Implementation of Ascon based VPN for IoT Devices

Syed Ahad Ali
Computer Science
Habib University
Karachi, Pakistan
sa07753@st.habib.edu.pk

Asad Ullah Chaudhry
Computer Science
Habib University
Karachi, Pakistan
ac07408@st.habib.edu.pk

Azkaa Nasir
Computer Science
Habib University
Karachi, Pakistan
an08017@st.habib.edu.pk

Syeda Haya Fatima
Computer Science
Habib University
Karachi, Pakistan
sf07503@st.habib.edu.pk

Ilsa Shariff
Computer Science
Habib University
Karachi, Pakistan
is07310@st.habib.edu.pk

***Abstract*—The Internet of Things (IoT) has introduced unprecedented convenience and connectivity but faces critical security challenges due to constrained device resources and diverse architectures. Traditional cryptographic algorithms, while robust, are often too resource-intensive for IoT applications, leaving devices vulnerable to attacks such as hacking, data breaches, and denial-of-service (DoS). This report explores ASCON, a lightweight cryptographic protocol designed to address these challenges by providing efficient, secure encryption and authentication for resource-constrained environments. Leveraging ASCON's lightweight design and sponge-based construction, this study implements a Python-based authentication and verification system inspired by the Fiat-Shamir protocol to secure communication in IoT networks, particularly in VPNs. The report reviews ASCON's cryptographic operations, highlights its advantages over traditional algorithms, and demonstrates its ability to balance strong security with minimal resource usage. Through a simulation-based approach, this report underscores ASCON's potential to enhance IoT security and offers insights into its real-world applicability.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

IoT devices face, on average, a lot of security threats. Since these devices are built with limited processing power and memory to reduce costs, the implementation of traditional security methods becomes extremely difficult. Hence, as a result, IoT devices are vulnerable to various cyber threats, including hacking, data breaches and denial-of-service (DoS) attacks, which can compromise their functionality a lot and the sensitive information they handle.

VPNs (Virtual Private Networks) are used to secure communication by creating encrypted tunnels for the exchange of data. However to maintain security of a VPN, strong cipher algorithms must be implemented to prevent any unauthorized access. While there are many existing cryptographic algorithms which offer robust security, they are often too resource-intensive for IoT devices. This creates an urgent need for lightweight cryptographic solutions specifically tailored for resource constrained environments like IoT.

ASCON, a lightweight cryptographic protocol, is a promising solution. Its lightweight design is great for low power devices and provides strong encryption and authentication while using minimal computational resources.

This paper explores how ASCON can be used as a cipher algorithm to enhance VPN security for IoT systems. The goal is to implement ASCON using python to develop an authentication and verification system inspired by Fiat-Shamir protocol. Through this, we can demonstrate how ASCON's lightweight and efficient design makes it ideal for securing communication.

II. Literature Review

IoT devices face unique challenges due to their constrained resources and diverse architectures, making lightweight cryptographic algorithms critical for ensuring security. Ascon, a NIST-approved lightweight cryptographic standard, has emerged as a promising solution for securing IoT environments. The following review examines studies exploring Ascon's implementation and relevance in IoT systems.

Dobraunig et al. highlighted the dual capabilities of the Ascon cryptographic algorithm in authenticated encryption (Ascon-128) and hashing (Ascon-128a). The algorithm employs a 320-bit permutation, enabling secure data processing while minimizing computational overhead. This work emphasized Ascon's suitability for resource-constrained IoT devices due to its efficiency in balancing security and performance. For instance, its block-by-block encryption ensures each ciphertext block depends on the successful encryption of the preceding block, enhancing data integrity and confidentiality. [2]

