Performance Analysis of ASCON-AEAD for Securing Gas Lift SCADA Telemetry in the Oil and Gas Sector

Lidya Marthadilla - 18223134
Information System and Technology
School of Electrical Engineering and Informatics
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: lidyamarthadila@gmail.com, 18223134@std.stei.itb.ac.id

***Abstract*—Gas-lift operations in the oil and gas sector rely on SCADA telemetry to monitor critical well conditions such as annulus pressure, tubing pressure, choke size, compressor status, and production flow. If these telemetry values are intercepted or modified, attackers may hide abnormal operating conditions and influence safety decisions. This paper evaluates the performance of ASCON-AEAD for securing gas-lift SCADA-like telemetry generated from the Volve Production Data. The historical Volve dataset was used as a real production baseline and converted into 2,000 telemetry records. The experiment separates telemetry fields into associated data and encrypted payload, then benchmarks three ASCON-AEAD variants supported by the Python ascon library, Ascon-128, Ascon-128a, and Ascon-80pq. The results show that Ascon-128a achieved the best performance, with an average encryption time of 3267.228 us and an average decryption time of 3275.091 us per record. All variants produced a constant 16-byte authentication tag overhead. In the false data injection simulation, all 1,914 forged abnormal records were rejected by every variant. These results indicate that ASCON-AEAD can provide confidentiality and integrity protection for gas-lift telemetry with low payload overhead.**

Keywords—ASCON-AEAD, lightweight cryptography, SCADA, gas lift, oil and gas, false data injection, telemetry security

I. Introduction

The oil and gas industry uses a lot of automation and digital systems to keep track of production and make sure everything runs safely. At the center of this setup is the SCADA (Supervisory Control and Data Acquisition) system, which works as a remote monitoring network to collect data from offshore platforms and send it back to the control room on shore [1], [2]. In daily operations, these networks connect field devices like PLCs and RTUs at the wellsite directly to the engineers [2].

In offshore production, one of the most common methods to maintain oil flow from old reservoirs is gas lift operations [1]. This method works by injecting high-pressure gas into the wellbore annulus to make the oil lighter so it can flow easily to the surface [1]. To monitor this process, the SCADA system continuously collects telemetry data from sensors, such as average annulus pressure, downhole temperature, tubing pressure, and choke sizes [2]. Getting accurate and real-time data from these sensors is very important, because if the gas injection rate is wrong, it can damage the compressor or cause serious issues in the well [1].

The main problem is that traditional SCADA networks are not very secure. Old industrial protocols like Modbus RTU or basic Ethernet were designed long ago for isolated networks, so they do not have built-in security or encryption features [2]. Because of this, critical gas-lift data is usually sent in unencrypted plaintext over the network [2]. This opening allows hackers to intercept the telemetry data and launch False Data Injection Attacks (FDIAs).

We can see the real danger of sensor data manipulation from famous cyberattacks in the past. For example, the Stuxnet malware showed how attackers can change sensor data to hide physical damage while showing normal numbers on the operator's monitor [7]. Another example is the Colonial Pipeline cyberattack, which proved that network vulnerabilities can completely shut down a critical energy system [8]. In a gas-lift system, a hacker could change the casing pressure or gas volume numbers. This fake data would trick the automated system or the operator into opening or closing valves incorrectly, which can cause severe damage to the offshore platform hardware.

However, encrypting this data is difficult because wellhead sensors and RTUs are resource-constrained devices [1]. As seen in actual offshore implementations, these remote devices often run on limited battery modules and have very low-power processors and small memories [1]. If we try to use heavy standard encryption like AES-256, the device's hardware cannot handle the processing workload. It will cause high computational overhead and network latency, which disrupts the real-time data requirements needed for oilfield safety.

To fix this security and resource limitation issue, lightweight cryptography (LWC) was created. In 2023, NIST officially chose the Ascon cipher suite as the new standard for lightweight cryptography on small devices [4], [5]. Ascon-AEAD is an ideal choice because it provides Authenticated Encryption with Associated Data, meaning it encrypts the sensor data to keep it confidential and

automatically generates authentication tags to protect data integrity [6].

Therefore, this paper analyzes the performance of the Ascon-AEAD protocol to secure gas-lift SCADA telemetry data. To make the evaluation realistic, we use actual production telemetry data from the official Volve Field Data Sharing project provided by Equinor [3]. By simulating a resource-constrained environment, this study benchmarks the execution time, throughput, and anti-tampering validation capabilities of Ascon-AEAD. The final goal of this research is to prove that the new NIST standard can successfully protect gas-lift networks without causing network delays.

