

Certificate Authentication and Revocation System for IoT Devices using CoAP and CBOR Encoding in LoRaWAN

¹Duke, Tamunotonye Juliana, ²Sako, D.J.S., ³Bennett, E.O.

Department of Computer Science,
Rivers State University, Port-Harcourt, Nigeria.

{¹tamunotonye.duke, ²dum.sako, ³bennett.okoni}@ust.edu.ng
<https://doi.org/10.56201/ijcsmt.vol.12.no9.2026.pg77.86>

Abstract

Certificate revocation protocols are mechanisms used in a public key infrastructure (PKI) to check if a digital certificate is still valid before its expiration date. Organizations deploy PKI across web servers, email platforms, virtual private networks, and, more recently, Internet of Things (IoT) devices to authenticate endpoints and encrypt data in transit. Certificate revocation in Long Range Wide Area Network (LoRaWAN) is highly inefficient because traditional PKI mechanisms, like Certificate Revocation Lists (CRLs) or the Online Certificate Status Protocol (OCSP), require data payloads and energy consumption that easily overwhelm resource-constrained IoT devices, introducing transaction sizes that are unsuitable for the 51-byte payload ceiling at LoRaWAN spreading factor 12 (SF12). This paper presents a system for a compact certificate revocation protocol for resource-constrained IoT devices operating on LoRaWAN networks. The proposed protocol combines Constrained Application Protocol (CoAP) and Concise Binary Object Representation (CBOR) encoding so that revocation request and response payloads remain within the LoRaWAN SF12 payload limit of 51 bytes. We implemented the proposed protocol in Python using standard cryptographic and constrained-network libraries, including OpenSSL, cryptography, aiocoap, and cbor2, together with a LoRaWAN simulation and energy-modelling environment. The protocol produced a 10-byte request and 19-byte response, giving a total payload of 29 bytes and remaining within the SF12 payload limit. Cache hit rates exceeded 97% across all tested scenarios under the shared peer-certificate workload. A web application interface was developed to expose the certificate-revocation workflow through configurable scenario inputs, communication testing, event logging, and report export controls.

Keywords: LoRaWAN, Internet of Things, Concise binary Object Representation, Constrained Application Protocol, Public Key Infrastructure

1. Introduction

Since the introduction of X.509 certificate standards in 1988, Public Key Infrastructure (PKI) has underpinned secure digital communications by binding public keys to verified identities through digitally signed certificates (Khan et al., 2023). Certificate Authorities (CAs) issue these certificates and enforce validity through expiration dates and revocation procedures that withdraw trust from a certificate before it expires, for instance when a private key is stolen or an entity's affiliation changes. The growth of Internet of Things (IoT) deployments has increased the need for

scalable certificate management. Many IoT systems operate in industrial monitoring, smart agriculture, environmental sensing, healthcare, and asset-tracking environments where devices may remain deployed for several years with limited physical access. These devices frequently operate with constrained processors, limited memory, low data rates, and battery supplies that must be preserved for long-term operation (Bormann & Hoffman, 2020; Sinha, 2024). Consequently, security mechanisms that are acceptable in conventional IP networks may become impractical in constrained IoT environments.

Deploying certificate-based authentication on Long Range Wide Area Networks (LoRaWANs) exposes a fundamental incompatibility: once a certificate is compromised, there is no bandwidth-feasible and energy-efficient way to revoke it. Two specific problems arise:

Firstly, existing certificate revocation mechanisms are not payload-feasible for LoRaWAN. CRL downloads range from 10 KB to 10 MB, and a single traditional OCSP exchange consumes roughly 3.5-5 KB (Khan et al., 2023). Even TinyOCSP, with an approximately 40-byte request, exceeds the 51-byte SF12 payload ceiling once CoAP, DTLS, and LoRaWAN headers are included (Hoglund et al., 2023). Therefore, LoRaWAN deployments lack a revocation protocol whose request and response payloads can fit within the most restrictive packet limit.

Secondly, existing approaches do not adequately reduce repeated communication and energy overhead for battery-powered LoRaWAN IoT devices. Standard OCSP requires a fresh round-trip for each certificate-status check, CRL requires large periodic downloads, and TinyOCSP does not provide LoRaWAN-specific gateway caching, batch checking, or spreading-factor-aware packet enforcement. Therefore, repeated revocation checking remains inefficient for resource-constrained devices that must preserve battery life.

