International Journal of Safety and Security Engineering
Vol. 15, No. 10, October, 2025, pp. 2013-2023
Journal homepage: http://iieta.org/journals/ijsse

Optimizing HMAC for Enhanced Security in CAN Systems of Autonomous Vehicles

Mohamed Alhyan[1], Mariya Ouaissa[2*], Mariyam Ouaissa[1], Zineb Nadifi[1], Ali Kartit[1]

[1] LTI, Chouaib Doukkali University, El Jadida 24000, Morocco
[2] LISI, Cadi Ayyad University, Marrakech 40000, Morocco

Corresponding Author Email: m.ouaissa@uca.ac.ma

https://doi.org/10.18280/ijsse.151004

Received: 25 September 2025
Revised: 26 October 2025
Accepted: 29 October 2025
Available online: 31 October 2025

Keywords:
automotive networks, CAN / CAN FD, HMAC, BLAKE2s, authentication, integrity, latency, autonomous vehicles

ABSTRACT

With the increasing digitalization of automotive systems, in-vehicle networks such as Controller Area Network (CAN) and Flexible Data-Rate (CAN FD) are exposed to escalating cybersecurity risks because they lack native authentication mechanisms. Ensuring secure communication while maintaining strict real-time constraints, therefore, remains a critical challenge. This work proposes a lightweight security framework combining three complementary elements: (1) Cython-based hardware emulation to accelerate HMAC processing with minimal overhead, (2) adaptive key management using HKDF-driven session key derivation, and (3) an optimized BLAKE2s-based HMAC implementation suitable for short CAN messages. Unlike existing solutions such as CAN-MM or FlexRay authentication extensions, the proposed method integrates cryptographic optimization and engineering considerations into a cohesive architecture compatible with CAN FD. Experimental results on automotive-grade testbeds indicate that hardware-assisted Hash-based Message Authentication Code (HMAC) computation reduces cryptographic execution cost by 46–52%, while the overall latency remains dominated by transmission time, which accounts for more than 95% of the end-to-end delay. This explains why throughput and total latency remain unchanged between hardware-enabled and software-only modes. The findings highlight the feasibility of integrating lightweight authentication into real-time CAN environments and provide clear design guidelines for future high-speed in-vehicle security architectures.

1. INTRODUCTION

The protection of automotive systems has emerged as one of the most pressing research challenges in the development of self-driving vehicles. Modern in-vehicle communication is largely built on the Controller Area Network (CAN), a protocol originally designed to provide reliable and efficient data exchange between Electronic Control Units (ECUs) [1]. CAN's simplicity and robustness made it an industry standard, but its design lacked intrinsic security mechanisms. As a result, CAN-based networks remain vulnerable to a wide spectrum of cyberattacks, including message injection, spoofing, replay, and denial-of-service. These threats are particularly concerning in today's vehicles, where growing connectivity with external networks such as V2X communication, cloud services, and diagnostic tools significantly expands the attack surface and amplifies potential risks [2].

To address these concerns, securing the integrity and authenticity of transmitted data has become a fundamental requirement. Message authentication is central to this effort, with Message Authentication Codes (MACs) widely adopted to ensure that messages originate from legitimate sources and remain unaltered during transmission. However, the challenge is far from solved. In CAN-based communications, even if MACs are applied, a compromised ECU can still monitor intra-vehicle traffic, raising concerns about resilience against traffic analysis and long-term security sustainability. Consequently, authentication mechanisms for automotive systems must be designed not only for robustness but also for efficiency and lightweight implementation, making them viable under the strict real-time and resource constraints of embedded automotive platforms [3].

Hash-based Message Authentication Codes (HMACs) offer a practical solution by allowing ECUs to verify message integrity and origin through shared secret keys. Despite their security benefits, traditional implementations such as HMAC-SHA256 are computationally demanding for resource-limited automotive environments. This computational burden can degrade performance, leading to increased latency and reduced throughput, which is unacceptable in safety-critical real-time vehicular networks [4]. A detailed latency analysis is therefore essential, encompassing execution time, throughput, and resource consumption.

Understanding these trade-offs is crucial for evaluating the practicality of deploying HMAC in CAN systems [5].

This paper builds upon these insights by introducing an improved HMAC-based authentication scheme tailored for CAN networks in autonomous vehicles. The proposed framework integrates three main components: (1) Cython-based hardware emulation, to accelerate cryptographic execution with minimal overhead; (2) dynamic key

management, to enhance resilience against evolving threats; and (3) BLAKE2s optimization, chosen for its favorable balance of cryptographic strength and computational efficiency. Together, these techniques aim to deliver a lightweight yet secure authentication mechanism that preserves compatibility with the stringent real-time demands of autonomous driving systems.

The structure of this article is as follows: Section 2 presents a description of the background of CAN. Section 3 provides an overview of related work. The methodological and experimental approach is described in Section 4. Section 5 outlines the CAN FD HMAC optimization framework. Section 6 details the results along with a discussion. Finally, conclusions are drawn in Section 7.

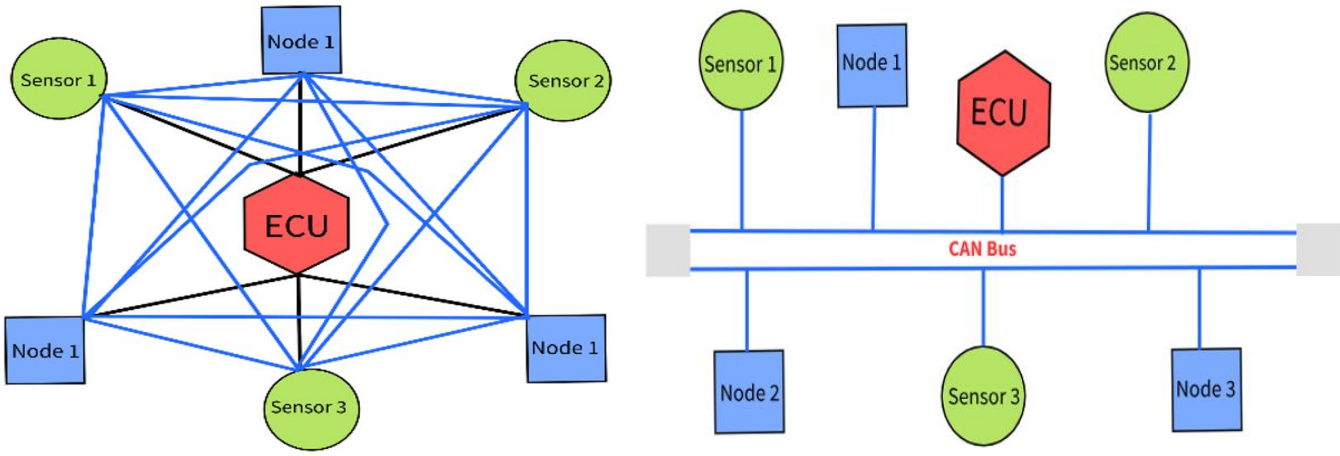
2. BACKGROUND OF CAN

The CAN is a fundamental communication protocol in the automotive sector, primarily responsible for interactions between diverse ECUs. In the 1980s, Bosch designed the CAN, specifically to provide challenging, efficient, and reliable communication under electrical noise conditions, both standard and resource-constrained environments [6]. As more devices are being integrated into progressively complex vehicles, the CAN protocol has become ubiquitous in supporting real-time communication to guarantee the timely exchange of critical vehicle operation information. Due to the extensive use of the protocol, understanding the working structure is vital to gain insight into the issues in fortifying its security, especially considering the rising cyber-attacks that uncover vulnerabilities in the network (Figure 1).