II. Theoretical Background

A. SCADA Telemetry Frame and Policy Separation

A Supervisory Control and Data Acquisition (SCADA) system in gas-lift operations works by continuously polling data registers from edge Remote Terminal Units (RTUs) [2]. To secure this industrial data stream using Authenticated Encryption with Associated Data (AEAD), the SCADA telemetry frame must be divided into two distinct technical protection policies based on its cryptographic requirements:

1) Associated Data (Authenticated Only)

   This policy covers structural network metadata components that must remain in plaintext so the SCADA infrastructure can parse and route the packet. However, this data must be cryptographically bound to the payload to detect unauthorized context modification or packet tampering. Replay protection would require additional nonce or timestamp tracking at the receiver side. In gas-lift telemetry frameworks, this metadata includes tracking identifiers, temporal markers, and source components such as `telemetry_id`, `timestamp`, `source_date`, `well_bore_code`, `npd_well_bore_name`, and `facility_name` [3].

2) Encrypted Payload (Confidentiality and Integrity)

   This policy covers the sensitive industrial process variables that must be completely hidden from unauthorized network eavesdroppers. This operational dataset includes continuous physical sensor measurements, production targets, and engineering indicators such as pressures (annulus pressure, tubing differential pressure), temperatures, surface choke valve sizes, volumetric production rates (oil, gas, water), gas-oil ratio (GOR), estimated gas-lift injection rates, valve states, compressor health metrics, and active alarm targets [1], [2].

B. ASCON-AEAD Variants and Cryptographic Trade-offs

The Ascon cipher suite provides different design profiles tailored to varying hardware capabilities and security margins [5]. Evaluating these different architectural profiles is necessary to find the optimal operational balance for low-power edge nodes running industrial fieldbuses [4]. This study focuses on benchmarking three ASCON-AEAD variants supported by the Python ascon library, Ascon-128, Ascon-128a, and Ascon-80pq. The Ascon family itself has been standardized by NIST for lightweight cryptography.

1) Ascon-128

   The baseline profile utilizing a 128-bit secret key, a 128-bit Nonce, and a 128-bit authentication tag [6]. It operates with a 64-bit block rate ($r = 64$) and runs 12 initialization permutation rounds ($a = 12$) and 6 internal data processing rounds ($b = 6$)[5].

2) Ascon-128a

   A high-throughput profile designed for faster operational processing [5]. It maintains the same key and state sizes as Ascon-128 but doubles the block rate to 128 bits ($r = 128$) and utilizes 8 internal permutation rounds ($b = 8$) [6], allowing it to absorb and process larger payload bytes per execution cycle [5].

3) Ascon-80pq

   A post-quantum extended profile designed to resist future quantum computing attack vectors [5]. It increases the secret key length to 160 bits (20 bytes) to achieve a higher security margin while maintaining a standard 64-bit block rate ($r = 64$) and standard permutation round counts ($a = 12, b = 6$) [5], [6].

C. Performance and Integrity Benchmarking Metrics

To perform a rigorous empirical evaluation of lightweight ciphers on resource-constrained automation systems, specific operational and cryptographic behavioral metrics must be explicitly measured:

1) Computational Latency and Throughput: Measured via average encryption time (`avg_encryption_us`) and decryption time (`avg_decryption_us`) in microseconds, alongside their respective processing speeds (`encryption_throughput_kib_s` and `decryption_throughput_kib_s`). These metrics determine whether the lightweight cipher can execute within the strict real-time polling windows of industrial SCADA servers without introducing communication timeouts [2].
2) Size Overhead: Because AEAD ciphers append a security verification tag to the payload, size metrics must track plaintext bytes, ciphertext bytes, and the resulting size overhead (`avg_overhead_bytes`) [4]. For all Ascon variants, this transmission size overhead is mathematically fixed at exactly 16 bytes due to the generation of the 128-bit authentication tag [6].
3) Forgery and Tampering Detection: To evaluate the network's resilience against malicious False Data Injection Attacks (FDIAs), the simulation tracks the total forgery attempts and the total forgery rejections [7]. The forgery rejection rate is calculated using the following mathematical relation:

$$Forgery\ Rejection\ Rate\ =$$

$$\left(\frac{Forgery\ Rejected}{Forgery\ Attempts}\right) \times\ 100\%$$

A secure AEAD implementation must achieve a 100% forgery rejection rate, proving that even if an attacker alters anomalous telemetry values to mimic normal operational trends, the validation check will instantly catch the mismatch and drop the packet [6], [7].

