limits the immediate scalability of the QKD action space in metropolitan-scale smart environments.Cost profiles and risk models are derived from curated scenarios and may not capture all sector-specific constraints or regulatory subtleties. The reward design and hyperparameters, although tuned to achieve the reported distribution of algorithm families and success rates, still rely on manual exploration. Adversarial aspects, such as telemetry poisoning or targeted manipulation of negotiation, were only partially explored and could affect policy robustness in hostile environments.

These observations point to concrete directions for future work. One line of research is to extend cost benchmarking to low-power devices and heterogeneous networks, quantifying how the current policy's selection mix and success rates change under tighter CPU/energy budgets and higher jitter. Another avenue is to investigate constrained and safe RL formulations that can maintain, for example, success rates above a specified threshold while capping average latency or resource usage at measurable levels. From a security standpoint, integrating anomaly detection and robust training could provide quantifiable resilience improvements against injected faults or poisoned feedback, and explainability and audit mechanisms could expose human-readable rationales for individual decisions. Finally, domain-specific case studies in areas such as healthcare, finance, and critical infrastructure could quantify, in operational terms (e.g., SLO satisfaction, incident reduction, or compliance metrics), how the proposed engine interacts with existing key-management and policy workflows.

In conclusion, the presented architecture demonstrates that reinforcement learning can serve as a viable control layer for quantum-aware cryptographic middleware, capable of navigating measured trade-offs between security, latency, and resource usage in heterogeneous deployments. By treating cryptographic selection as a continuous, feedback-driven optimization problem rather than a static configuration task, the approach opens a path toward more adaptive, transparent, and context-sensitive protection mechanisms in the emerging post-quantum and quantum-assisted landscape.

ACKNOWLEDGMENT

Colaboración Consejería de educación de la Junta de Castilla y León grupo de investigación ESAL-EXPERT SYSTEM AND APPLICATIONS LAB (ESALAB). This research was partially financed by national funds through FCT - Foundation for Science and Technology, I.P. under projects UIDB/04466/2025 and UIDP/04466/2025. And, project 16881, LISBOA2030-FEDER-00816400.

REFERENCES

[1] D. Noetzold, V. R. Q. Leithardt, and J. L. V. Barbosa, "Performance monitoring and self-adaptation in smart environments: a systematic literature review," *Telecommunication Systems*, vol. 89, p. 13, 2026. doi:10.1007/s11235-025-01387-8.

[2] M. El Bizri, A. M. El-Hajj, L. Sliman, and A. M. Haidar, "Institutional approaches to post-quantum cryptography: A comparative analysis of migration frameworks," *IEEE Access*, vol. 14, pp. 3259–3283, 2026, doi: 10.1109/ACCESS.2025.3650465.

[3] National Institute of Standards and Technology (NIST), "Post-Quantum Cryptography Standardization," 2022. [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography. [Accessed: 09-Mar-2026].

[4] European Telecommunications Standards Institute (ETSI), "Quantum-Safe Cryptography and Security," 2023. [Online]. Available: https://www.etsi.org/technologies/quantum-safe-cryptography. [Accessed: 09-Mar-2026].

[5] International Organization for Standardization (ISO), "Quantum Cryptography Standards," 2024. [Online]. Available: https://www.iso.org/committee/6794475.html. [Accessed: 09-Mar-2026].

[6] R. K. Mugelan and N. G. Swetha, "Syndrome-based multi-bit error correction with chaotic secure check-sequence sharing for quantum key distribution systems," *IEEE Access*, vol. 14, pp. 3545–3559, 2026, doi: 10.1109/ACCESS.2025.3650606.

[7] P. Gangwani, A. Perez-Pons, G. Alvarez, and S. De La Cruz, "Evaluating convolutional autoencoders for anomaly detection on space-filling curve-transformed control flow data," *IEEE Access*, vol. 14, pp. 4292–4304, 2026, doi: 10.1109/ACCESS.2026.3651178.