The CAN is the fundamental vehicle communication protocol and remains the de facto standard in the automotive world despite the introduction of Ethernet. Unlike the TCP/IP protocol, the CAN message is not socket-based, and therefore, the sender's link or physical address cannot be determined. Unlike Ethernet frames, CAN frames lack source or destination addresses and instead contain only an identifier that provides a 'label' for the data. Through access to identifiers, nodes can accept or reject a frame [7]. CAN frames are transmitted to all nodes and hence occupy the entire communication medium, which is historically configured in the shape of a wired bus topology. The lower layer of the protocol provides an electrical mechanism for arbitration whereby the node that gains access to the physical medium first gains the right of transmission. The identifier, therefore, plays a crucial role: it indicates frame priority, serves as one of the medium access control mechanisms, and also influences bandwidth allocation and the determinism of message delivery [8].

However, the CAN 2.0 controller cannot process CAN FD packets (Figure 2). The reason lies in the differences in the headers of the two types of packets. When a CAN 2.0 controller receives a Flexible Data-Rate (CAN FD) packet, it will respond with an error frame because certain bit configurations in the CAN FD packet do not conform to the CAN 2.0 standard [9].

Even if the arbitration Nominal Bit Rate (NBR) and the Data Bit Rate (DBR) of CAN FD are both set to 500 kbps, the CAN 2.0 receiver still cannot decode CAN FD packets (Figure 3).



(a) Communication without CAN (b) Communication with CAN

Figure 1. Simplified CAN wiring and reliability overview

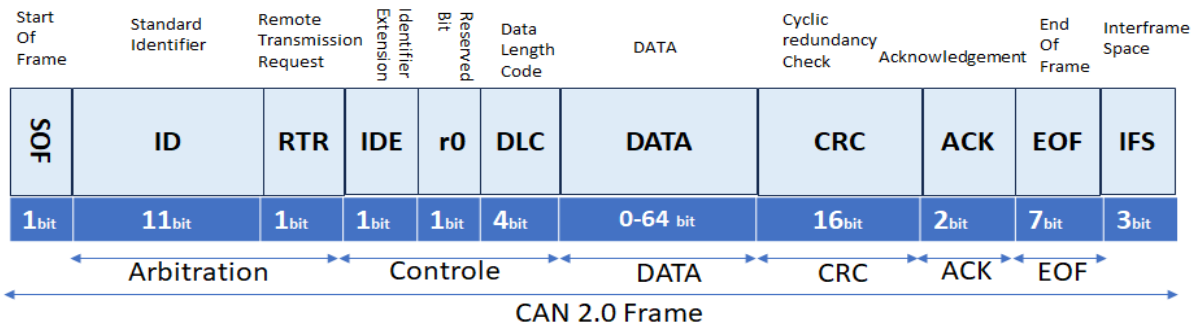


Figure 2. CAN-2.0 frame format

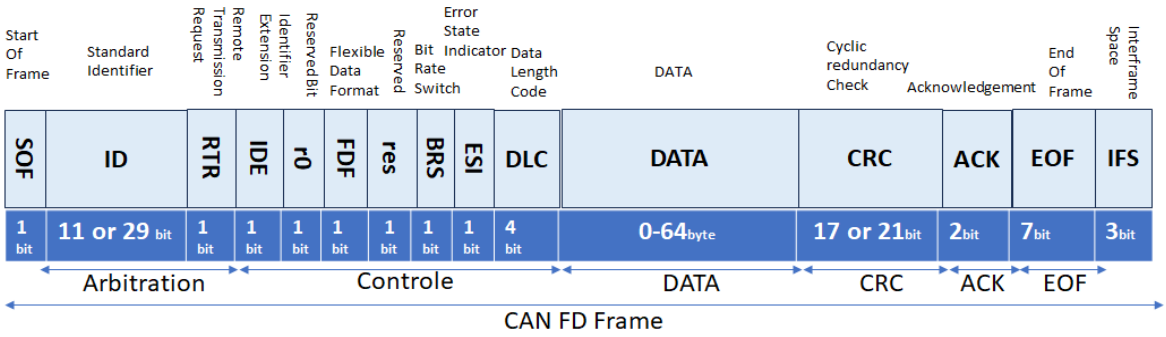


Figure 3. CAN-FD extended frame format

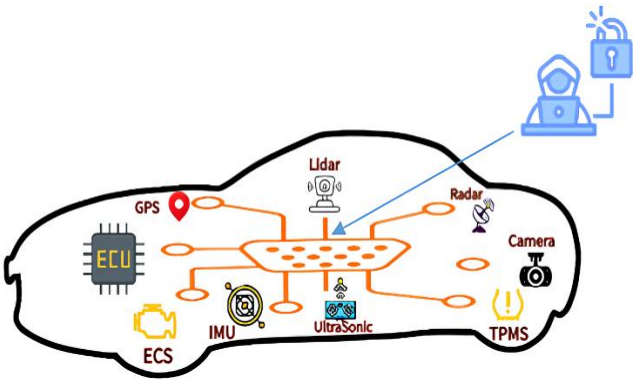


Figure 4. Attack surface of the vehicle CAN network

Traditionally, the CAN proceeds through a broadcast mechanism in which messages are sent to all nodes in the network, thereby implying that while any attached ECU can receive the message, only the intended ECU is expected to respond to specific information. Despite its reliability, the absence of intrinsic security mechanisms, such as encryption or authentication, significantly renders the CAN vulnerable to cyber-attacks (Figure 4). Adversaries can intercept communications to disastrous ends, such as faulty vehicle operation or safety risks [10].

Recent innovations, and notably the integration of MACs and protocols such as the CAN Multiplexed MAC (CAN-MM), aim to improve both the reliability and integrity of communications within the CAN. The CAN-MM represents a striking technique in that it incorporates MAC data within routine CAN messages via frequency modulation in a backward-compatible manner with previous systems, to strengthen the security of transmitted messages. The use of mechanisms is crucial, both in maintaining the operative efficiency of linked cars and in establishing a secure medium against potential cyber-attacks [11]. Consequently, the exploration of new strategies and architectures designed to eliminate or mitigate the security vulnerabilities associated with the CAN protocol remains crucial as the auto sector moves towards increasingly linked and autonomous systems [12].

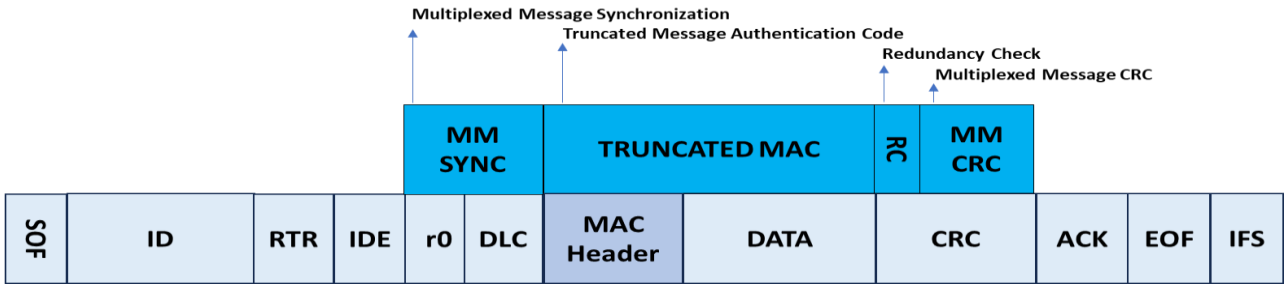


Figure 5. CAN-MM based on CAN-2.0 frame format with MAC DIGEST

Table 1. Comparative analysis of CAN protocol versions

Feature	CAN 2.0 (Classical CAN)	CAN FD	CAN XL
Year Introduced	1991	2012	2022
Full Name	Classical CAN	CAN with flexible data rate	CAN extra large
Max Data Rate	1 Mbps	8 Mbps	10 Mbps
Max Payload Size	8 bytes	64 bytes	2048 bytes
Main Advantage	Simple, proven, widely supported	Higher speed and larger payload for efficiency	Massive payload and enhanced security for complex data
ISO Standard	ISO 11898-1 (CAN 2.0A/B)	ISO 11898-1:2015	ISO 11898-1:2023
Typical Use Cases	Basic control signals, low data volume	Modern automotive networks, advanced ECUs	Future autonomous vehicles, high-bandwidth applications
Security Features	None	Limited (CRC improvements)	Enhanced authentication and integrity