III. Methodology

A. Experimental Workflow

The overall research workflow is illustrated in Figure 1. The experiment starts from the Volve Production Data workbook, which is used as the historical baseline. The gas_lift.py pipeline then converts the selected production rows into 2,000 gas-lift SCADA-like telemetry records. After the telemetry records are generated, the fields are separated into associated data and encrypted payload based on their security requirements. The protected telemetry payload is then processed by the benchmark.py module using three ASCON-AEAD variants supported by the Python ascon library: Ascon-128, Ascon-128a, and Ascon-80pq. Finally, a false data injection scenario is simulated, and the resulting performance and integrity metrics are extracted for analysis.

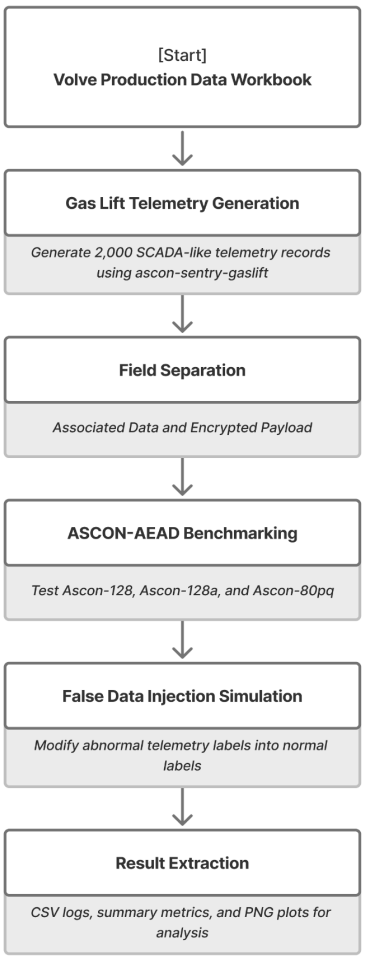

Fig 1. Experimental workflow for gas-lift telemetry generation and ASCON-AEAD benchmarking.

The workflow consists of four main functional stages. First, data ingestion and parsing are performed to extract daily production baselines from the Volve Production Data workbook. Second, telemetry generation is performed by expanding the historical baseline into SCADA-like gas-lift telemetry records using deterministic jitter and event labeling. Third, the cryptographic benchmarking stage measures encryption latency, decryption latency, throughput, and payload size overhead. Fourth, the security validation stage evaluates whether ASCON-AEAD can reject modified telemetry packets in a false data injection scenario.

B. Data Source and Telemetry Generation

This research uses the Volve Production Data as the historical data source. The dataset contains real oil and gas production records, including daily production values and well operation information. However, the dataset is not a raw real-time SCADA packet dataset. Therefore, this research uses the Volve daily production data as a realistic historical baseline to generate gas-lift SCADA-like telemetry records.

The telemetry dataset was generated from the daily production sheet in the Volve workbook. Each historical production row was transformed into several telemetry samples to simulate continuous SCADA monitoring. The generated telemetry includes operational values such as on-stream hours, downhole pressure, downhole temperature, tubing differential pressure, annulus pressure, choke size, wellhead pressure, wellhead temperature, oil rate, produced gas rate, water rate, gas-oil ratio, estimated gas-lift rate, lift valve state, compressor discharge pressure, and compressor power.

In this experiment, a controlled subset of 2,000 telemetry records was used. This size was selected as a balance between scenario diversity and practical benchmark runtime, since the ASCON implementation used in this experiment is a Python implementation. The subset is sufficient to compare encryption time, decryption time, throughput, size overhead, and forgery detection across the selected ASCON-AEAD variants.

C. Data Protection Model

The generated telemetry fields were divided into two groups: associated data and encrypted payload. Associated data is not encrypted, but it is authenticated by ASCON-AEAD. This means the receiver can detect unauthorized changes to those fields.

The associated data fields are `telemetry_id`, `timestamp`, `source_date`, `well_bore_code`, `npd_well_bore_name`, and `facility_name`. These fields represent context information that may be needed for routing, logging, or indexing. Although they are kept visible, they are cryptographically bound to the authentication tag. Therefore, any unauthorized modification to these fields will cause authentication failure.

