Beyond Traditional Message Authentication Codes: Future Solutions for Efficient Authentication of Message Streams in IoT Networks

Gaurang Bansal, Biplab Sikdar, *Senior Member, IEEE*

***Abstract*—One of the major issues of secure communication in resource-constrained contexts is the overhead of integrity protection. A message authentication code (MAC) is a few-byte block used to authenticate a message. The receiver can check this block to ensure that a third party has not tampered with the message. A Message Authentication Code (MAC) is a tag attached to a message to ensure the message's integrity and authenticity. It is created by applying a MAC algorithm to a message in conjunction with a secret key. Traditional Message Authentication Code (MAC) schemes' tags (e.g., HMAC) are generally quite long and thus consume the majority of the total payload in IoT networks. This paper proposes a new MAC approach that uses state chaining, random access messaging, and tag truncation for improved performance and security levels to address the issue. The proposed MAC protocol improves on existing MAC solutions such as CuMACs, Mini-MACs, and ProMACs.**

***Index Terms*—MACs, IoT, Authentication, Security**

I. Introduction

Most internet protocols rely heavily on Message Authentication Code (MAC) techniques [1, 2]. They assure the message's legitimacy between two or more participants in the transaction. Despite their importance, MAC algorithms are often not properly understood in the design of cryptosystems. For instance, while designing a new protocol, a common mistake is to concentrate entirely on the communication's privacy and ignore the repercussions of a message alteration (whether by transmission error or malicious attacker). One may argue that if the message is properly encrypted, then the contents of messages are hidden, so what is needed for message authentication. But this assumption has a logical flaw. In general, an attacker can acquire a decent notion of the approximate content of the ongoing communication, and this information is more than enough to tamper with the message meaningfully. Consider a basic banking protocol as an example. One may transmit a transaction to the bank for permission, and the bank responds with a single bit: 0 for denied transactions and 1 for successful transactions. If the transmission is not authorized and the attacker can modify messages on the communication line, it may create several problems. For example, the attacker could transmit false credentials to the merchant, which the bank would rightfully reject. Still, because the attacker already knows the message will be rejected, he could convert the encrypted zero the bank sends back to a one simply by flipping the bit value. Thus message integrity is very important. These are the sorts of attacks that MACs are meant to prevent. MAC algorithms operate in the same way as symmetric ciphers do. They are pre-programmed algorithms that accept a secret key to govern the mapping from input to output (typically called the tag).

Gaurang Bansal and Biplab Sikdar are with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 119077, Singapore (email: e0622339@u.nus.edu, bsikdar@u.nus.edu)

A. Purpose

The purpose of a MAC is to guarantee that two (or more) parties that share a secret key may communicate while being able to detect changes to the message in transit. As previously mentioned, this prevents an attacker from changing the message to achieve negative results [3]

B. Working

MAC algorithms do this by taking the message and secret key as input and creating a fixed-size MAC tag. The message and tag are sent to the other party, who may then recalculate the tag and compare it to the tag sent. If they are identical, the message is nearly probably accurate. Otherwise, the message is inaccurate and should be disregarded, or the connection should be terminated, depending on the circumstances.

To authenticate a message m, the sender uses the tag generation algorithm that generates the corresponding tag t. The verification algorithm enables the recipient to evaluate whether the received tag is valid upon receiving a message. This verification is done by computing the tag for the received message m and comparing it to t. A MAC scheme is considered secure if it is computationally infeasible to generate a (m', t) pair that the receiver would accept without knowing the m. This requirement can, e.g., be achieved by using keyed hash functions such as HMAC-SHA256 [4].

An attacker would need to disable the MAC function to counterfeit a message. This is certainly not a simple task. It will be equally as difficult as cracking the encryption that safeguards the message's confidentiality.

Drawbacks

With low latency requirements and restrictions on packet size or energy usage, traditional MACs have too much overhead in these situations. The length of the tag is fixed (e.g., 64 bits, 128 bits, 160 bits, etc.), which may not fit the actual application environment and need to be transformed into a smaller size.