Closing this gap requires a purpose-built LoRaWAN revocation protocol with compact request-response payloads and resource-optimisation mechanisms for repeated certificate-status checking. In this work, we develop a system for a compact certificate revocation protocol for resource-constrained IoT devices operating on LoRaWAN networks. The proposed protocol combines Constrained Application Protocol (CoAP) and Concise Binary Object Representation (CBOR) encoding so that revocation request and response payloads remain within the LoRaWAN SF12 payload limit of 51 bytes. We implement the proposed protocol in Python using standard cryptographic and constrained-network libraries, including OpenSSL, cryptography, aiocoap, and cbor2, together with a LoRaWAN simulation and energy-modelling environment. aiocoap is a native Python implementation of CoAP (RFC 7252) and cbor2 is an implementation of CBOR (Concise Binary Object Representation, RFC 8949). A web application interface is developed to expose the certificate-revocation workflow through configurable scenario inputs, communication testing, event logging, and report export controls.

2. Theoretical Background and Related Work

The X.509 certificate standard, originally defined by the International Telecommunication Union and adopted as IETF RFC 5280, defines a structured format containing the subject's distinguished name, public key, validity period, issuer information, extensions for usage constraints, and a cryptographic signature binding these elements together under the issuing CA's private key (Cooper et al., 2008). Version 3 certificates introduced critical extensions including key usage constraints, subject alternative names, authority key identifiers, and Certificate Revocation List (CRL) distribution points that enable fine-grained control over certificate application and validation requirements. Certificate validation requires verifying the complete chain from the end-

entity certificate through intermediate CAs to a trusted root, checking each certificate's signature, validity period, key usage constraints, and revocation status (Khan et al., 2023). This multi-step validation process imposes substantial computational and bandwidth overhead particularly problematic for resource-constrained IoT devices operating on networks with severe packet size limitations (Ferreira et al., 2023).

Certificate signature verification involves asymmetric cryptographic operations where the validator computes a hash of the certificate's to-be-signed portion and verifies this matches the decrypted signature using the issuer's public key, mathematically expressed in Equation 1:

$$\text{Verify}(H(\text{TBS}_{\text{cert}}), \sigma, PK_{\text{issuer}}) = \begin{cases} \text{valid} & \text{if } H(\text{TBS}_{\text{cert}}) = \text{Decrypt}(\sigma, PK_{\text{issuer}}) \\ \text{invalid} & \text{otherwise} \end{cases} \quad (1)$$

where H represents the cryptographic hash function (SHA-256), TBS_{cert} is the to-be-signed certificate content, σ is the digital signature, and PK_{issuer} is the issuer's public key.

This verification process consumes significant CPU cycles on constrained devices, with Elliptic Curve Digital Signature Algorithm (ECDSA)-P256 signature verification requiring approximately 120,000 cycles on Advanced RISC Machine (ARM) Cortex-M4 microcontrollers compared to 600,000 cycles for Rivest-Shamir-Adleman (RSA)-2048 verification (Kannwischer et al., 2019). Figure 1 depicts the hierarchical certificate validation process where end-entity certificates are verified through intermediate Certificate Authorities back to a trusted root CA. The validation process (right side) shows three critical steps: signature verification using the issuer's public key (Equation 1), temporal validity checking against NotBefore and NotAfter timestamps, and revocation status confirmation via Online Certificate Status Protocol (OCSP) or CRL queries. Each certificate in the chain must successfully pass all validation checks for the end-entity certificate to be accepted.

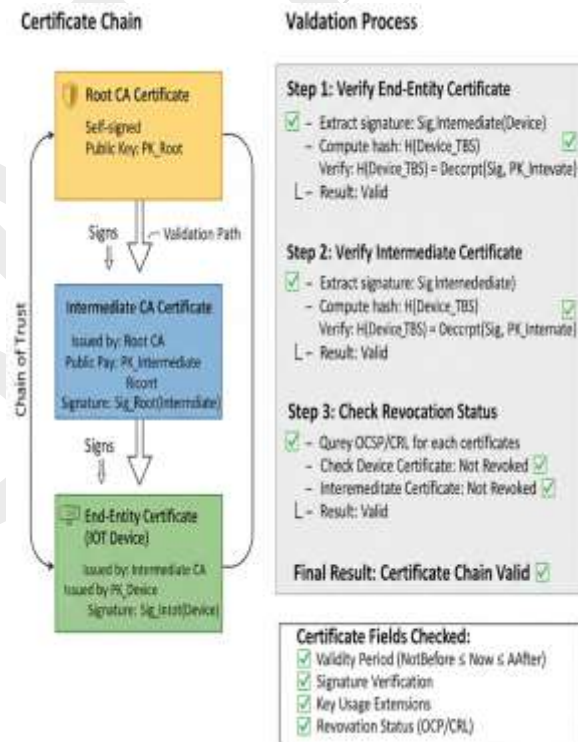


Figure 1: X.509 certificate validation chain.
(Khan et al., 2023)