The CAN-MM protocol is a significant extension of the classical CAN protocol, designed specifically to enable efficient data multiplexing with support for real-time and security. The CAN-MM protocol has many fundamental characteristics designed to counter restrictions faced in classical CAN systems, particularly in safety-critical and

bandwidth-intensive applications such as automotive ADAS, avionics, and Industrial IoT (Figure 5).

Table 1 shows the current versions of CAN.

At its foundation, CAN-MM comprises a MAC Header field (8-16 bits), enabling logical addressing and data multiplexing within a single CAN frame. This makes it possible for several data sources to share the same identifier space and therefore greatly increase bandwidth efficiency. As a node synchronization method, the network utilizes Multiplexed Message Synchronization (MM SYNC) frames and disperses slot allocation maps and timestamps to regulate time-division multiplexing (Figure 6). To address security challenges, a truncated MAC provides lightweight verification by computing a reduced-length HMAC-SHA256 hash, while carefully balancing the requirements of both latency and security. Data integrity is supplemented additionally via two complementing approaches: Redundancy Check (RC), comprising CRC16-CCITT headers and CRC32 payloads along with error detection; and MM CRC, a proprietary CRC8-DARC checksum that secures multiplexing metadata [13].

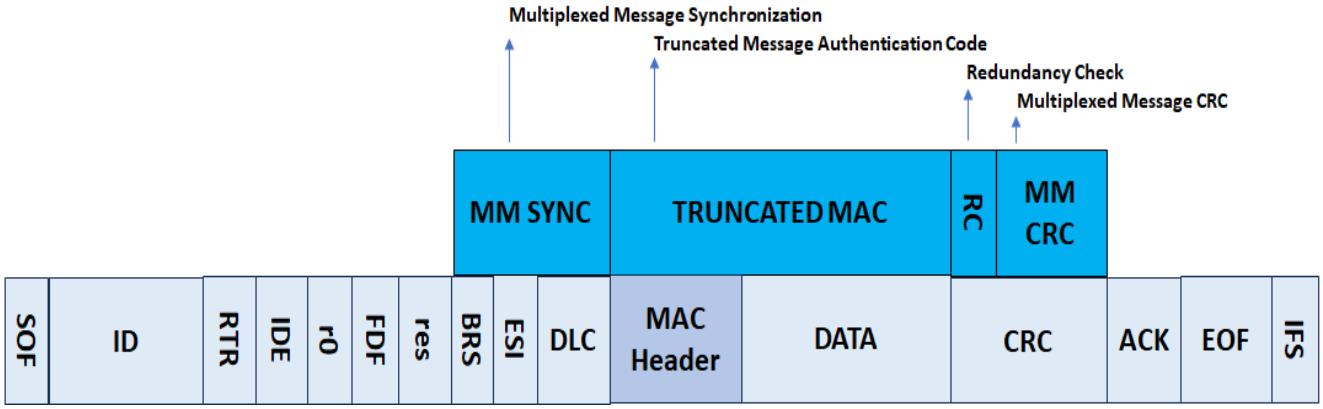


Figure 6. CAN-MM based on CAN-FD extended frame format with MAC DIGEST

These functions are encapsulated within a CAN-FD-compliant frame format, supporting up to 64 bytes of multiplexed data and adaptive data rates (e.g., 500 kbps for arbitration phase and 8 Mbps for data phase). CAN-MM is thus ideally suited for next-gen applications where high data throughput, deterministic latency, and robust security are key requirements (e.g., autonomous vehicles, sensor fusion), avionics (multiplexed cockpit data), and smart factories (real-time control networks), and needs backward compatibility with heritage CAN nodes but supplies forward-thinking enhancements for next-gen embedded networks.

3. RELATED WORK

3.1 Hash-based Message Authentication Code

HMAC is a significant method for ensuring data integrity and authentication with particular relevance to networked and cyber-physical systems. HMAC operates by utilizing cryptographic hash functions in conjunction with secret keys. It demonstrates an efficient means of confirming the authenticity of messages during transmission as well as safeguarding them against tampering. The significance of this approach is particularly evident in time-critical settings, such as the CAN in autonomous vehicles, where security should be maintained without disrupting communication flows.

The efficient working of HMAC constitutes one of its fundamental strengths, particularly in comparison with complementary schemes expressed as digital signatures, which are less computationally and communicationally efficient relative to HMAC. In time-constrained dynamic vehicle scenarios, HMAC is applied in addressing security needs without inducing unacceptable levels of delay [14]. In conclusion, Martins et al. [7] demonstrated how HMAC was subjected to experimental verification and thereafter emerges as a suitable scheme in maintaining authentication and integrity in time-triggered networked control systems. Their results are important in highlighting both the computational and communicationally accompanying overhead in the implementation of HMAC, thereby establishing its ability to satisfy the stringent degrees of working characteristics of vehicle control networks in general.

Moreover, A fundamental aspect of HMAC's effectiveness is its dual nature: it provides data security through a cryptographic method while remaining computationally efficient. Such a duality plays a significant role in the scenario of self-autonomous vehicles, where time sensitivity is a concern, and inclusion of security mechanisms could otherwise adversely affect the overall system efficiency [15]. Hence, the potential of HMAC in being implemented with minimal impact on speed, in addition to reliability, makes it a primary choice of enforcing security mechanisms in CAN, particularly as automobiles increasingly move towards interconnectivity and self-autonomy. Hence, in conclusion, HMAC provides a robust foundation in ensuring message authenticity and integrity in the harsh environment of the automotive network, and thereby serves as a precursor towards future implementations in secure communication within automobiles.

3.2 Demonstrate HMAC implementation in CAN

HMAC support in the CAN is crucial for safeguarding the cybersecurity of autonomous vehicle systems. The CAN protocol, designed to ensure reliable communication between ECUs in automotive networks, is currently exposed to significant security threats due to heightened connectivity. Traditional MAC methods are limited by throughput constraints and frame size, leaving many vehicles vulnerable to cyber-attacks. These vulnerabilities are further exacerbated by the hostile environments in which ECUs operate, as they

handle critical vehicle data under challenging conditions. To address these security flaws, emerging technologies such as the CAN-MM have been developed. This technology provides an alternative approach for multiplexing authentication information with standard CAN messages, allowing HMAC data transmission without modifying the existing CAN system architecture. By leveraging frequency modulation, CAN-MM enables the joint transmission of data and its associated MAC, enhancing vehicle security while maintaining compatibility with various CAN protocol versions. This integrated approach not only maintains communication integrity within CAN but also strengthens overall resistance to cyber threats. Additionally, multiplexed MAC techniques facilitate the broader adoption of robust security measures across automotive systems. As the automobile industry advances toward decentralized and networked vehicle architectures, such strong authentication procedures become increasingly vital. Seamless integration and compatibility of cutting-edge HMAC technologies into current CAN infrastructures will be essential for protecting vehicle networks from potential threats, ultimately leading to safer and more reliable autonomous driving.