Tran et al. investigated the integration of Ascon into the IPSec protocol's Encapsulating Security Payload (ESP) mode. A hardware architecture was proposed for real-time IoT systems, achieving a throughput of 8.806 Gbps and a latency of 427 ns for 64-byte packets. The study demonstrated Ascon's suitability for high-speed IoT networks, including applications such as real-time video streaming and large-scale data aggregation. The authors further showed that Ascon's high throughput-to-area (TPA) ratio outperforms traditional algorithms like AES-GCM by processing 128-bit data per clock cycle. [4]

Tariq et al. conducted a systematic review categorizing IoT vulnerabilities and exploring advanced security measures, including lightweight cryptography, blockchain, and machine learning. The review emphasized the need for interdisciplinary approaches to IoT security. While not focusing solely on Ascon, the study supported its relevance as a lightweight

cryptographic algorithm designed to address IoT-specific challenges. [3]

Dobraunig et al. further discussed the broader advantages of lightweight cryptography for IoT, highlighting Ascon's ability to balance strong encryption with minimal resource usage. The sponge construction of the algorithm enables efficient data processing in blocks, making it particularly useful for devices with limited power and memory. The study underscored Ascon's capability to secure communications without compromising performance requirements in IoT applications. [2]

Cintas-Canto et al. explored the implementation of Ascon using ChatGPT, leveraging the AI model's ability to generate human-like code. The methodology included iteratively generating and testing Python code for Ascon, identifying issues, and refining functions to align with the algorithm's specifications. This research highlighted the potential of AI tools in assisting with cryptographic algorithm development, enabling rapid prototyping and debugging of Ascon's encryption and decryption processes. [5]

The design and operational phases of Ascon were detailed by Dobraunig et al., who described its sponge-based methodology for authenticated encryption and hashing. The algorithm employs double-keyed initialization and finalization phases to enhance resistance against internal state attacks. The operational phases include initialization, associated data processing, plaintext/ciphertext processing, and finalization. These phases incorporate constant addition, substitution with a 5-bit S-box, and linear diffusion for strong cryptographic resistance. [2]

Dobraunig et al. also highlighted Ascon's efficiency in securing short messages, a common requirement in IoT communications. Its minimal memory footprint and support for low-energy operations make it ideal for constrained devices. Furthermore, the ability to process data with high throughput ensures compatibility with real-time communication scenarios. [2]

Cintas-Canto et al. noted that Ascon's design facilitates easy integration with countermeasures against side-channel attacks, such as masking, due to its low-degree S-box and bitsliced architecture. These features are particularly advantageous in IoT environments where physical security is often compromised. [5]

The reviewed studies demonstrate Ascon's effectiveness in addressing IoT security challenges through its lightweight, secure, and efficient cryptographic design. Its sponge-based architecture and applicability to resource-constrained environments make it a valuable solution for IoT security.

III. Methodology

A. System Design

Our system is designed as a Python-based sender-receiver model to simulate secure communication between two participants. The sender initiates the communication process by encrypting a message using the ASCON encryption protocol and transmitting it to the receiver. The receiver then authenticates the sender using pre-defined mechanisms and decrypts the received message, ensuring its integrity and confidentiality. The communication takes place over a simulated secure channel, which represents a conceptual VPN. This channel facilitates the exchange of messages while assuming a secure transport layer. Python's built-in libraries and socket programming are used to manage the flow of communication between the sender and receiver.

The process starts with the encryption phase, in which the sender encrypts a plaintext message using a shared key and a nonce. ASCON generates an authentication tag alongside the ciphertext, ensuring the message's integrity. During the transmission phase, the encrypted message and its authentication tag are sent through the simulated secure channel, representing the conceptual VPN. The decryption process is performed by the receiver, who uses the shared key to decrypt the ciphertext and validates the authentication tag to confirm the message's integrity. Finally, an integrity check determines whether the message is valid. If the tag is correct, the message is processed by the receiver; otherwise, the message is rejected, ensuring that no tampered data is accepted.