Hoglund et al. (2023) proposed TinyOCSP, a lightweight certificate revocation protocol for low-power IoT devices using CoAP and CBOR. TinyOCSP is the closest baseline to this study because it targets constrained revocation rather than conventional web PKI. It significantly reduces transaction size and energy use compared with traditional OCSP. Nevertheless, TinyOCSP assumes a constrained IP environment with larger packet budgets than the 51-byte SF12 ceiling found in LoRaWAN. When a TinyOCSP request is combined with CoAP, DTLS, and LoRaWAN frame overheads, the full exchange no longer satisfies the strictest LoRaWAN payload condition evaluated in this study. Research on LoRaWAN security has shown that protocol design must

account for strict bandwidth and energy limits. Sanchez-Iborra et al. (2018) proposed a lightweight authenticated key-management approach for LoRaWAN, showing that security enhancements must remain compatible with low-power operation. Haxhibeqiri et al. (2018) surveyed LoRaWAN architecture and highlighted the relationship between spreading factor, range, payload capacity, and energy cost. Stanco et al. 2024 further examined LPWAN security weaknesses and identified the need for lightweight mechanisms tailored to low-power wide-area constraints.

Other studies confirmed that public-key cryptography is increasingly feasible on constrained devices, but lifecycle management remains incomplete. Milani and Chatzigiannakis (2021) evaluated elliptic-curve cryptography for LoRaWAN rejoin mechanisms, while Astorga et al. (2022) discussed public-key cryptography feasibility in Industrial IoT. Goppert et al. (2024) surveyed certificate lifecycle management in industrial networking and reported that certificate revocation remains absent in many industrial standards. These findings support the need for a LoRaWAN-specific revocation mechanism that remains compatible with X.509 PKI while respecting severe packet and energy constraints.

3 Methodology and System Design

This section presents the behavioural and structural model of the proposed certificate revocation protocol. The protocol combines Constrained Application Protocol (CoAP) (Shelby et al. 2014) and Concise Binary Object Representation (CBOR) (Bormann & Hoffman, 2020) encoding so that revocation request and response payloads remain within the LoRaWAN SF12 payload limit of 51 bytes.

CoAP, designed as a lightweight alternative to HTTP, structured for constrained devices (like microcontrollers) and lossy networks, uses similar request/response model (GET, POST, PUT, DELETE). CBOR, is compact binary data serialization format loosely based on the JSON data mode but uses a binary instead of text-based format. It reduces payload size significantly compared to text-based formats like JSON by avoiding redundant metadata and repeating key names.

Certificate Authority generates and signs X.509 certificates, tracks revocations, and resolves status queries. CBOR Encoder handles all serialisation (single requests, responses, and batch arrays) plus the inverse decode operations. CoAP Transport wraps the aiocoap client and server, adding an MTU-enforcement gate that rejects any payload too large for the current spreading factor. Revocation Cache manages the TTL-bounded entry store and exposes a hit-rate metric for the evaluator. Figure 2 shows the Proposed System Architecture for Certificate Revocation Protocol for LoRaWAN IoT-based Network.

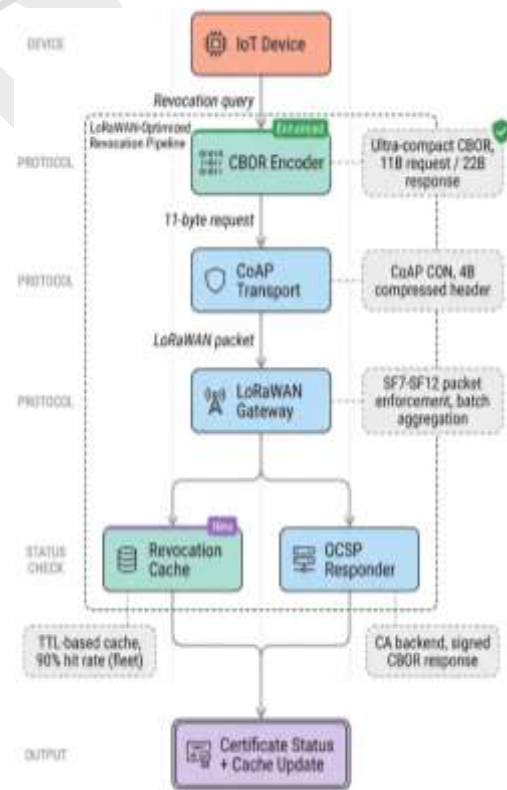


Figure 2: System Architecture

The proposed system extends the TinyOCSP model with LoRaWAN-specific optimisations. It consists of an IoT device layer, LoRaWAN radio link, LoRaWAN gateway with revocation cache, network server or OCSP responder, CBOR encoding engine, and batch aggregation module. The device constructs a compact CBOR query using a compressed certificate identifier and nonce, and the request is transported as a CoAP confirmable message. CBOR was selected because it provides compact binary serialization for constrained nodes (Bormann & Hoffman, 2020), while CoAP provides a lightweight request-response transport suitable for constrained Internet devices (Shelby et al. 2020). The gateway checks whether the certificate status is already cached. On a cache hit, the gateway returns the cached result without contacting the responder. On a cache miss, the query is forwarded to the responder, which checks the revocation database and returns a compact status response containing a status code, validity window, and truncated response-authentication tag.