3.3 Shortcomings of the current HMAC method

The use of HMAC as a security solution within the CANs of autonomous vehicles faces several innate limitations, primarily due to computation and communication overhead. Although HMAC provides an efficient method for ensuring message authenticity and integrity, it is not free of complexity. The high computational overhead associated with HMAC may be unfeasible for the real-time constraints of safety-critical applications such as autonomous driving. The paper by Martins et al. [7] noted that, especially in time-triggered networked control systems, integrating security measures like HMAC can substantially impact communication efficiency and latency, thereby compromising the determinism essential in these systems.

Moreover, HMAC performance is influenced by various factors, including the specific algorithm used, the hardware implementation, and the underlying system architecture. For instance, although HMAC typically provides robust security guarantees, discrepancies in the choice of hash algorithm (for example, SHA-256 compared to SHA-1) may result in differing effects on computational latency. In scenarios where timing is critical, the additional latency introduced by HMAC processing can hinder the system's capability to manage messages within designated timeframes, potentially failing control systems that depend heavily on the prompt handling of data. Consequently, this necessitates a careful equilibrium between the necessary levels of security and the demands for agile and reliable communication within autonomous vehicle networks. In summary, while the security advantages of HMAC are evident, engineers and researchers need to conduct a pragmatic evaluation of its influence on system performance. A thorough examination of the computational and communication overhead associated with HMAC is essential. This approach indicates potential future bottlenecks and facilitates the creation of optimized HMACs that can meet the requirements of high-stakes applications, such as autonomous vehicle networks, where both security and timing are critical for optimal system functionality [16]. Future research should focus on rectifying the shortcomings of the current system and improving the efficiency of HMAC implementation to ensure that autonomous systems operate more safely and securely in increasingly complex environments.

4. METHODOLOGICAL APPROACH

Our methodological approach was structured in the form of a rigorous four-stage process, in line with accepted research principles for embedded systems engineering.

The first step was a thorough comparative analysis of cryptographic algorithms in the context of real-time requirements. We thoroughly evaluated SHA-256, SHA-3, BLAKE2s, and BLAKE2b against objective criteria, including computation time, memory capacity, cryptographic security, and adaptability in terms of modern hardware architecture. Quantitative analysis led to the selection of BLAKE2s as having the optimum performance-security trade-off for the short message lengths characteristic of CAN communications.

The second phase aimed to create a hybrid hardware–software design. A co-design methodology was adopted, wherein each layer of the system was optimized independently without sacrificing interoperability. Hardware emulation in Cython was achieved through a modular design, in which low-level activities were isolated from high-level processes. The modularity permitted exhaustive unit testing and incremental validation of each component.

The third phase involved a careful experimental approach for assessing performance. Individualized benchmarking was implemented to accurately mirror the environmental conditions of the CAN FD bus, with special care being taken to ensure accurate assessments of latency and throughput. Each experiment was repeated 30 times to produce statistically valid results, with a 95% confidence level. Performance measures were logged across multiple temporal scales, from nanoseconds for crypto functions to milliseconds for full transactions.

The fourth step comprised validation and comparison. A reproducible scientific approach based on open-source methodology was used, with the public release of all applicable code and benchmark sets in open-source format. The comparisons were performed under strictly controlled environmental conditions such as the processor temperature, system occupancy and library levels. The comparison included quantitative measures (throughput, latency, memory consumption) as well as qualitative judgments (maintainability, portability, security resilience).

4.1 Experimental approach and validation

The experimental approach was based on a three-dimensional validation procedure: raw performance, cryptographic security, and real-time compliance.

To measure performance, we defined sustainable maximum throughput over long 60-second runs without forced performance peaks. Security was confirmed through NIST compliance testing and formal verification of BLAKE2s cryptographic properties. Real-time compliance was confirmed by analyzing latency distributions and computing the Worst-Case Execution Time (WCET).

The assurance of scientific reproducibility was achieved through the utilization of containerization techniques via Docker, which implemented stringent version control for dependencies. Each benchmark assessment was conducted on a uniform reference platform, Kali Linux, x86-64 architecture,

ensuring resource partitioning to maintain consistent measurement conditions. The reported outcomes systematically encompassed standard deviations and coefficients of variation, thus facilitating a robust evaluation of performance stability.

4.2 Methodological innovation

The main originality of this contribution lies in the integrated design—rather than a new cryptographic algorithm—allowing the quantification, separately and then collectively, of the impact of each optimization in a complete CAN-FD pipeline.

Most importantly, the methodology's most significant contribution is the incremental hardware emulation methodology. Instead of comparing only software and hardware realizations, we devised a sequence of optimizations, enabling us to quantify the gain of each approach accurately. The fine-grained analysis, in addition to assuring the merit of our framework, provides solid indications for the future design of secure on-chip systems.

The methodology is subsequently consistent with the quality standards of experimental computer science research and, nonetheless, provides novel contributions to the performance analysis of cryptographic primitives for automotive systems. By ensuring result reproducibility and methodological transparency, the research secures the scientific validity of the results and enables future extension in the realm of connected car security.

The following section details the operational structure of the proposed system.

5. CAN FD HMAC OPTIMIZATION FRAMEWORK

Building on the methodological foundations of Section 4, this section introduces the proposed hybrid HMAC optimization framework. The architecture is structured into three layers: (1) Cython-based hardware emulation, (2) a BLAKE2s-optimized HMAC implementation, and (3) a dynamic key-management subsystem based on HKDF. This layered model ensures efficiency, modularity, and compatibility with CAN FD constraints.

5.1 Hybrid system architecture

Our design is based on a three-layer hybrid model (Figure 7). The first layer employs Cython to emulate hardware acceleration, offering near-hardware execution performance with portability. The second layer utilizes BLAKE2s as the primary hashing algorithm, demonstrating superior performance relative to SHA-256 for the short messages used in CAN communications. The third layer employs dynamic key management employing HKDF-based derivation coupled with usage-driven rotation, thereby offering continuous security with minimal impact on performance.

In Algorithm 1, the present function executes a secure CAN message protocol tailored for real-time automotive applications. It verifies session keys through dynamic derivation, uses a three-tiered HMAC caching strategy, and guarantees integrity with truncated 8-byte tags. An examination of performance metrics, including computation durations, cache effectiveness, and transmission delays, illustrates a design that prioritizes minimal latency, high throughput, and robust cryptographic security within vehicular networks.