The encrypted payload contains sensitive operational values and safety-related labels. These include `on_stream_hours`, `downhole_pressure_bar`, `downhole_temperature_c`, `dp_tubing_bar`,

`annulus_pressure_bar`, `choke_size_pct`, `wellhead_pressure_bar`, `wellhead_temperature_c`, `oil_rate_sm3_d`, `produced_gas_rate_sm3_d`, `water_rate_sm3_d`, `gas_oil_ratio`, `estimated_gas_lift_rate_sm3_d`, `lift_valve_state`, `compressor_discharge_pressure_bar`, `compressor_power_kw`, `alarm_triggered`, `event_type`, and `target`. These fields are encrypted to preserve confidentiality and authenticated to detect unauthorized modification.

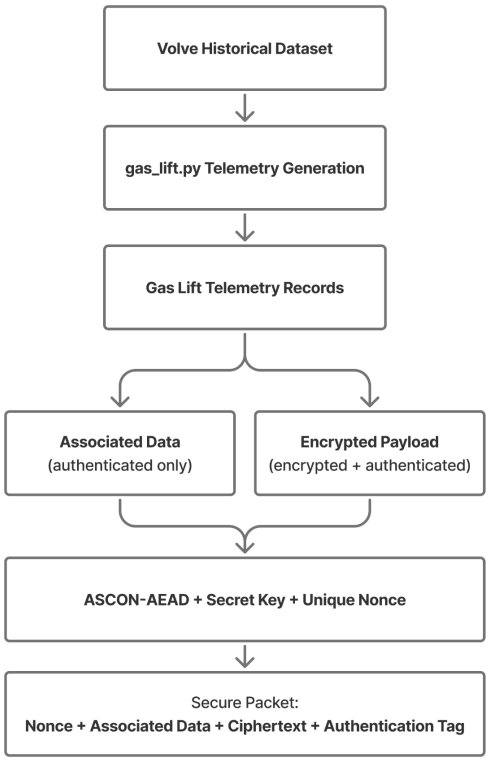

Fig 2. ASCON-Sentry gas-lift telemetry protection architecture.

D. ASCON-AEAD Variants

This research evaluates three ASCON-AEAD variants supported by the Python ascon library: Ascon-128, Ascon-128a, and Ascon-80pq. The Ascon family itself has been standardized by NIST for lightweight cryptography through SP 800-232. Each variant was tested using the same generated gas-lift telemetry dataset.

For every telemetry record, the protected payload was encrypted, then decrypted and verified using the authentication tag. A unique nonce was generated for each record by combining a random 64-bit prefix with a 64-bit counter. This design prevents nonce reuse under the same secret key during a single benchmark run.

E. False Data Injection Attack Scenario

A False Data Injection Attack (FDIA) scenario was simulated to evaluate the integrity protection of ASCON-AEAD. In this scenario, an attacker attempts to modify abnormal telemetry into normal telemetry. For example, the attacker changes `alarm_triggered` from 1 to 0, `event_type` from an anomaly condition to normal, target from 1 to 0, and `lift_valve_state` to open.

After the forged packet is created, the receiver attempts to decrypt and authenticate it. If the authentication tag does not match, the packet is rejected. This test shows whether ASCON-AEAD can detect unauthorized modification in both real-time transmission and stored telemetry records.

F. Performance Metrics

The benchmark records several performance and security metrics. The collected metrics are average encryption time per record, average decryption and verification time per record, encryption throughput, decryption throughput, average plaintext payload size, average ciphertext payload size, average size overhead, total number of forgery attempts, total number of rejected forgeries, and forgery rejection rate.

The benchmark output is stored in CSV files and visual plots. The benchmark.csv file stores per-record measurements, while summary_by_scenario.csv stores grouped results by event type. The generated plots are used to visualize latency and size overhead.

G. Experimental Procedure

The experiment was implemented in Python using uv for dependency management. First, the Volve workbook was parsed and converted into gas-lift SCADA-like telemetry records using gas_lift.py. Second, the generated telemetry records were processed by benchmark.py using each ASCON-AEAD variant independently. Third, false data injection was simulated on abnormal telemetry records. Finally, the performance results, integrity validation results, CSV logs, and plots were extracted from the results directory for analysis.

H. Limitations

This experiment uses generated SCADA-like telemetry derived from real historical Volve production data. Therefore, the generated telemetry should not be interpreted as actual raw industrial SCADA packet logs. In addition, the benchmark was executed using a Python implementation of ASCON, so the absolute latency values may differ from performance on embedded microcontrollers or industrial SCADA devices. However, the experiment remains useful for comparing ASCON-AEAD variants under the same software environment, dataset, and telemetry workload.

The main contribution of this experiment is not the creation of a new cryptographic algorithm, but the empirical evaluation of ASCON-AEAD for protecting gas-lift SCADA-like telemetry derived from real historical oil and gas production data.