The class diagram captures the static structure of the implementation; the sequence diagram traces the dynamic interaction between components during a revocation check; and the activity and state diagrams represent the control flow and lifecycle states of the protocol in operation. Together, these models provide a complete specification of system behaviour from both structural and dynamic perspectives. Six classes make up the implementation, linked by composition. The central orchestrator, LoRaWAN Revocation Protocol, owns one instance of each specialist class and delegates work through four public methods: `check_revocation()` for single-certificate queries, `batch_check()` for multi-serial gateway aggregation, `run_evaluation()` for benchmark runs, and `generate_report()` for metric output. Figure 3 shows the full class structure

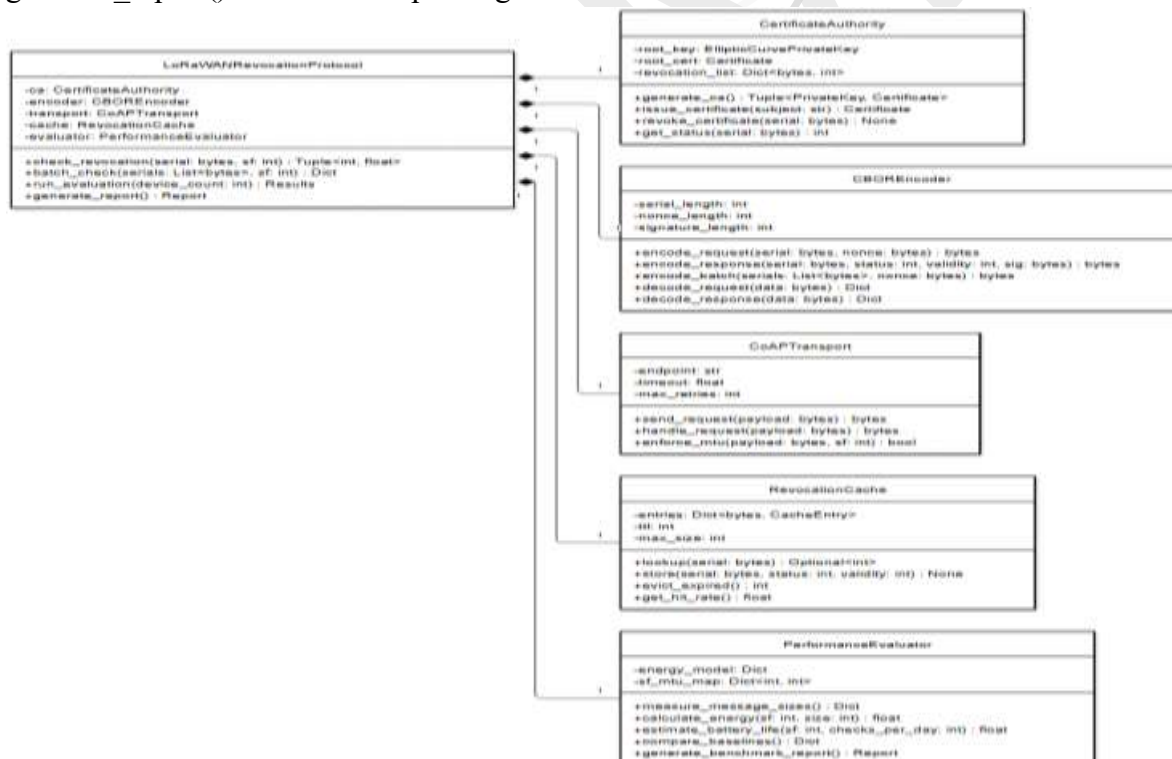


Figure 3: System Class Diagram Showing Modular Lightweight Certificate Revocation Protocol Architecture

Figure 4 traces a single revocation check through the system, showing both the fast path (cache hit) and the full path (cache miss with server round-trip). On a cache hit the gateway answers from

local storage, with no server contact and no responder round-trip. On a miss the gateway relays the query, receives a 19-byte CBOR response, writes it into the cache for future use, and forwards the answer back to the device. The implemented single-query payloads are 10 bytes for the request and 19 bytes for the response; with the compact 4-byte CoAP request header included for MTU enforcement, the request frame remains 14 bytes and the response remains 19 bytes, both below the SF12 limit of 51 bytes. Figure 5 traces the end-to-end workflow of a single-certificate revocation check, from the moment an IoT device triggers a query to the point where a status result is returned. The diagram captures the two principal execution paths, namely the fast path through the gateway cache and the full path through the OSCP responder, along with the validation gates that guard against oversized packets and invalid response-authentication tags.