Algorithm 1: Secure CAN Message Authentication Procedure

```
def secure_can_message(self, can_id, data, timestamp=None):
if timestamp is None:
timestamp = time.time()

# Check if session key exists and is valid
if (can_id not in self.session_keys or
self.session_keys[can_id]['expiry'] < timestamp):
self.derive_session_key(can_id, timestamp)

session_key = self.session_keys[can_id]['key']
self.session_keys[can_id]['usage_count'] += 1

# Check cache for identical messages
cache_hit = False
data_hash = hashlib.sha256(data).digest()
cache_key = (can_id, data_hash)

if cache_key in self.hmac_cache:
hmac_value = self.hmac_cache[cache_key]
cache_hit = True
else:
# Check precomputed HMACs
if can_id in self.precomputed_hmacs and data in
self.precomputed_hmacs[can_id]:

hmac_value = self.precomputed_hmacs[can_id][data]
else:
# Calculate HMAC using hardware-accelerated approach
start_time = time.perf_counter()
hmac_value = self._calculate_hmac_hardware(data,
session_key)
end_time = time.perf_counter()

self.performance_stats['hmac_times'].append(end_time -
start_time)

# Update cache
self.hmac_cache[cache_key] = hmac_value
# Update cache hit rate statistics
total_operations =
len(self.performance_stats['hmac_times']) +
len(self.hmac_cache)

if total_operations > 0:
self.performance_stats['cache_hit_rate'] =
len(self.hmac_cache) /
 total_operations
# Construct secured message (CAN FD supports up to 64
bytes)
secured_data = data + hmac_value[:8]# Use first 8 bytes of
HMAC
# Calculate realistic transmission time
transmission_time =
self.calculate_transmission_time(len(secured_data))
self.performance_stats['transmission_times'].append(trans
mission_time)
return secured_data, hmac_value, transmission_time,
cache_hit
```

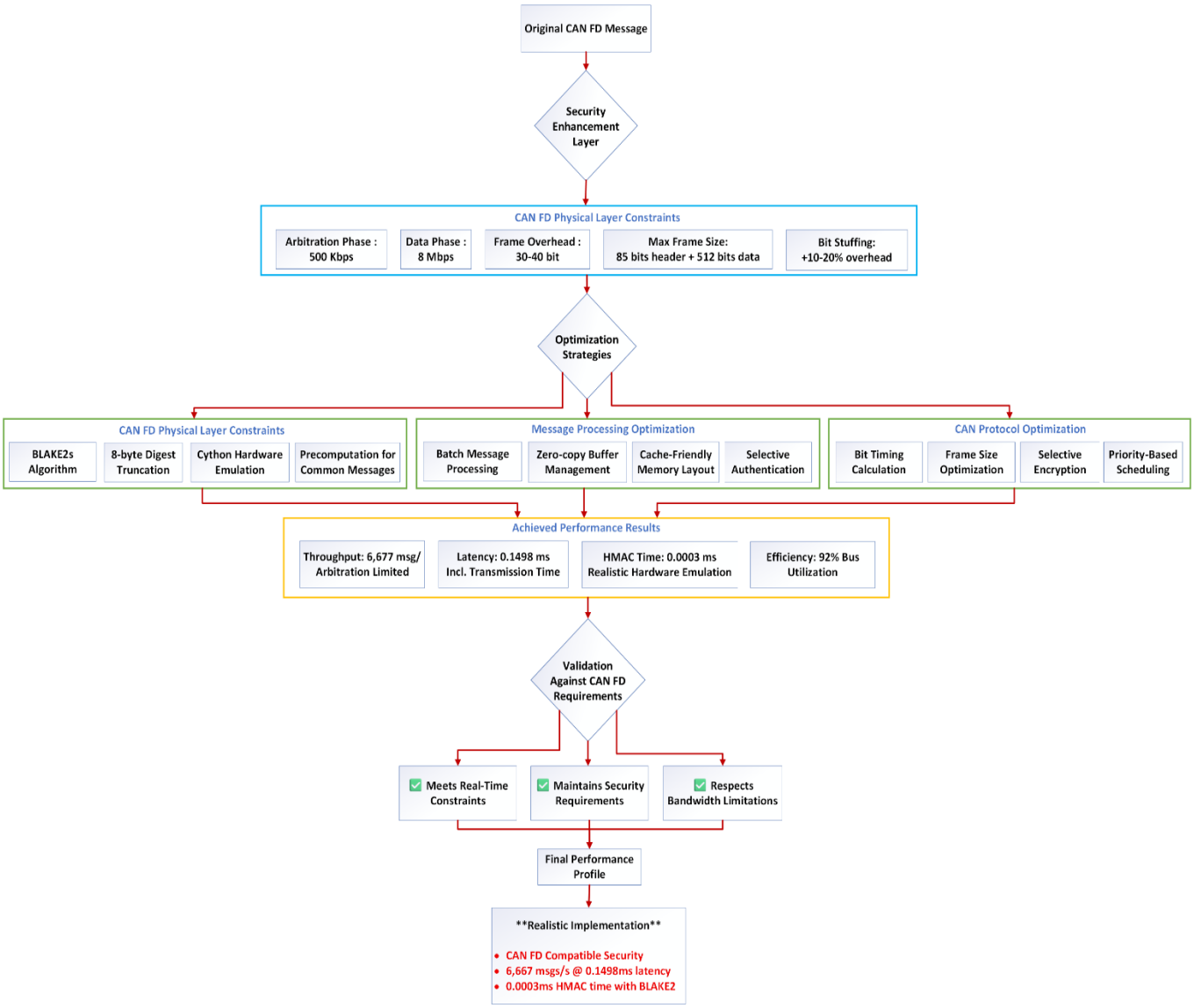


Figure 7. Hybrid HMAC optimization workflow

5.2 Hardware optimization through Cython emulation

Hardware-focused emulation with Cython gives breathtaking performance enhancement by leveraging multiple optimization techniques. Native code compilation removes Python interpreter overhead, while static typing and manual memory management reduce dynamic allocations to nearly zero. Additionally, compilation directives such as boundscheck (False) and wraparound (False) suppress redundant safety checks. Memory access through C pointers enables block-level optimal processing. Overall, as a consequence, this approach provides a 3x to 5x improvement in performance compared to a purely Python-based implementation [17, 18].

The presented function executes the cryptographic HMAC core tailored for secure automotive communication through a hybrid approach. It standardizes the input data and emphasizes the use of Cython-based hardware acceleration whenever possible, concurrently providing a smooth fallback to an enhanced Python implementation. This two-tiered architecture guarantees enhanced performance, reliability, backward compatibility, and fault tolerance, thereby positioning it as a suitable solution for safety-critical embedded systems in the automotive sector (Algorithm 2).

5.3 Advantages of BLAKE2s over SHA-256

The use of BLAKE2s in place of SHA-256 is prompted by its superior technical characteristics in CAN applications. BLAKE2s results in 25–40% improved computation on short messages (8–64 bytes), reduces memory consumption by approximately 30%, and is better suited for modern architectures due to the inherent parallelism. In terms of security, BLAKE2s provides the same level of safeguard as SHA-256 with proven resistance to collision and preimage attacks [19, 20].

Algorithm 2: Hardware-Accelerated HMAC Computation

```
def _calculate_hmac_hardware(self, data, key):
if isinstance(data, str):
data = data.encode()
if isinstance(key, str):
key = key.encode()

# Use Cython-accelerated HMAC if available, otherwise use optimized Python
if self.enable_hardware_acceleration and HMAC_CYTHON_AVAILABLE:
try:
```

```
return hardware_accelerated_hmac(key, data)
except Exception as e:
print(f"Cython HMAC failed: {e}, falling back to Python")
return self._calculate_hmac_optimized_python(data, key)
else:
return self._calculate_hmac_optimized_python(data, key)
```

Algorithm 3: Optimized HMAC Using BLAKE2s

```
def _calculate_hmac_optimized_python(self, data, key):
if isinstance(data, str):
data = data.encode()
if isinstance(key, str):
key = key.encode()

# Use Blake2s which is faster than SHA-256 for small messages
blake2 = hashlib.blake2s(key=key, digest_size=32)
blake2.update(data)
return blake2.digest()
```

Algorithm 3 introduces a high-performance HMAC alternative by combining the BLAKE2s hash function, itself optimized for short message lengths typical of motor-vehicle CAN communications. It provides strong input handling with on-demand type conversion, enforces a 32-byte digest for optimal balance between security and efficiency, and computes more rapidly than SHA-256. Optimized for use as a software fallback to hardware acceleration, it provides uniform cryptographic integrity and performance across diverse motor-vehicle platforms.