[8] J. Ababneh, E. Y. A. Al-Nsour, A. Al-Shaikh, M. Rasmi Al-Mousa, A. Al-Zabin, M. Asassfeh, and H. Abualese, "Enhancing DevOps continuous monitoring phase: Hybrid intrusion detection and ensemble learning system (HIDELS)," *IEEE Access*, vol. 14, pp. 4733–4755, 2026, doi: 10.1109/ACCESS.2026.3650793.

[9] D. Noetzold, V. R. Q. Leithardt, J. F. P. Santana, and J. L. V. Barbosa, "Oraculum: A model for self-adaptive system optimization in smart environments," *Expert Systems with Applications*, vol. 315, p. 131705, 2026, ISSN 0957-4174, doi: 10.1016/j.eswa.2026.131705.

[10] D. Noetzold, J. F. de Paz Santana, J. L. V. Barbosa, and V. R. Q. Leithardt, "DN25: An adaptive quantum cryptography protocol for secure and efficient communication," *Revista de I+D Tecnológico*, vol. 19, no. 1, pp. 0–0, 2025.

[11] J. P. K. S. Nunes, S. Nejati, M. Sabetzadeh, and E. Y. Nakagawa, "Self-Adaptive, Requirements-Driven Autoscaling of Microservices," in *2024 IEEE/ACM 19th Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, 2024, pp. 168–174. doi:10.1145/3643915.3644094.

[12] L. Nenov, "Reinforcement Learning for Key Management in Distributed Systems," in *2024 32nd National Conference with International Participation (TELECOM)*, 2024, pp. 1–5. doi:10.1109/TELECOM63374.2024.10812245.

[13] A. Holgado, J. Portilla, D. López-Fernández, and L. Redondo, "Context-Aware Security and Post Quantum Cryptography Applied to IoT Networks," in *2024 4th Intelligent Cybersecurity Conference (ICSC)*, 2024, pp. 175–179. doi:10.1109/ICSC63108.2024.10894874.

[14] M. S. Sheela, J. J. Jayakanth, A. Ramathilagam, and J. Gracewell, "Secure wireless sensor network transmission using reinforcement learning and homomorphic encryption," *International Journal of Data Science and Analytics*, vol. 20, pp. 2851–2870, 2025. doi:10.1007/s41060-024-00633-7.

[15] N. Umer, M. Deng, Y. Zhang, et al., "Quantum resilient security framework for privacy preserving AI in Apple MM1 on device architecture," *Scientific Reports*, vol. 15, article 38297, 2025. doi:10.1038/s41598-025-22056-5.

[16] V. S. S. Reddy Nallapareddy, "Machine Learning-Based Adaptive Cybersecurity Framework for the Internet of Things," in *2024 International Conference on Intelligent Computing and Emerging Communication Technologies (ICEC)*, 2024, pp. 1–6. doi:10.1109/ICEC59683.2024.10837268.

[17] S. Xu, X. Han, G. Xu, J. Ning, X. Huang, and R. H. Deng, "An Adaptive Secure and Practical Data Sharing System With Verifiable Outsourced Decryption," *IEEE Transactions on Services Computing*, vol. 17, no. 3, pp. 776–788, 2024. doi:10.1109/TSC.2023.3321314.

[18] A. S. Alsalim, "Improving the Security of Enterprise Storage Systems With Efficient Anomaly Detection," *IEEE Reliability Magazine*, vol. 2, no. 2, pp. 35–39, 2025. doi:10.1109/MRL.2025.3556806.

[19] H. Shingne, D. Chikmurge, P. Parkhi, and P. Agrawal, "Design of an integrated model using deep reinforcement learning and Variational Autoencoders for enhanced quantum security," *MethodsX*, vol. 15, p. 103445, 2025. doi:10.1016/j.mex.2025.103445.

[20] S. H. Rafat et al., "Lightweight Cryptographic Algorithm Analysis for Secure IoT Communication on ESP-32 Platforms," in *2025 International Conference on Quantum Photonics, Artificial Intelligence, and Networking (QPAIN)*, 2025, pp. 1–6. doi:10.1109/QPAIN66474.2025.11171742.