IV. IMPLEMENTATION

A. Project Structure

The implementation consists of three main modules. The gas_lift.py module is responsible for generating gas-lift SCADA-like telemetry from the Volve Production Data

workbook. The crypto.py module provides the ASCON-AEAD encryption and decryption wrapper. The benchmark.py module runs the performance benchmark, simulates false data injection, and exports the experiment results.

The simplified **project structure**

```
src/ascon_sentry/
  __init__.py
  crypto.py
  gas_lift.py
  benchmark.py
tests/
  test_crypto.py
  test_gas_lift.py
```

The __init__.py file marks ascon_sentry as a Python package. The tests directory contains unit tests for the cryptographic and gas-lift telemetry logic.

B. Gas-Lift Telemetry Module

The gas_lift.py module reads the Volve Production Data workbook and extracts records from the daily production sheet. Since the Volve workbook contains historical production data instead of raw real-time SCADA packets, this module transforms the historical rows into SCADA-like telemetry records.

To simulate continuous SCADA telemetry, each historical production row can be expanded into multiple samples per day. The generated values use controlled deterministic jitter so that the telemetry records are not identical copies of the original historical row. The generated fields are summarized in Table I.

TABLE I. GENERATED GAS LIFT TELEMETRY FIELDS

Field Group	Fields	Purpose
Context Fields	`telemetry_id, timestamp, source_date, well_bore_code,npd_well_bore_name, facility_name`	Identify telemetry source and provide routing or indexing context
Operational Fields	`on_stream_hours, downhole_pressure_bar, downhole_temperature_c, dp_tubing_bar,annulus_pressure_bar, choke_size_pct,wellhead_pressure_bar, wellhead_temperature_c`	Represent physical gas-lift and well operating conditions
Production Fields	`oil_rate_sm3_d, produced_gas_rate_sm3_d,water_rate_sm3_d, gas_oil_ratio`	Represent production performance derived from Volve baseline data
Gas-Lift Fields	`estimated_gas_lift_rate_sm3_d, lift_valve_state,compressor_discharge_pressure_bar, compressor_power_kw`	Represent gas-lift injection and compressor-related behavior
Safety Labels	`alarm_triggered, event_type, target`	Mark normal or abnormal telemetry for FDIA simulation and analysis

The module also assigns event labels such as normal, `shut_in`, `low_lift_injection`, `tubing_instability`, `valve_instability`, or `high_annulus_pressure` based on the generated operational conditions. These labels are used for scenario grouping and false data injection simulation.

C. Cryptographic Module

The crypto.py module wraps the ASCON-AEAD functions provided by the Python ascon library. It supports three ASCON-AEAD variants used in this experiment: Ascon-128, Ascon-128a, and Ascon-80pq.

The module handles key generation, nonce construction, encryption, decryption, and authentication failure handling. Ascon-128 and Ascon-128a use a 16-byte key, while Ascon-80pq uses a 20-byte key. For each benchmark run, the nonce for every record is created from a random 64-bit prefix and a 64-bit counter. This design ensures that every record uses a unique nonce under the same key during one run.

If decryption fails because the authentication tag is invalid, the module raises an error. This behavior is used by the benchmark module to count forged packets that are rejected by ASCON-AEAD.

D. Benchmark Module

The benchmark.py module is responsible for executing the main experiment. It loads the generated gas-lift telemetry CSV file, separates each record into associated data and encrypted payload, and runs the selected ASCON-AEAD variant.

For every telemetry record, the module performs encryption, decryption, and authentication verification. It records plaintext size, ciphertext size, encryption time, decryption time, verification status, and size overhead.

The module also supports false data injection simulation. For abnormal telemetry records, it creates a forged payload by changing `alarm_triggered` to 0, `event_type` to normal, `target` to 0, and `lift_valve_state` to open. The forged payload is then tested against the authentication mechanism. If ASCON-AEAD rejects the forged packet, the event is counted as a rejected forgery.

E. Output Files

Each benchmark run creates one result folder. The output consists of four files:

1) benchmark.csv stores per-record benchmark measurements, including encryption time, decryption time, plaintext size, ciphertext size, verification status, and forgery detection status.
2) summary_by_scenario.csv stores aggregated metrics grouped by event type.
3) encryption_latency.png visualizes encryption and decryption latency across telemetry records.

4) size_overhead.png visualizes the difference between plaintext size and ciphertext size.

These files are used as the basis for the results and analysis section.