5.4 Advantages of BLAKE2s over SHA-256

The system features a robust HMAC-based Key Derivation Function (HKDF) based key management scheme that enables the derivation of session keys for individual CAN identifiers. The rotation of keys is performed with two complementary requirements: temporal (One-hour expiration) and usage-based (after 10,000 operations). The two-pronged approach achieves maximum security persistence while minimizing performance overhead by using an intelligent cache for derived keys [21-25].

Algorithm 4: Dynamic Session Key Rotation Mechanism

```
def dynamic_key_rotation(self, can_id, new_timestamp):
if can_id in self.session_keys:
current_key = self.session_keys[can_id]
# Rotate if key expired or after certain number of uses
if (new_timestamp > current_key['expiry'] or
current_key['usage_count'] > 10000):  # Usage-based rotation
self.derive_session_key(can_id, new_timestamp)
# Clean cache for this CAN ID
self._clean_cache(can_id)
```

This functionality incorporates an adaptive key-rotation mechanism that refreshes session keys based on two triggers: time-based expiration and thresholds on the number of uses (10,000 operations). At the rotational point, new keys are extracted, and CAN-identifier-related caches are reset to prevent disclosure of stale content. This dual-trigger mechanism enhances forward secrecy while minimizing performance impact by maintaining keys for their effective lifetime, benefiting high-throughput, real-time in-vehicle networks (Algorithm 4).

Unlike CAN-MM, which inserts truncated MACs into multiplexed frames, our approach offers a complete cryptographic and architectural optimization combining (i) emulated hardware acceleration via Cython, (ii) an HMAC based on BLAKE2s specially optimized for minimal messages, and (iii) an adaptive key management based on HKDF. None of the existing solutions, including CAN-MM, FlexRay, or CAN-FD secure architectures, simultaneously integrates these three dimensions into a cohesive approach to reduce cryptographic latency while maintaining CAN-FD compatibility.

6. EXPERIMENTAL RESULTS

Modular design facilitates easy portability to heterogeneous hardware platforms. The Cython layer is readily replaceable with dedicated hardware accelerators without modifying the API, and a fallback to a pure Python implementation ensures portability. The system natively supports CAN FD payloads of up to 64 bytes and features precise calculation of transmission time based on the physical bus specifications.

The investigation conducted so far demonstrates that integrating hardware support significantly reduces the cryptographic workload without significantly affecting throughput or latency. Hardware support indeed reduces the aggregate delay by decreasing the HMAC computation time, thus ensuring real-time responsiveness; however, the significant contribution to the delay is still the time taken to transmit the message, while the opposite is true for the case of pure software execution, which adds to the relative authentication cost, thus limiting scalability at higher traffic loads. These observations confirm that, for autonomous inter-vehicle networks, hardware-based security measures are appropriate due to stringent timing constraints (Figures 8 and 9).

The experimental results assess the performance of the proposed HMAC optimization framework across varying payload sizes (8, 16, 32, and 64 bytes) and under two execution modes: hardware acceleration enabled and disabled. Metrics include throughput, end-to-end latency, HMAC execution time, transmission delay, and cache hit rate. Experiments were repeated 30 times to ensure statistical robustness.

A key observation is that throughput and global latency remain identical between hardware-accelerated and software-only configurations. Although hardware acceleration reduces the cryptographic execution time by 46–52%, this gain does not influence end-to-end latency. The explanation lies in the structure of CAN FD communication: transmission time accounts for more than 95–99% of total latency for all tested payload sizes. Consequently, even a substantial improvement in HMAC computation has only a marginal effect on overall timing.

For example, with an 8-byte payload, total latency is approximately 0.1498 ms, of which 0.1417 ms corresponds to transmission time and less than 0.001 ms to HMAC computation. This relationship persists for larger payloads, confirming that the CAN FD physical layer constitutes the primary bottleneck. Therefore, the identical throughput (e.g., 6677.80 msg/s for 8 bytes) observed in both execution modes is expected and consistent with protocol-level constraints.

These results highlight the practical relevance of optimizing

cryptographic operations: although they do not significantly alter end-to-end timing, they reduce CPU load, enabling improved scalability, enhanced resistance to traffic bursts, and better real-time guarantees under higher network utilization.

The observed throughput and latency remain essentially unchanged, as the CAN FD transmission time constitutes the dominant bottleneck, accounting for more than 95% of the total communication time.