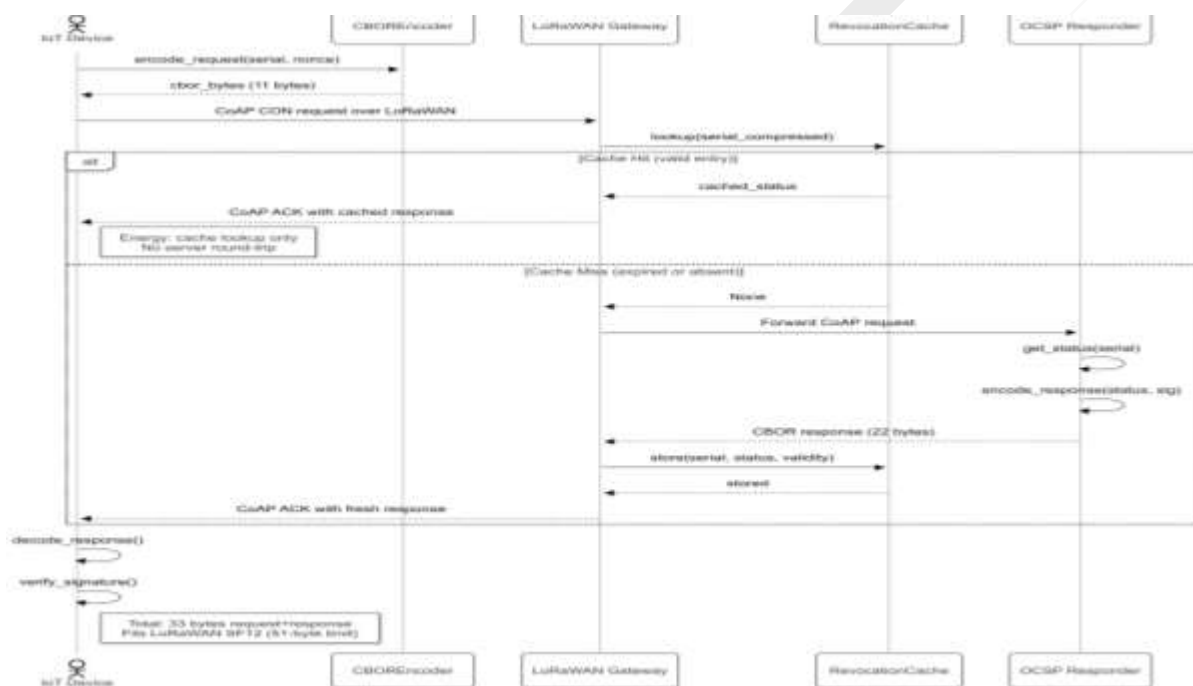


Figure 4: Revocation checking sequence diagram.

The workflow begins with serial-number compression, hashing the full 20-byte X.509 serial down to 4 bytes, followed immediately by a cache lookup at the gateway. On a hit the protocol short-circuits: the cached status is returned with negligible cache-lookup energy and no responder round-trip. On a miss the request enters the encoding stage, where the compressed serial, a 2-byte nonce, and a type code are packed into a 10-byte compact CBOR array. Before transmission, an MTU validation gate checks that the encoded request payload plus the 4-byte CoAP header fits within the active spreading factor's ceiling; a violation halts the process rather than fragmenting the packet. If the size check passes, the CoAP confirmable message is transmitted, a 19-byte response is received, and the payload is decoded. A second gate verifies the compact response-authentication tag, rejecting responses whose status fields are not bound to the original request, and only then does the protocol update the gateway cache and return the status to the requesting device.