F. Testing

The implementation includes unit tests to verify the main program behavior. The cryptographic tests verify successful encryption-decryption round trips, authentication failure when ciphertext is modified, and support for all selected ASCON-AEAD variants. The gas-lift tests verify telemetry schema detection, separation between associated data and encrypted payload, and the false data injection transformation.

Successful execution of these tests indicates that the core encryption, decryption, telemetry handling, and attack simulation logic work as expected.

V. Results and Discussion

A. Dataset Distribution

The experiment used 2,000 generated gas-lift telemetry records derived from the Volve Production Data. The generated records were grouped into four event types: `normal`, `shut_in`, `low_lift_injection`, and `valve_instability`. The distribution of event types is shown in Table II.

TABLE II. Event Distribution in the Generated Telemetry Dataset

Event Type	Number of Records	Percentage
`low_lift_injection`	1,184	59.20%
`shut_in`	716	35.80%
`normal`	86	4.30%
`valve_instability`	14	0.70%
`Total`	2,000	100.00%

The dataset is dominated by abnormal telemetry records, especially `low_lift_injection` and `shut_in` events. This distribution is useful for evaluating integrity protection because most records can be used in the false data injection scenario. In total, 1,914 records were treated as abnormal and used for forgery simulation.

B. Performance Comparison

TABLE III. Overall ASCON-AEAD Performance Comparison

Variant	Avg. Enc. Time (us)	Avg. Dec. Time (us)	Enc. Throughput (KiB/s)	Dec. Throughput (KiB/s)
Ascon-128	4427.554	4429.695	125.862	129.328
Ascon-128a	3267.228	3275.091	170.560	174.922
Ascon-80pq	4434.818	4435.450	125.656	129.160

Based on the results, Ascon-128a achieved the best performance among the tested variants. Its average encryption time was 3267.228 us per record, which is lower than Ascon-128 and Ascon-80pq. Ascon-128a also achieved the highest encryption throughput at 170.560 KiB/s and the highest decryption throughput at 174.922 KiB/s.

Compared with Ascon-128, Ascon-128a reduced the average encryption time by approximately 26.2%. Compared with Ascon-80pq, Ascon-128a reduced the average encryption time by approximately 26.3%. This result is consistent with the expected behavior of Ascon-128a, which is designed with a larger rate and can process longer payload blocks more efficiently.

C. Latency Behavior

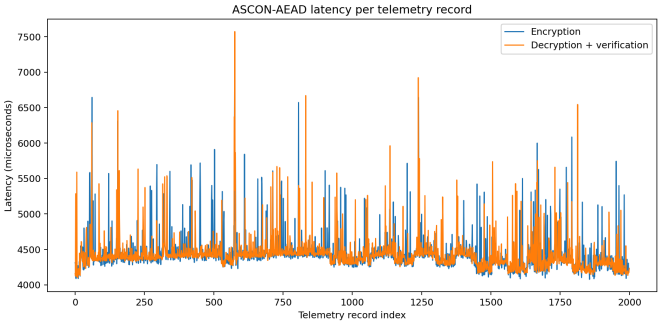

Fig 3. Latency Ascon-128

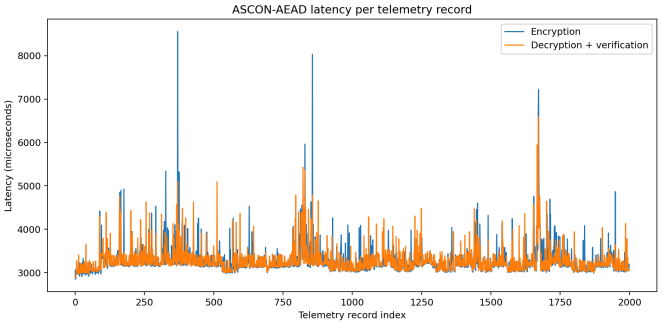

Fig 4. Latency Ascon-128a

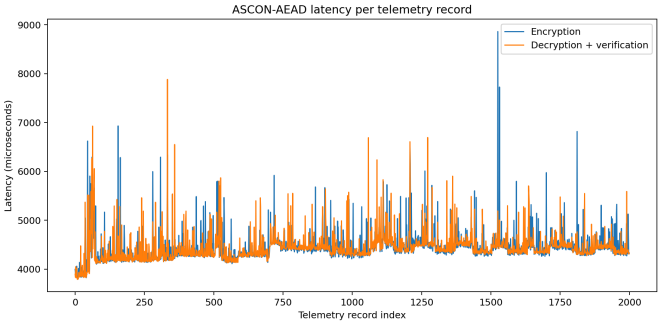

Fig 5. Latency Ascon-80pq

The latency plots in Fig. 3, Fig. 4, and Fig. 5 show the encryption and decryption latency for each telemetry record. In general, encryption and decryption times follow similar patterns for all variants. This indicates that the computational

cost of encryption and authenticated decryption is relatively balanced in the tested implementation.