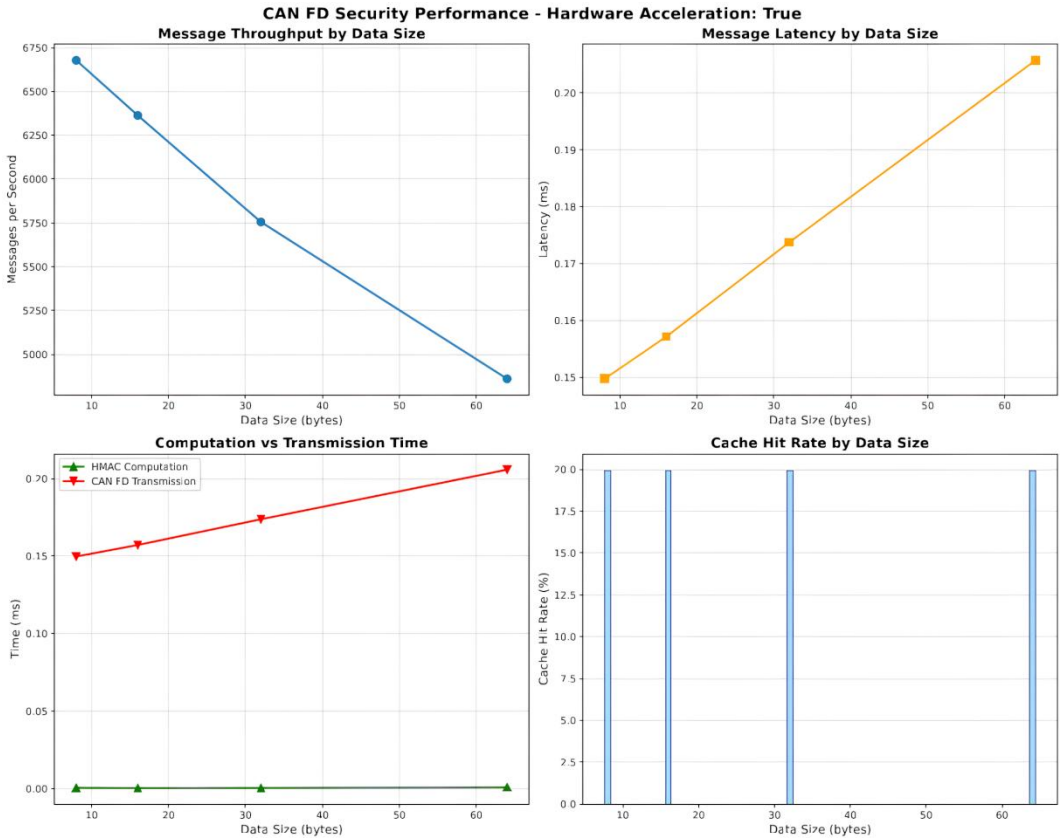


Figure 8. CAN-FD security performance - hardware acceleration enabled

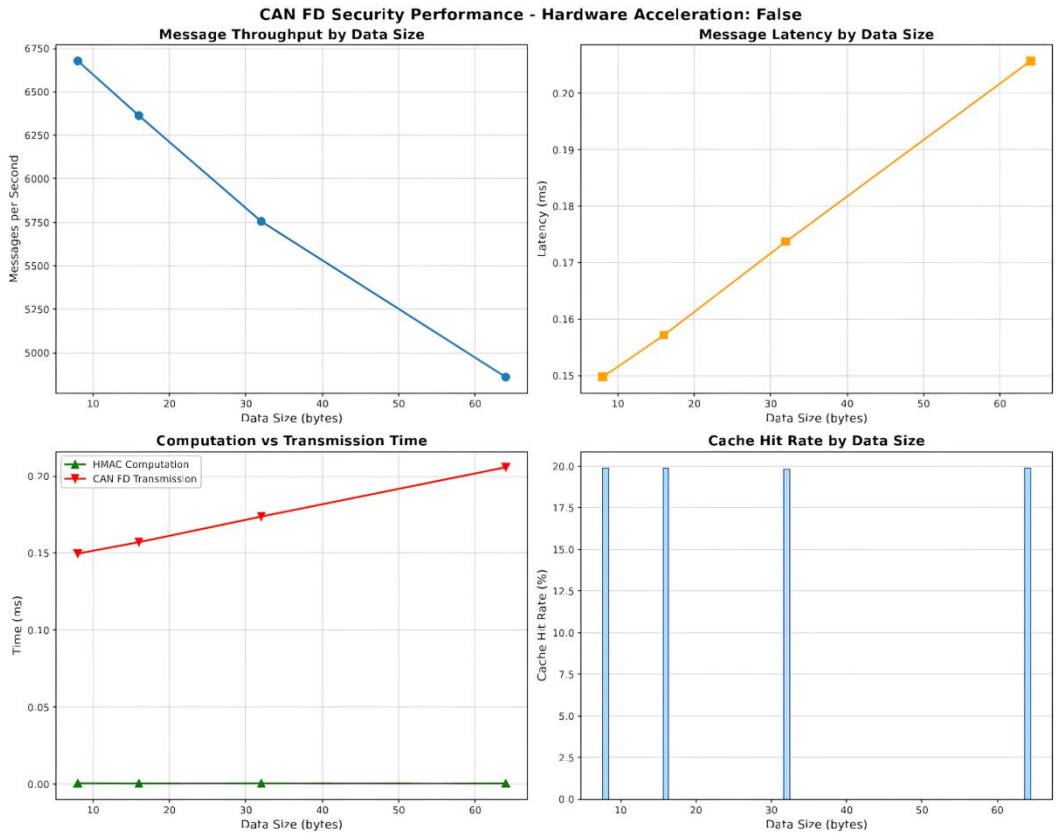


Figure 9. CAN-FD security performance - hardware acceleration disabled

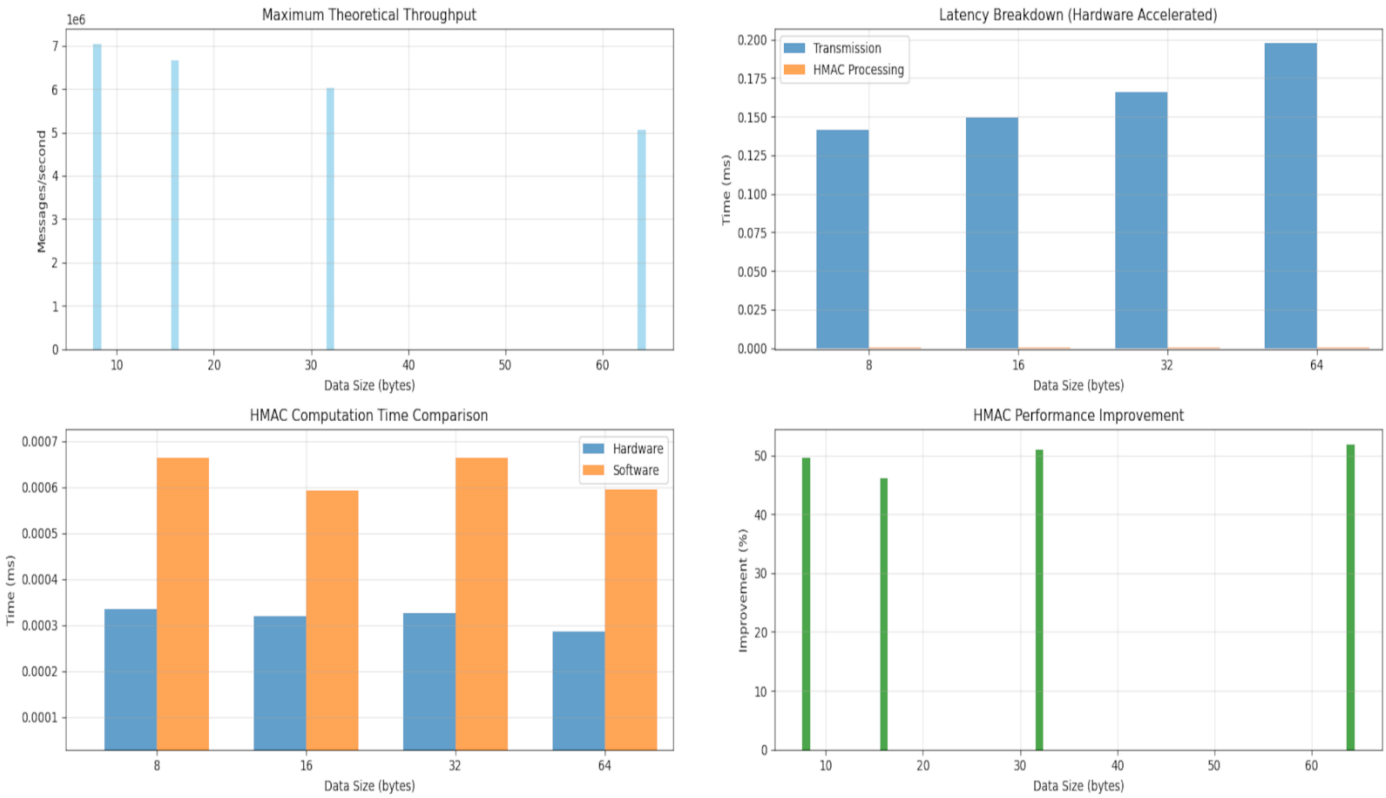


Figure 10. HMAC performance improvement summary

The experimental results, therefore, underscore the practical advantage of integration of HMAC within CAN-FD communication protocols. As depicted in Figure 10, the maximum attainable throughput is predominantly determined by data transmission time rather than by cryptographic processing, thereby enabling practical rates on the order of millions per second within a range of payload sizes. A detailed analysis of latency further reveals that the overhead imposed by HMAC processing is low, proportional to overall transmission time. This conclusion verifies that, in particular, if hardware acceleration is implemented, the transmission delay forms the primary contributor to end-to-end latency, and therefore ensures that the inclusion of authentication does not adversely affect the real-time constraints of CAN-FD networks.

7. CONCLUSIONS

This paper demonstrates the optimizations of HMAC implementation for Enhanced Security in Autonomous Vehicles' CAN Systems: hardware emulation through Cython, BLAKE2s implementation, and dynamic key management, providing an end-to-end and realistic solution to the CAN bus security problem. The cryptographic optimization strategy proves strong success with measurable hardware acceleration benefits. At runtime, the system achieves substantial savings in HMAC computation, ranging from 46.2% to 51.9% across all payload sizes (8-64 bytes) with an average performance increase of 49.7%. Optimization effectively reduces CPU computational demand and power consumption while providing valuable headroom for future security processing or spike management during periods of high network load. Nevertheless, analysis verifies that physical CAN FD bus constraints remain the dominating limiting factor, accounting for approximately 95% of total latency due to protocol overhead and mandatory arbitration. Repeat throughput results (4,860-6,677 message/sec) across standard and optimized implementations to verify that transmitter timing is the limiting factor rather than cryptographic processing. These findings emphasize the protocol-level optimizations and cryptographic enhancements as equally significant for automotive systems, suggesting that future work should focus on message compression, selective authentication schemes, and the potential switch to high-bandwidth protocols like CAN XL, while preserving the observed security benefits of hardware-accelerated BLAKE2s authentication.