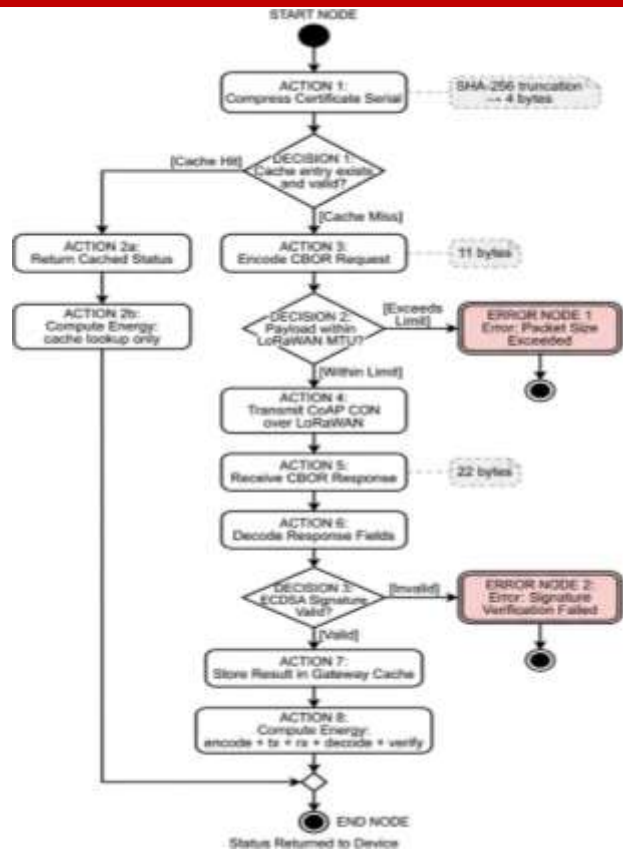


Figure 5: Single-certificate revocation activity diagram.

Table 2 shows representative output from a 100-device scenario. The protocol produced a 10-byte request and 19-byte response, giving a total payload of 29 bytes and remaining within the SF12 payload limit. A web application interface was developed to expose the certificate-revocation workflow through configurable scenario inputs, communication testing, event logging, and report export controls. Figure 6 shows the scenario-configuration view used to define certificate, device, payload, cache, and revocation-rule inputs before running a simulation.

Table 1: Gateway cache hit rate

Device Population	Cache Hit Rate
100	0.9792
200	0.9896
300	0.9931
400	0.9948
500	0.9958

4. Implementation and Results

We implemented the proposed protocol in Python using standard cryptographic and constrained-network libraries, including OpenSSL, cryptography, aiocoap, and cbor2, together with a LoRaWAN simulation and energy-modelling environment. aiocoap is a native Python implementation of CoAP (RFC 7252) and leverages Python's built-in asyncio framework to handle concurrent requests asynchronously without requiring multi-threading. cbor2 is an implementation of CBOR (Concise Binary Object Representation, RFC 8949).

A pool of 500 X.509 v3 certificates was generated using Python's cryptography package. The simulated deployments covered 100, 200, 300, 400, and 500 devices. Each deployment was evaluated across LoRaWAN spreading factors SF7 to SF12. Cache hit rates exceeded 97% across all tested scenarios (device population) under the shared peer-certificate workload as presented in Table 1.

The web application also records individual communication outcomes. Figure 7 shows the event log after three communication attempts, including two allowed transactions and one blocked transaction caused by expired certificate. This interface output is consistent with the revocation and feasibility checks reported in the simulation tables. Figure 8 shows the reporting view, where evidence files can be exported and summary counts are displayed for certificate status, communication result, handshake completion, constraints, and cache outcomes.

Table 2: Representative output from a 100-device scenario

Device Population	Spreading Factor	Request (bytes)	Response (bytes)	Total (bytes)	Energy per Check (mJ)	Cache Hit Rate	Battery Lifetime (days)	Batch Capacity
100	SF7	10	19	29	23.3906	0.9792	11,957.87	42
100	SF8	10	19	29	44.6796	0.9792	9,511.95	42
100	SF9	10	19	29	76.7820	0.9792	7,269.68	21
100	SF10	10	19	29	152.8140	0.9792	4,665.10	8
100	SF11	10	19	29	304.8780	0.9792	2,717.70	8
100	SF12	10	19	29	609.0060	0.9792	1,481.14	8

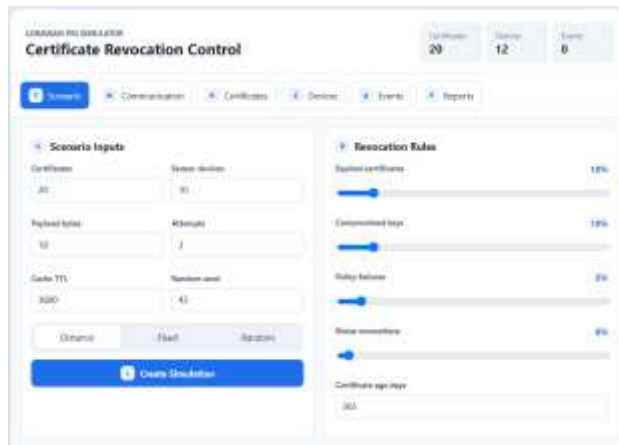


Figure 6: Web application scenario-configuration screen.