Since the traditional MACs are not suitable for IoT networks, the next sections provide various approaches to mitigate

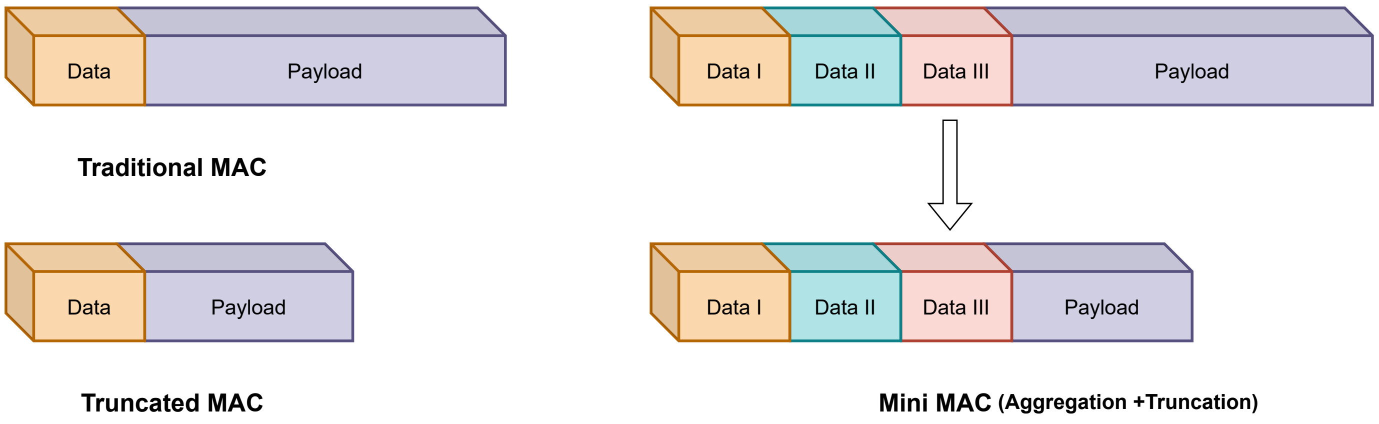

Figure 1. Truncated MACs and Mini-MACs. Truncated MACs reduce the tag length by truncation. In Mini-MACs, truncation is preceded by data aggregation.

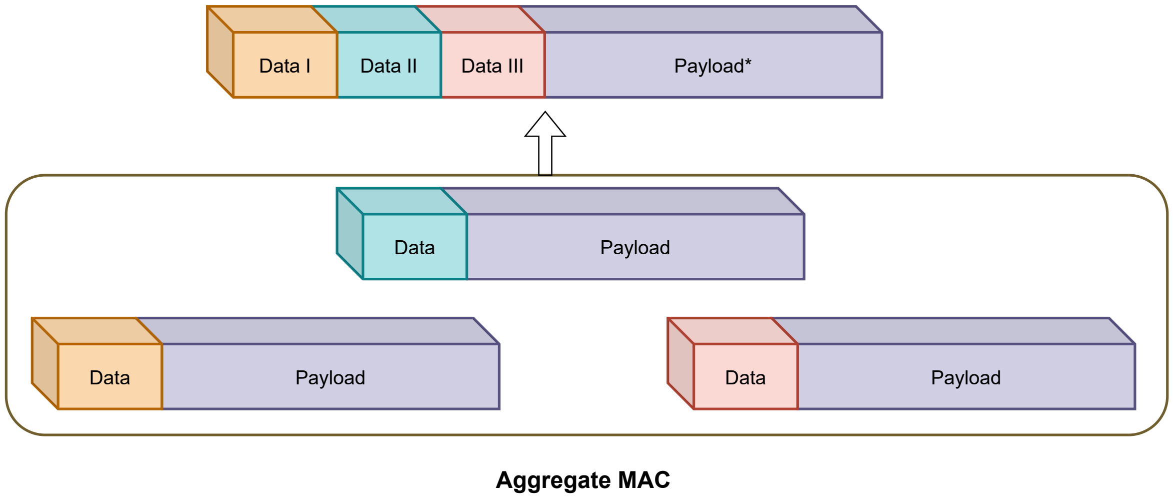

Figure 2. Aggregated MACs

the large overhead and provide efficient MAC evaluation. Section II discusses truncated and mini MACs, and section III presents aggregated MACs. While cumulative MACs and progressive MACs are described in section IV. Finally, we propose our novel approach, ”RAM-MAC,” in section V and discuss its advantages over traditional MAC in section VI. Future works and conclusions are summarised in sections VII and VIII, respectively.