B. A Brief Breakdown of ASCON's Operations

a) Initialization:: Create a 320-bit initial state using the secret key (K), a 128-bit nonce (N), and an Initialization Vector (IV) that defines the algorithm's parameters. The IV includes details like the key size (k), rate (r), and the number of rounds for the initialization (a) and intermediate permutations (b). Then, apply a rounds of a round transformation, denoted as p, to this state. Following these rounds, the secret key (K) is XORed with a portion of the state.

b) Processing Associated Data:: ASCON processes any associated data (A) in blocks of r bits. If associated data is present, it is padded with a '1' followed by the necessary number of '0's to make its length a multiple of r. The padded data is then divided into blocks $(A_1, A_2, \ldots, A_s)$. Each associated data block is XORed with the first r bits of the state, followed by the application of b rounds of a permutation denoted as p_b. After processing all associated data blocks, a domain separation constant $(0319|1)$ is XORed with the state to distinguish it from the plaintext processing phase.

c) Processing Plaintext/Ciphertext:: The plaintext (P) is padded, divided into r-bit blocks $(P_1, P_2, \ldots, P_t)$, and processed iteratively in a similar manner.

During encryption, each plaintext block is XORed with the first r bits of the state to produce a ciphertext block (C). The state then undergoes b rounds of the p_b permutation, except for the last block. The final ciphertext block is truncated to match the length of the original last plaintext block fragment.

Decryption follows a similar process but replaces plaintext processing with ciphertext processing. Ciphertext blocks are XORed with the state to recover the corresponding plaintext blocks.

d) Finalization:: In the finalization stage, a rounds of the p_a permutation are applied to the state. The first r bits of

the resulting state are then XORed with the secret key (K) to generate the 128-bit authentication tag (T).

C. ASCON Permutations: Core Operations

The core of ASCON's operations lies in the two 320-bit permutations: p_a and p_b, which differ only in the number of rounds they execute. The number of rounds (a and b) are adjustable parameters impacting the security level. These permutations use a Substitution-Permutation-Network (SPN)-based round transformation (p) consisting of three steps:

- **Constant Addition (p_C):** A round-specific constant is added to a designated register word of the state in each round.
- **Substitution Layer (p_S):** This layer applies a 5-bit S-box 64 times in parallel to update each bit-slice of the 320-bit state. This S-box is designed for efficient hardware and software implementation using Boolean logic. The sources provide both a circuit diagram [source 1, Figure 2] and a lookup table [source 2, Table 7] for this S-box.
- **Linear Diffusion Layer (p_L):** This layer ensures diffusion within each 64-bit word of the state. It achieves this by rotating register words with different shift values and performing modulo-2 addition on the shifted words.

These three steps, when combined, form the round transformation p. The repeated application of p in the p_a and p_b permutations provides the cryptographic strength of ASCON.

IV. Conclusion

Our program successfully is able to encrypt and decrypt messages using the ASCON algorithm. The system resource usage measurement has not been a part of our project, but we can infer that the algorithm is lightweight and efficient due to its design. The implementation of the Fiat-Shamir protocol for authentication and verification further enhances the security of the communication process. The system can be further extended to include additional security features and optimizations for real-world deployment. Future work may involve integrating ASCON into IoT devices and evaluating its performance in resource-constrained environments. Overall, the implementation of ASCON in our system demonstrates its potential to enhance security in IoT networks and VPNs, offering a lightweight and secure cryptographic solution for secure communication.