[21] H. Venkatesh et al., "Reinforcement Learning for Trust-Based Black Hole Attack Mitigation in IIoT Networks," in *2025 2nd International Conference On Multidisciplinary Research and Innovations in Engineering (MRIE)*, 2025, pp. 364–369. doi:10.1109/MRIE66930.2025.11156663.

[22] M.-A. Kourtis et al., "Adaptive Post-Quantum Cryptography for Blockchain: Enhancing Security and Energy Efficiency with Cryptographic Agility," in *2025 6th International Conference in Electronic Engineering & Information Technology (EEITE)*, 2025, pp. 1–5. doi:10.1109/EEITE65381.2025.11166257.

[23] J. Zhang, Z. Liu, and D. Yao, "Fine-grained Privately Puncturable Signatures with Adaptive Security from Lattices," in *Proceedings of the 10th International Conference on Cyber Security and Information Engineering (ICCSIE '25)*, 2025, pp. 251–255. doi:10.1145/3759179.3760358.

[24] S. Narayanan, K. S. Archana, A. Rajesh, N. Parthiban, V. Srinivasan, and S. N. Sheela, "Quantum-Resilient IoT Communication Framework Using Post-Quantum Cryptography and Blockchain for Secure Edge Devices," *Iranian Journal of Science and Technology, Transactions of Electrical Engineering*, 2026. doi:10.1007/s40998-025-01002-1.

[25] I. Cherkaoui, C. Clarke, J. Horgan, and I. Dey, "Categorical framework for quantum-resistant zero-trust AI security," *Scientific Reports*, vol. 16, article 7030, 2026. doi:10.1038/s41598-026-37190-x.

[26] B. K. Kumar, A. D. P. Kumar, M. R. Sundari, D. Tejaswi, A. Dogga, and S. Bilgaiyan, "Automated cryptographic weakness discovery: A reinforcement learning approach for adaptive cryptanalysis," in *2025 International Conference on Advancements in Smart, Secure and Intelligent Computing (ASSIC)*, 2025, pp. 1–5. doi:10.1109/ASSIC64892.2025.11158372.

[27] J. Du, B. Ai, J. Liang, S. Liu, Y. Lu, and B. Shao, "An adaptive secure identity-based signcryption with equality test for high-speed railway train–ground communication," *IEEE Transactions on Network Science and Engineering*, vol. 13, pp. 1872–1886, 2026. doi:10.1109/TNSE.2025.3605157.

[28] Y. Huang, M. Ma, W. J. K. Raymond, and C.-O. Chow, "An adaptive intrusion detection system for the internet of things using large language models and post-quantum-secure blockchain," *Computer Networks*, vol. 274, p. 111819, 2026. doi:10.1016/j.comnet.2025.111819.

[29] Z. A. Haider, A. Zeb, A. K. M. M. Islam, T. Rahman, A. Arishi, and I. Ullah, "Enhancing IoT security with resource-efficient cryptography: A comprehensive review of lightweight and hybrid algorithms," *Computer Science Review*, vol. 59, p. 100861, 2026. doi:10.1016/j.cosrev.2025.100861.

[30] T.-C. Ho, Y.-M. Tseng, and S.-S. Huang, "LHSC-SGC: A lightweight hybrid signcryption scheme for smart grid communications in heterogeneous cryptographic public-key systems," *Computer Standards & Interfaces*, vol. 96, p. 104078, 2026. doi:10.1016/j.csi.2025.104078.

[31] D. Noetzold, V. R. Q. Leithardt, J. F. de Paz Santana, and J. L. V. Barbosa, "A Self-Adaptive Architecture for Predictive and Reinforcement-Based Optimization in Smart Environments," in *2025 International Symposium on Networks, Computers and Communications (ISNCC)*, 2025, pp. 1–6. doi:10.1109/ISNCC66965.2025.11250434.

[32] D. Noetzold, A. G. de Moraes Rossetto, L. A. Silva, P. Crocker, and V. R. Q. Leithardt, "JVM optimization: An empirical analysis of JVM configurations for enhanced web application performance," *SoftwareX*, vol. 28, p. 101933, 2024. doi:10.1016/j.softx.2024.101933.