II. Truncated and Mini MACs

Message Authentication Codes (MACs) have traditionally been used to assist in authenticating the sender of a message. However, due to their limited payload sizes, typical MACs are inappropriate for usage on IoT networks such as wireless sensor networks and vehicular Adhoc networks. Furthermore, the necessity not to delay messages or increase bus traffic that goes into HMAC calculation significantly limits how effectively the traditional scheme can secure the network. Two techniques may be used to address this challenge:

A. Truncated MACs

Short-term protection of packets may be all that is needed to fix the problem. The security assurances offered by Truncated MACs [5] reduce the MAC overhead by truncating the MAC tag (shown in figure 1). Truncation is a common technique of transformation that has been specified in all of the MAC standard publications, including FIPS-133, FIPS-198, ISO/IEC 9797-1, ISO/IEC 9797-2, RFC 2104, and NIST 800-38B [5]. The truncation is the same in all of these files: truncate the n-bit tag such that the leftmost $t(1 \leq t \leq n)$ bits remain.

B. Mini-MACS

Malicious attacks using bus access are possible on the Controller Area Network (CAN) bus. Mini-MAC ([6]) is built on a counter-seeded keyed-Hash MAC (HMAC) that has been enhanced with message history and trimmed to fit available message space. It does not affect bus traffic and has a negligible performance disadvantage compared to the provably secure HMAC. Mini-MAC reduces MAC bandwidth overhead by adjusting the tag length based on the available space. It first aggregates different packets in the CAN bus and then truncates the MAC (Figure 1).