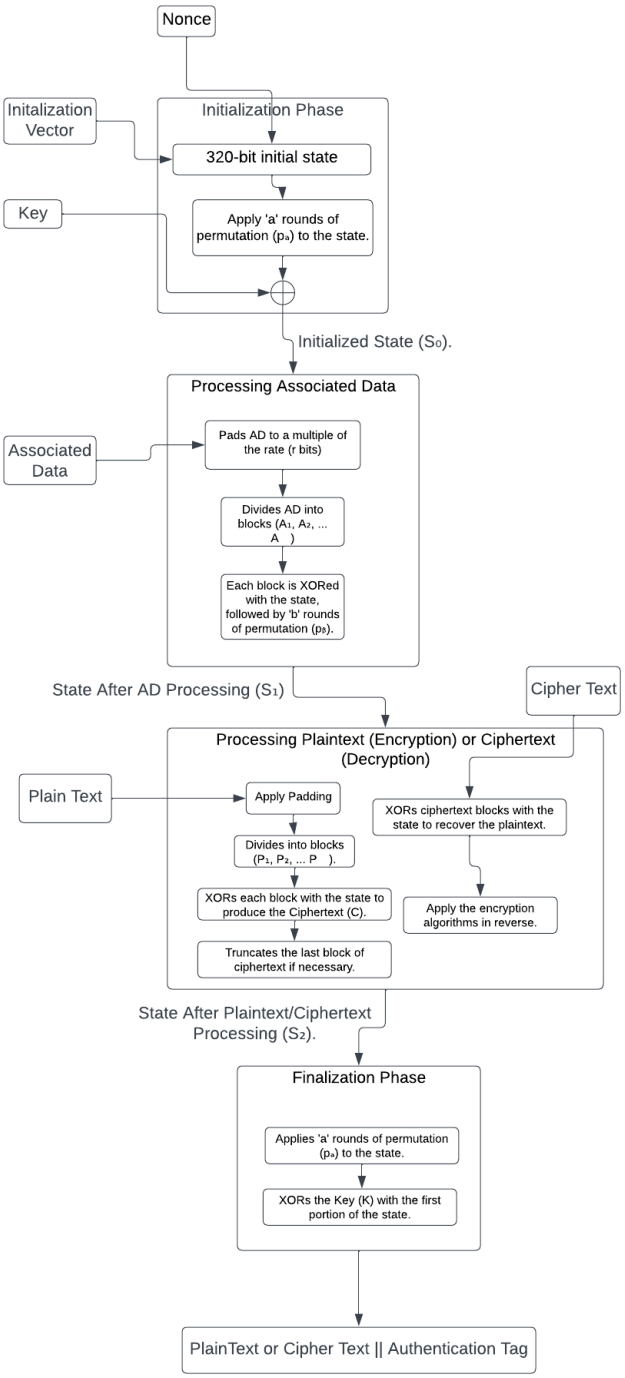

Fig. 1. A higher level block diagram of the encryption and decryption process in one.

References

[1] Zipit Wireless, "Complete Guide to VPNs for IoT," Zipit Wireless Blog, Jul. 15, 2023. [Online]. Available: https://www.zipitwireless.com/blog/complete-guide-to-vpns-for-iot. [Accessed: Nov. 24, 2024].

[2] C. Dobraunig, M. Eichlseder, F. Mendel, et al., "Ascon v1.2: Lightweight Authenticated Encryption and Hashing," J. Cryptol., vol. 34, no. 33, 2021. [Online]. Available: https://doi.org/10.1007/s00145-021-09398-9.

[3] U. Tariq, I. Ahmed, M. A. Khan, and A. K. Bashir, "Fortifying IoT against crimpling cyber-attacks: a systematic review," Karbala Int. J. Modern Sci., vol. 9, no. 4, 2023. [Online]. Available: https://doi.org/10.33640/2405-609X.3329.

[4] S.-N. Tran, V.-T. Hoang, and D.-H. Bui, “A Hardware Architecture of NIST Lightweight Cryptography Applied in IPSec to Secure High-Throughput Low-Latency IoT Networks,” IEEE Access, vol. 11, pp. 89240–89248, 2023. [Online]. Available: https://doi.org/10.1109/ACCESS.2023.3306420.

[5] A. Cintas-Canto, J. Kaur, M. Mozaffari-Kermani, and R. Azarderakhsh, “ChatGPT vs. Lightweight Security: First Work Implementing the NIST Cryptographic Standard ASCON,” arXiv preprint, 2023. [Online]. Available: https://arxiv.org/abs/2306.08178.