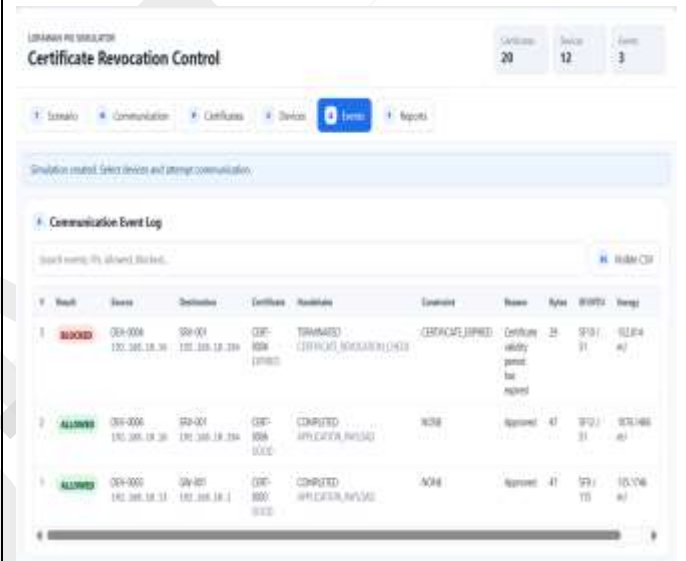


Figure 7: Web application communication event log.

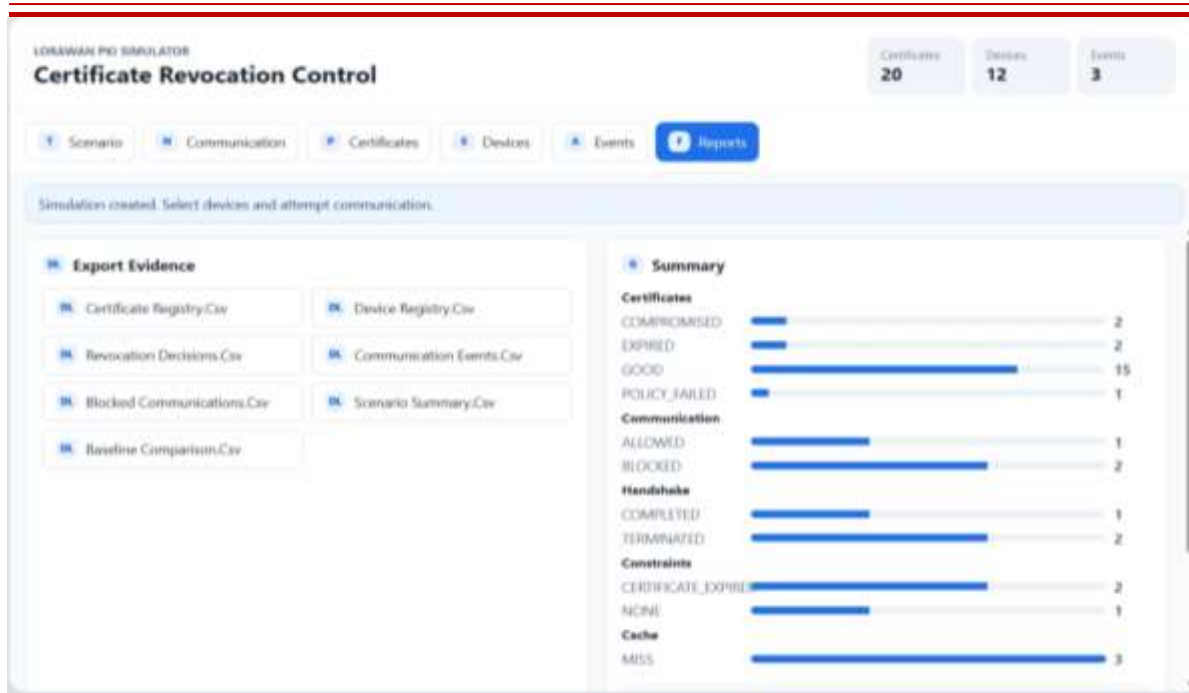


Figure 8: Web application evidence-export and summary screen.