Ascon-128 and Ascon-80pq show average latency around 4.4 ms per record, while Ascon-128a is faster at around 3.3 ms per record. Some latency spikes appear in all variants. These spikes are expected in a Python-based benchmark because execution time can be affected by interpreter overhead, operating system scheduling, memory management, and background processes. Therefore, the absolute latency values should not be interpreted as embedded hardware performance. However, the comparison is still valid because all variants were tested under the same environment and dataset.

D. Payload Size Overhead

TABLE IV. PAYLOAD SIZE OVERHEAD

Variant	Avg. Plaintext Size (bytes)	Avg. Ciphertext + Tag Size (bytes)	Avg. Overhead (bytes)	Overhead Percentage
Ascon-128	570.634	586.634	16	2.80%
Ascon-128a	570.634	586.634	16	2.80%
Ascon-80pq	570.634	586.634	16	2.80%

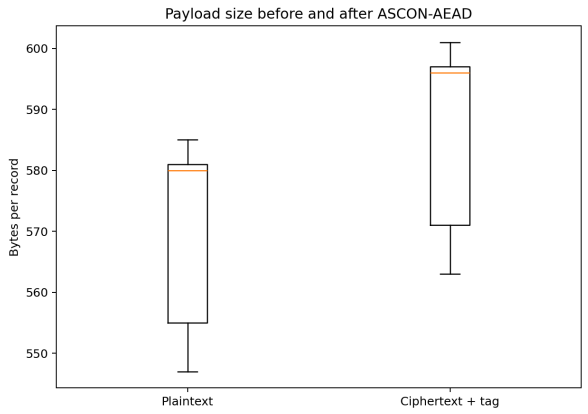

Fig 6. Payload Size Overhead Boxplot

All variants produced the same size overhead of 16 bytes per record. This overhead comes from the ASCON-AEAD authentication tag. The boxplots in Fig. 6 show that the ciphertext size distribution follows the plaintext size distribution with a constant 16-byte increase.

This result is important for SCADA telemetry because the payload size remains small even after protection. The overhead percentage is approximately 2.80% relative to the average plaintext size. This suggests that ASCON-AEAD can provide confidentiality and integrity protection without significantly increasing telemetry payload size.

It should be noted that this size metric only measures the encrypted payload output, which consists of ciphertext and authentication tag. The nonce and associated data are handled separately by the protocol and are not included in the ciphertext size measurement.

E. False Data Injection Detection

The false data injection scenario was used to evaluate whether ASCON-AEAD can detect unauthorized modification of telemetry records. In this scenario, abnormal records were modified to appear as normal records. The attacker attempted to change fields such as `alarm_triggered`, `event_type`, `target`, and `lift_valve_state`.

TABLE V. FALSE DATA INJECTION DETECTION RESULT

Variant	Forgery Attempts	Rejected Forgeries	Rejection Rate
Ascon-128	1,914	1,914	100%
Ascon-128a	1,914	1,914	100%
Ascon-80pq	1,914	1,914	100%

All variants rejected every forged packet. This result confirms that ASCON-AEAD successfully detects unauthorized modifications to the protected telemetry payload. Even if an attacker attempts to change an abnormal gas-lift condition into a normal condition, the authentication tag verification fails and the forged packet is rejected.

This result is significant for gas-lift SCADA telemetry because manipulated safety labels can lead to incorrect operational decisions. For example, changing a `low_lift_injection` or `shut_in condition` into normal could hide a real operational anomaly. The experiment shows that ASCON-AEAD provides not only confidentiality but also strong integrity protection for safety-critical telemetry fields.

F. Scenario-Based Performance

The summary results also show that performance is consistent across different event types. The average plaintext size varies slightly depending on the scenario. For example, `valve_instability` records have the largest average plaintext size, while `shut_in` records have the smallest average plaintext size. However, the overhead remains constant at 16 bytes for every scenario and every ASCON-AEAD variant.

The scenario-based results also show that only normal records were not used in the forgery scenario. All abnormal scenarios, including `low_lift_injection`, `shut_in`, and `valve_instability`, were used as forgery targets and all forged packets were rejected.

G. Discussion

The results show three main findings. First, Ascon-128a provides the best performance in this experiment. It has the lowest encryption and decryption latency and the highest throughput. This makes it the most efficient option among the tested variants for the generated gas-lift telemetry workload.