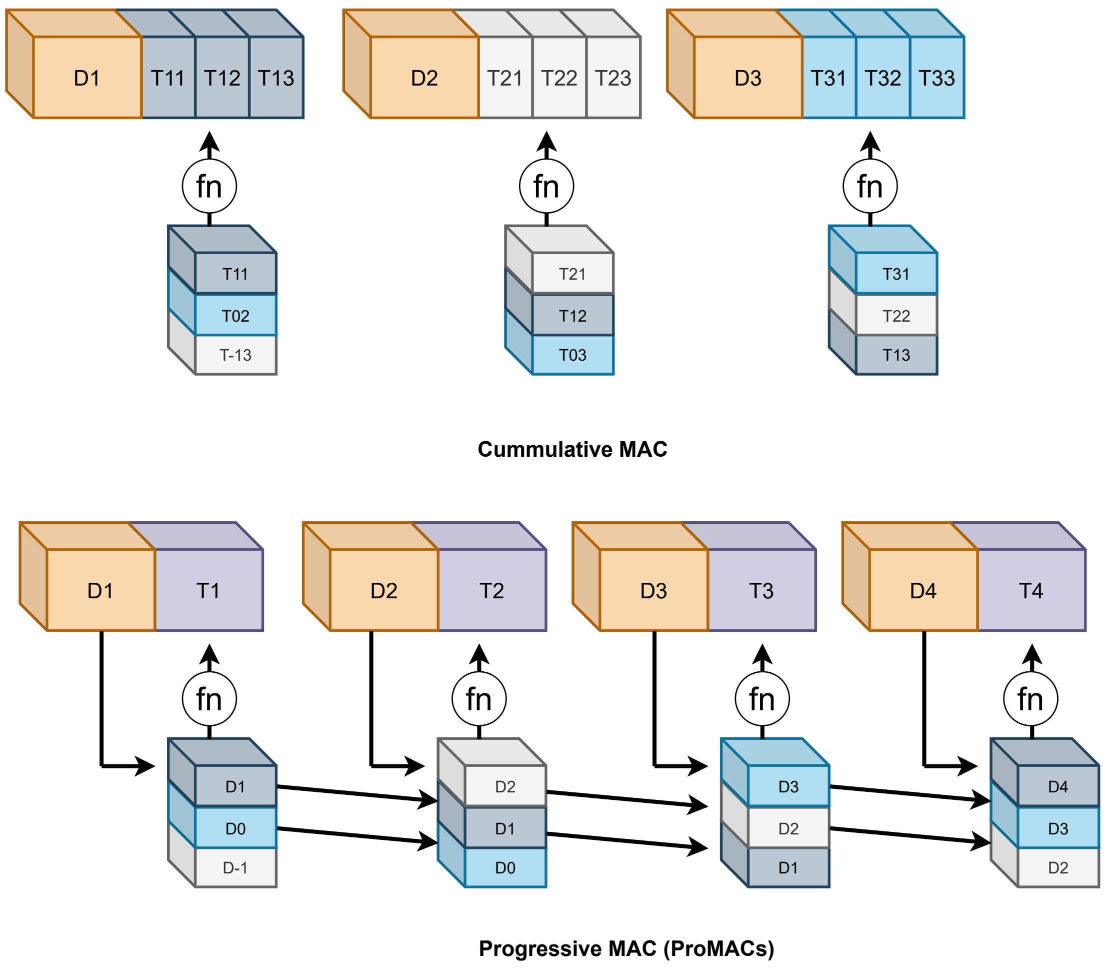

Figure 3. The core idea of CuMACs and ProMACs is to enrich each message with small authentication tag that are aggregated at receiver end to achieve high security levels. Both CuMAC and ProMAC promise tag size reductions without compromising on security by distributing and aggregating tags across multiple messages.

Drawback

The bandwidth in truncated MAC and Mini-MAC is reduced by reducing the bit size of MAC tags. Performance increases linearly as the number of bits sent decreases. However, since only the tag bits broadcast are accessible for verifying the packet's integrity, this leads to an exponential security loss. This may jeopardize the safety and security of underlying systems and is therefore not always acceptable.

When the tag is shortened, the security of the truncated MAC may be lost abruptly. For example, let $H(X)$ be the function employed in building variable-input-length MACs from fixed-input-length MACs. The underlying component $H(X)$ is assumed to be an unexpected function (UPF), and the proof is that the MAC built on it is likewise an unpredictable function. Is it still unexpected if the tag is truncated? The answer is "no." As a result, the security of truncated MACs and Mini-MACs cannot be guaranteed.

III. Aggregated MACs

A. Aggregate MACs

J. Katz and A. Lindell proposed aggregated MACs [7] (shown in figure 2), where each packet's tag is calculated separately and then merged into a single value. It is possible to create an aggregate MAC scheme in which the tags are calculated individually by each sender and then concatenated into an aggregate tag. There are no secret keys involved; therefore, anybody may do the aggregation, even if they are not linked. For sequential aggregate systems, it is usual to validate the authenticity of an aggregate received thus far before calculating the new aggregate. The aggregation method takes earlier messages into account while doing calculations. In contrast, the non-sequential aggregation process is independent of prior communications.

There is yet another line of study on compressing MAC tags known as sequential aggregate MACs. We may examine the validity of many messages (as in aggregate MACs) and the (sequential) sequence of messages in sequential aggregate MACs. This trait is necessary for applications such as IoT networks and mobile ad-hoc networks, which use networks of resource-constrained devices (MANET).

Drawback

Aggregated MACs employ tag aggregation of packet integrity checks to yield a high-performance gain. However,