[33] K. Zou, S. Wang, Y. Li, and H. Zhao, "Research on Building System of Adaptive Security Protection System of Cloud Platform Based on Localization Large Model," in *2024 International Conference on Electronics and Devices, Computational Science (ICEDCS)*, 2024, pp. 980–985. doi:10.1109/ICEDCS64328.2024.00181.

[34] I. Ari, K. Balkan, S. Pirbhulal, and H. Abie, "Ensuring Security Continuum from Edge to Cloud: Adaptive Security for IoT-based Critical Infrastructures using FL at the Edge," in *2024 IEEE International Conference on Big Data (BigData)*, 2024, pp. 4921–4929. doi:10.1109/BigData62323.2024.10825993.

[35] S. J. Mirsadri, R. Chaves, and L. Pedrosa, "Energy-Aware Adaptive Security for Smart Farming (EAASF): A Hybrid IDS-IPS Framework with SDN-Orchestrated for Agriculture 4.0," in *2025 23rd International Symposium on Network Computing and Applications (NCA)*, 2025, pp. 328–329. doi:10.1109/NCA67271.2025.00066.

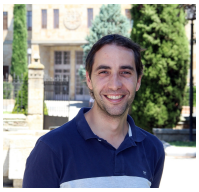
[36] H. Sohail, V. R. Q. Leithardt and A. Trigo, "PRISEC III: Cryptographic Techniques for Enhanced Security," 2025 25th International Conference on Control Systems and Computer Science (CSCS), Bucharest, Romania, 2025, pp. 354-361, doi: 10.1109/CSCS66924.2025.00059.

[37] Y. Zhang, X. Wang, and J. Liu, "Context-aware security management in smart environments: A machine learning approach," *Journal of Network and Computer Applications*, vol. 177, p. 102932, 2021, doi:10.1016/j.jnca.2020.102932.

[38] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018, xii+552 pp., ISBN 978-0-262-03924-6.

DARLAN NOETZOLD (Member, IEEE) is currently Ph.D. Student in Computer Engineering at the University of Salamanca, Spain. He received his Master's degree from the University of Vale do Rio dos Sinos (UNISINOS), São Leopoldo, Brazil. He also works as a Software Developer at CWI Software. His main research interests are High Performance Computing, Smart Environments, Cybersecurity, and Machine Learning.

JORGE L. V. BARBOSA received M.Sc. and Ph.D. in computer science from the Federal University of Rio Grande do Sul, Brazil. He conducted post-doctoral studies at Sungkyunkwan University (SKKU, Suwon, South Korea) and University of California Irvine (UCI, Irvine, USA). Jorge is a full professor at the Applied Computing Graduate Program (PPGCA) of the University of Vale do Rio dos Sinos (UNISINOS), head of the university's Mobile Computing Lab (MOBILAB), and a researcher at the Brazilian Council for Scientific and Technological Development (CNPq). His main research interests are Ubiquitous Computing, Ambient Intelligence, Big Data, Internet of Things (IoT), and Machine Learning.



JUAN F. P. SANTANA received a degree in Technical Engineering in Computer Systems in 2003, a degree in Computer Science Engineering in 2005, a degree in Statistics in 2007, and a Ph.D. in Computer Science in 2010, all from the University of Salamanca, Spain. He is currently a Full Professor at the University of Salamanca and a researcher at the Expert Systems and Applications Laboratory (ESALab). He is the coauthor of several papers published in journals, workshops, and symposiums.

VALDERI R. Q. LEITHARDT (Senior Member, IEEE) received the Ph.D. degree in computer science from INF-UFRGS, Brazil, in 2015. He is currently an Professor with the ISCTE - Instituto Universitario de Lisboa and a Researcher integrated with the Istar-Information Scienses, Technologies and Archicture Research Centre (ISTA), and Research Group Information Systems. He is also a collaborating Researcher with the Brazil and Spain. His research interests include distributed systems with a focus on data privacy, communication, and programming protocols, involving scenarios and applications for the intelligent systems, internet of things, smart cities, big data and cloud computing.