Second, all variants have the same payload size overhead. The additional 16 bytes per record come from the authentication tag. Since the average plaintext size is around 570 bytes, the relative overhead is small. This is suitable for

telemetry systems where bandwidth and message size may be limited.

Third, all variants achieved a 100% forgery rejection rate. This shows that ASCON-AEAD can detect false data injection attempts against protected gas-lift telemetry. The authentication tag prevents attackers from silently modifying operational values or safety labels.

Overall, the experiment indicates that ASCON-AEAD is suitable for securing gas-lift SCADA-like telemetry in a resource-constrained simulation. Among the tested variants, Ascon-128a offers the best performance while maintaining the same integrity protection and size overhead as the other variants.

VI. Conclusion

This paper evaluated ASCON-AEAD for securing gas-lift SCADA-like telemetry generated from the Volve Production Data. The experiment tested three ASCON-AEAD variants, namely Ascon-128, Ascon-128a, and Ascon-80pq, using 2,000 generated telemetry records.

The results show that Ascon-128a achieved the best overall performance, with the lowest encryption and decryption latency and the highest throughput. All variants produced the same 16-byte authentication tag overhead, which is small compared to the average plaintext payload size of 570.634 bytes.

The false data injection simulation also showed that all 1,914 forged abnormal records were rejected by every variant. This confirms that ASCON-AEAD can detect unauthorized modification of protected telemetry fields. Overall, ASCON-AEAD is suitable for protecting gas-lift SCADA-like telemetry in this resource-constrained simulation, with Ascon-128a being the most efficient variant in the tested environment.

Acknowledgment

The author expresses heartfelt gratitude to God Almighty for granting the strength and opportunity to complete this paper. The author also expresses sincere gratitude to Prof. Dr. Ir. Rinaldi Munir, M.T., the lecturer of the II4021 Cryptography course, for his guidance, knowledge, and inspiration throughout the learning process.

Appendix

The complete source code is available at: https://github.com/lidyamarth/asconsentry

References

[1] B. J. Challenger et al., "Innovative Method for Gas Lift Optimization on a Remote Satellite Offshore Platform: Reducing Environmental Footprint and Unblocking Hidden Optimization Opportunities," Society of Petroleum Engineers, SPE-220651-MS, Sep. 2024. [Online]. Available: https://doi.org/10.2118/220651-MS

[2] L. A. Hutchins, R. K. Burton, and D. J. Macintosh, "An Expert System For Analyzing Well Performance," Society of Petroleum Engineers, SPE-35705-MS, May 1996. [Online]. Available: https://doi.org/10.2118/35705-MS

[3] Equinor, "Volve Field Data Sharing Project," Equinor Energy, 2018. [Online]. Available: https://www.equinor.com/energy/volve-data-sharing

[4] National Institute of Standards and Technology, "NIST Selects 'Ascon' Family of Lightweight Cryptography Algorithms for Protecting Small Devices," NIST News and Events, Feb. 2023. [Online]. Available: https://www.nist.gov/news-events/news/2023/02/nist-selects-ascon-family-lightweight-cryptography-algorithms-protection

[5] Graz University of Technology, "Ascon Lightweight Cryptography," IAIK TU Graz, 2023. [Online]. Available: https://ascon.iaik.tugraz.at/

[6] PyPI, "ascon: Python implementation of the Ascon lightweight cipher suite," Python Package Index, 2024. [Online]. Available: https://pypi.org/project/ascon/

[7] National Institute of Standards and Technology, "Guide to Industrial Control Systems (ICS) Security," NIST Special Publication 800-82 Rev. 2, 2015. [Online]. Available: https://csrc.nist.gov/publications/detail/sp/800-82/rev-2/final

[8] Cybersecurity and Infrastructure Security Agency (CISA), "Joint Cyber Security Advisory: Threat Actor Profile and Mitigation on Colonial Pipeline," U.S. Department of Homeland Security, Alert AA21-131A, 2021. [Online]. Available: https://www.cisa.gov/news-events/cybersecurity-advisories/aa21-131a

[9] National Institute of Standards and Technology, "Ascon-Based Lightweight Cryptography Standards for Constrained Devices," NIST SP 800-232, 2025. [Online]. Available: https://csrc.nist.gov/pubs/sp/800/232/final

STATEMENT

I hereby declare that this paper is my own work, is not a translation or reproduction of another person's paper, and is not plagiarized.

Bandung, June 19, 2026

Lidya Marthadilla / 18223134