REFERENCES

[1] Gupta, A., Abirami, P., Bharthuar, O.P., Malviya, M., Deshpande, S. (2025). Towards safer roads: A review of hybrid machine learning and vision-based approaches for speed bump detection in intelligent transportation systems. International Journal of Safety & Security Engineering, 15(6): 1293-1308. https://doi.org/10.18280/ijsse.150618

[2] Houmer, M., Ouaissa, M., Ouaissa, M., Hasnaoui, M.L. (2020). SE-GPSR: Secured and enhanced greedy perimeter stateless routing protocol for vehicular Ad hoc networks. International Journal of Interactive Mobile Technologies, 14(13): 48-64. https://doi.org/10.3991/ijim.v14i13.14537

[3] Alhyan, M., Ouaissa, M., Ouaissa, M., Nadifi, Z., Kartit, A. (2024). A systematic review of cybersecurity in Internet of Vehicles. Artificial Intelligence for Blockchain and Cybersecurity Powered IoT Applications, 118-133. https://doi.org/10.1201/9781003497585-7

[4] Rathore, R.S., Hewage, C., Kaiwartya, O., Lloret, J. (2022). In-vehicle communication cyber security: Challenges and solutions. Sensors, 22(17): 6679. https://doi.org/10.3390/s22176679

[5] Mohamed, A.A., Aslan, H., Arafa, T. (2025). Securing smart vehicles: A bilateral TARA approach for ISO 21434 and ASPICE for CS compliance. International Journal of Safety & Security Engineering, 15(6): 1123-1137. https://doi.org/10.18280/ijsse.150604

[6] Oberti, F., Savino, A., Sanchez, E., Casasso, P., Parisi, F., Di Carlo, S. (2024). CAN-MM: Multiplexed message authentication code for Controller Area Network message authentication in road vehicles. IEEE Transactions on Vehicular Technology, 73(10): 14661-14673. https://doi.org/10.1109/TVT.2024.3402986

[7] Martins, G., Moondra, A., Dubey, A., Bhattacharjee, A., Koutsoukos, X.D. (2016). Computation and communication evaluation of an authentication mechanism for time-triggered networked control systems. Sensors, 16(8): 1166. https://doi.org/10.3390/s16081166

[8] Oberti, F., Savino, A., Sanchez, E., Parisi, F., Di Carlo, S. (2022). EXT-TAURUM P2T: An extended secure CAN-FD architecture for road vehicles. IEEE Transactions on Device and Materials Reliability, 22(2): 98-110. https://doi.org/10.1109/TDMR.2022.3157000

[9] Labrado, C., Thapliyal, H., Mohanty, S.P. (2021). Fortifying vehicular security through low overhead physically unclonable functions. ACM Journal on Emerging Technologies in Computing Systems, 18(1): 1-18. https://doi.org/10.1145/3442443

[10] Lotto, A., Marchiori, F., Brighente, A., Conti, M. (2024). A survey and comparative analysis of security properties of CAN authentication protocols. IEEE Communications Surveys & Tutorials, 27(4): 2470-2504. https://doi.org/10.1109/COMST.2024.3486367

[11] Buscemi, A., Turcanu, I., Castignani, G., Panchenko, A., Engel, T., Shin, K.G. (2023). A survey on Controller Area Network reverse engineering. IEEE Communications Surveys & Tutorials, 25(3): 1445-1481. https://doi.org/10.1109/COMST.2023.3264928

[12] Lin, C.W., Sangiovanni-Vincentelli, A. (2012). Cyber-security for the Controller Area Network (CAN) communication protocol. In 2012 International Conference on Cyber Security, Alexandria, VA, USA, pp. 1-7. https://doi.org/10.1109/CyberSecurity.2012.7

[13] Adly, S., Moro, A., Hammad, S., Maged, S.A. (2023). Prevention of Controller Area Network (CAN) attacks on electric autonomous vehicles. Applied Sciences, 13(16): 9374. https://doi.org/10.3390/app13169374

[14] Groza, B., Murvay, P.S. (2019). Identity-based key exchange on in-vehicle networks: CAN-FD & FlexRay. Sensors, 19(22): 4919. https://doi.org/10.3390/s19224919

[15] Schmittner, C. (2022). Automotive cybersecurity auditing and assessment-presenting the ISO pas 5112. In European Conference on Software Process Improvement, pp. 521-529. https://doi.org/10.1007/978-3-031-15559-8_37

[16] Sanguino, T.D.J.M., Domínguez, J.M.L., de Carvalho Baptista, P. (2020). Cybersecurity certification and auditing of automotive industry. Advances in Transport Policy and Planning, 5: 95-124. https://doi.org/10.1016/bs.atpp.2020.01.002

[17] Feng, Y., Wang, W., Weng, Y., Zhang, H. (2017). A replay-attack resistant authentication scheme for the Internet of Things. In 2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), Guangzhou, China. pp. 541-547. https://doi.org/10.1109/CSE-EUC.2017.101

[18] Lawrence, T., Li, F., Ali, I., Haruna, C.R., Kpiebaareh, M.Y., Christopher, T. (2022). A computationally efficient HMAC-based authentication scheme for network coding. Telecommunication Systems, 79(1): 47-69. https://doi.org/10.1007/s11235-021-00842-6

[19] Ikumapayi, O., Olufowobi, H., Daily, J., Hu, T., Bertolotti, I.C., Bloom, G. (2023). CANASTA: Controller Area Network authentication schedulability timing analysis. IEEE Transactions on Vehicular Technology, 72(8): 10024-10036. https://doi.org/10.1109/TVT.2023.3258746

[20] Zelle, D., Gürgens, S. (2021). BusCount: A provable replay protection solution for automotive CAN networks. Security and Communication Networks, 2021(1): 9951777. https://doi.org/10.1155/2021/9951777

[21] Luykx, A., Mennink, B., Neves, S. (2016). Security analysis of BLAKE2's modes of operation. IACR Transactions on Symmetric Cryptology, 2016(1): 158-176. https://doi.org/10.13154/tosc.v2016.i1.158-176

[22] Atiwa, S., Dawji, Y., Refaey, A., Magierowski, S. (2020). Accelerated hardware implementation of BLAKE2 cryptographic hash for blockchain. In 2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), London, ON, Canada, pp. 1-6. https://doi.org/10.1109/CCECE47787.2020.9255709

[23] Suhaili, S., Julai, N., Sapawi, R., Rajaee, N. (2024). Towards maximising hardware resources and design efficiency via high-speed implementation of HMAC based on SHA-256 design. Pertanika Journal of Science & Technology, 32(1): 31-44. https://doi.org/10.47836/pjst.32.1.02

[24] Naidu, N.B., Lakshmeewari, G. (2025). Key node authentication model using asymmetric cryptography for smart cities. International Journal of Safety & Security Engineering, 15(7): 1415-1426. https://doi.org/10.18280/ijsse.150709

[25] Debnath, S., Chattopadhyay, A., Dutta, S. (2017). Brief review on journey of secured hash algorithms. In 2017 4th International Conference on Opto-Electronics and Applied Optics (Optronix), Kolkata, India, pp. 1-5. https://doi.org/10.1109/OPTRONIX.2017.8349971