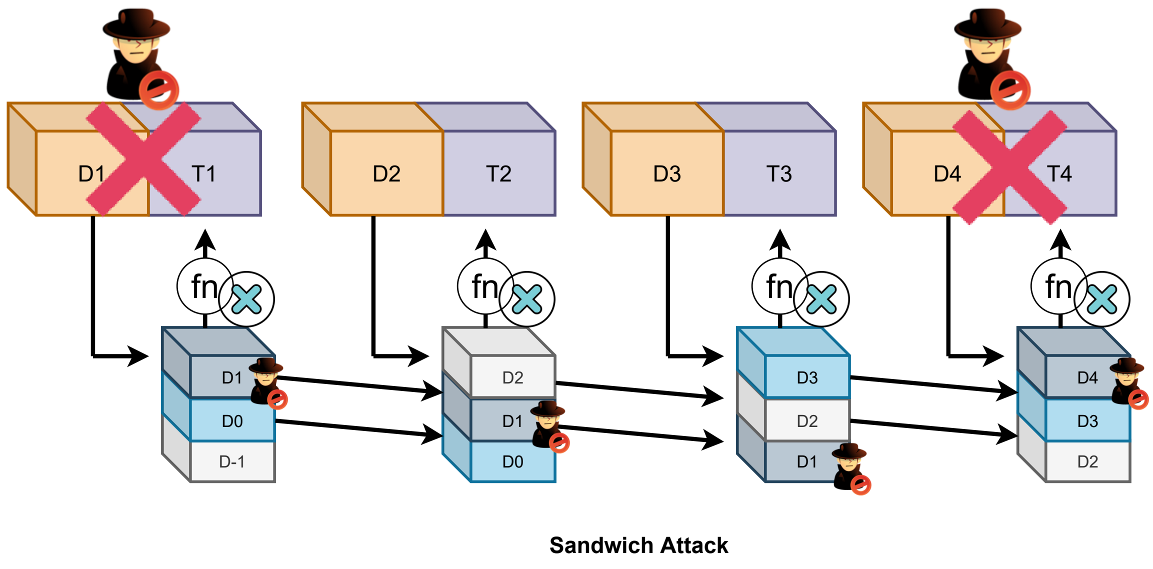

Figure 4. Sandwich attack on ProMAC. A similar attack can be formulated for CuMAC also.

losing a single packet leads to the invalidation of the entire message sequence. There is no possibility of resynchronizing the message once an error occurs.

IV. Cumulative MACs

Cumulative MACs or CuMAC proposed in [8, 9] is an alternative to typical MACs that assures low communication overhead so that the MAC may fit in a message packet while still guaranteeing that the cryptographic strength matches the application's security needs. During the first phase of tag construction, the sender produces segments and saves the message's MAC in a segment array (Figure 3). The second phase consists of getting various MAC segments from previously delivered messages and several MAC segments from the current message and combining them to form a tag. The recipient is responsible for doing tag verification after receiving each shipment. In the first stage, the receiver constructs an authentication tag for the received message using the same technique as the tag generation algorithm. In the second step, the receiver compares the created authentication tag to the received one. If the two authentication tags match, the MAC segment of a message is compared against MAC segments previously received to assess its legality.

A. CuMAC with Speculation (CuMAC/S)

CuMAC with Speculation (CuMAC/S) was an extension of CuMACs proposed in [8, 9]. This enables the sender and receiver to predict future message values, pre-compute corresponding MAC segments, and perform aggregation and accumulation. CuMAC/S significantly reduces the MAC verification latency for accurately anticipated communications without sacrificing cryptographic strength.

B. Progressive MACs (ProMACs)

Frederik Armknecht et al., proposed a novel framework ProMACs [10] (shown in figure 3), that extend the procedure of traditional MACs by reducing the tag size, and giving a set of recent historical messages instead as input $(m_1, \ldots, m_n)$. The generation and verification of authenticity tags are then based on multiple messages. The tag t_{i+2}, e.g., is computed from messages m_{i+2}, m_{i+1}, and m_i. Likewise, the integrity of message m_{i+2} is protected by tags $t_{i+2}, t_{i+3}, t_{i+4}$. Since each tag aggregates partial integrity protection for multiple messages, progressive integrity protection results in shorter tags.

Drawbacks

Both Progressive MACs and Cumulative MACs improved performance from traditional MACs, providing improved security, low bandwidth overhead, and fast packet authentication. However, both suffer from attacks by malicious adversaries. This framework did not consider an attacker that benefits from transmission failures, either caused by a lossy channel or by active interference, in their (formal) security proofs, thus making their framework susceptible to attack. We show that an attacker can drop or alter packets and can deliberately remove integrity protection from a complete sequence of messages by interfering with only two carefully chosen packets. If two transmission failures are less than the tracked message history apart, all messages "sandwiched" between these transmissions cannot be authenticated by current ProMACs, as shown in figure 4. For example, with a ProMAC spread over four messages, an attacker can remove integrity protection from 3 consecutive messages by selectively dropping only the two messages proceeding and succeeding in this sequence. Since most communication applications follow the packet circuit model, most packets that arrive at the receiving device arrive out of order. The model does not propose any technique for ordering the packets. Hence, the out-of-order packets must be dropped for the proposed algorithm's execution. This leads to the requirement of many retransmissions from the sender and increases delay. By spreading a message's integrity protection