5. Conclusion

This paper presented a system for a compact certificate authentication and revocation protocol for resource-constrained IoT devices operating on LoRaWAN networks. The protocol was motivated by the mismatch between conventional PKI revocation mechanisms and LoRaWAN's severe payload and energy constraints. The design used principally a compact CBOR encoding, CoAP transport assumptions so that revocation request and response payloads remain within the LoRaWAN SF12 payload limit of 51 byte. The protocol was implemented in Python using standard cryptographic and constrained-network libraries, including OpenSSL, cryptography, aiocoap, and cbor2, together with a LoRaWAN simulation and energy-modelling environment. We evaluated the system through a single-host LoRaWAN simulation using 500 synthetic X.509 v3 certificates, device populations from 100 to 500, and spreading factors SF7 to SF12. The implemented protocol produced a 10-byte request and 19-byte response, giving a total transaction payload of 29 bytes. This remained within the 51-byte SF12 payload ceiling. A web application interface was developed to expose the certificate-revocation workflow through configurable scenario inputs, communication testing, event logging, and report export controls.

Future work should validate the protocol on physical LoRaWAN hardware, perform formal security analysis of the compact response-authentication mechanism, and evaluate the design under broader trust topologies, adversarial conditions, and other low-power wide-area networks such as NB-IoT and Sigfox.

References

- Astorga, J., Barcelo, M., Urbieto, A & Jacob, E. (2022). Revisiting the feasibility of public key cryptography in light of IIoT communications. *Sensors*, 22(7), 2561.
- Bormann, C., & Hoffman, P. (2020). Concise binary object representation (CBOR) (RFC 8949). Internet Engineering Task Force. <https://doi.org/10.17487/RFC8949>
- Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., & Polk, W. (2008). Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile (RFC 5280). Internet Engineering Task Force. <https://doi.org/10.17487/RFC5280>
- Ferreira, C. M., Sousa, N. M., & Correia, M. E. (2023). Certificate-based authentication for LoRaWAN: Design and performance evaluation. *Ad Hoc Networks*, 141, 103089. <https://doi.org/10.1016/j.adhoc.2023.103089>
- Goppert, J., Walz, A. & Sikora, A. (2024). A survey on life-cycle-oriented certificate management in industrial networking environments, *Journal of Sensor and Actuator Networks*, 13(2), 26.
- Hoglund, J., Furuheid, M., & Raza, S. (2023). Lightweight certificate revocation for low-power IoT with end-to-end security. *Journal of Information Security and Applications*, 73, 103424. <https://doi.org/10.1016/j.jisa.2023.103424>
- Haxhibeqiri, J., De Poorter, E., Moerman, I. & Hoebeke, J. (2018). A survey of LoRaWAN for IoT: From technology to application, *Sensors*, 18(11), 3995.
- Kannwischer, M. J., Rijneveld, J., Schwabe, P., & Stoffelen, K. (2019). pqm4: Testing and Benchmarking NIST PQC on ARM Cortex-M4. *Cryptology ePrint Archive*, Paper 2019/844. <https://eprint.iacr.org/2019/844>
- Khan, S., Lu, Y., Jiang, T., Madni, H. A., Khan, K. M., Rathnayake, M. V., Alzaidi, M. S., Aljuaid, H., Aziz, A., Yar, H., Ali, K., & Akbar, A. (2023). A survey on X.509 public-key infrastructure, certificate revocation, and their modern implementation on blockchain and ledger technologies. *IEEE Communications Surveys & Tutorials*, 25(4), 2306-2343. <https://doi.org/10.1109/COMST.2023.3323640>
- Milani S. & Chatzigiannakis, I. (2021). Design, analysis, and experimental evaluation of a new secure rejoin mechanism for LoRaWAN using elliptic-curve cryptography, *Journal of Sensor and Actuator Networks*, 10(2), 36.
- Sanchez-Iborra, R., Sanchez-Gomez, J., Perez, S., Fernandez, P. J., Santa, J., Hernandez-Ramos, J. L., & Skarmeta, A. F. (2018). Enhancing LoRaWAN security through a lightweight and authenticated key management approach. *Sensors*, 18(6), 1833. <https://doi.org/10.3390/s18061833>
- Sanchez-Iborra, R., Sanchez-Gomez, J., Ballesta-Vinas, J., Cano, M. D., & Skarmeta, A. F. (2023). Performance evaluation of LoRa technology with different spreading factors. *Computer Communications*, 192, 236-246. <https://doi.org/10.1016/j.comcom.2023.015432>
- Shelby, Z., Hartke, K., & Bormann, C. (2014). The Constrained Application Protocol (CoAP) (RFC 7252). Internet Engineering Task Force. <https://doi.org/10.17487/RFC7252>
- Sinha, S. (2024). State of IoT 2024: Number of connected IoT devices growing 13% to 18.8 billion globally, *IoT Analytics*.
- Stanco, G., Navarro, A., Frattini, F., Ventre, G. & Botta, A. (2024). A comprehensive survey on the security of low power wide area networks for the Internet of Things, *ICT Express*, 10 (3), 519-552.