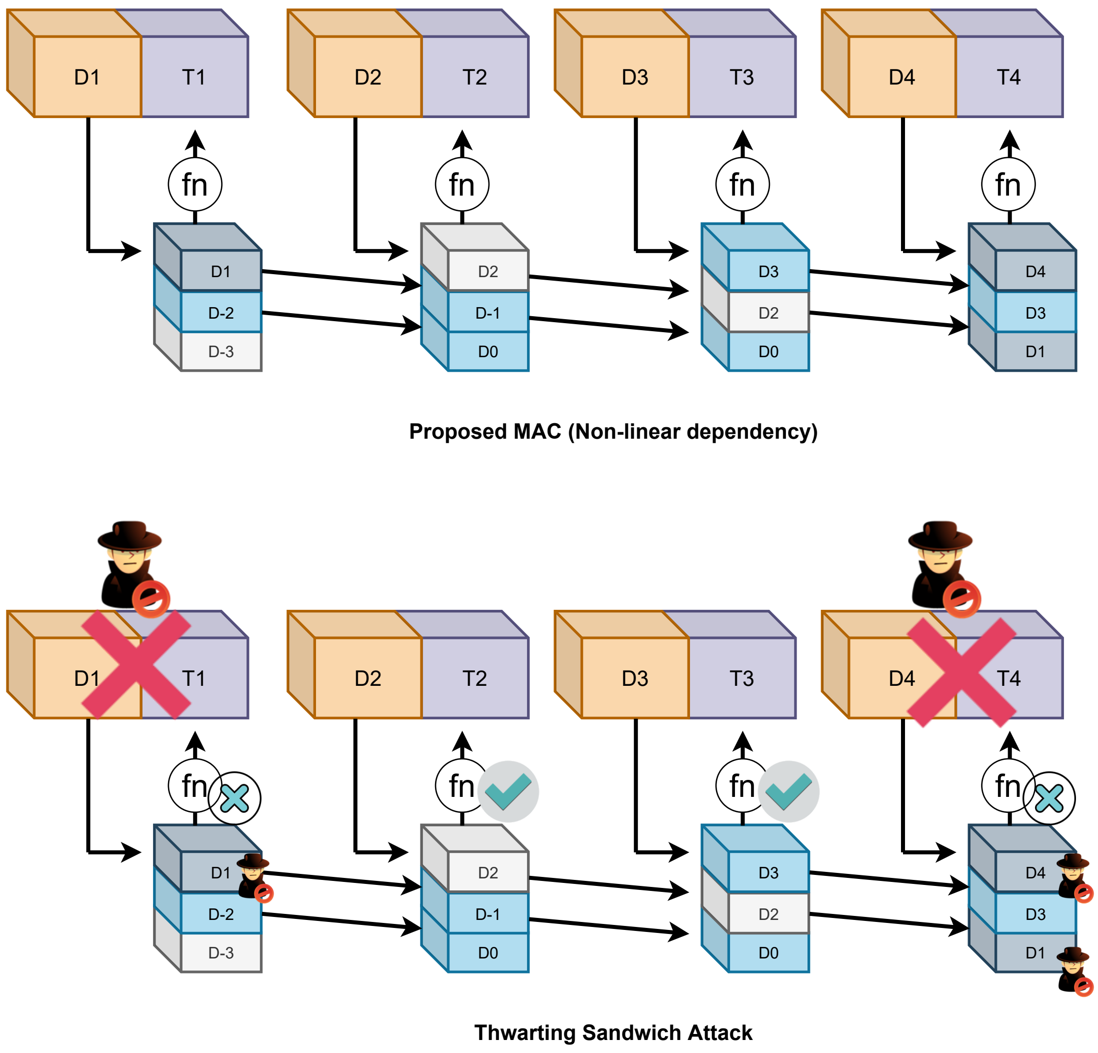
D1
T1
D2
T2
D3
T3
D4
T4
fn
D1
D-2
D-3
D2
D-1
D0
D3
D2
D0
D4
D3
D1
Proposed MAC (Non-linear dependency)
Thwarting Sandwich Attack

Figure 5. Proposed MAC and thwarting sandwich attack.

over consecutive messages, these schemes are susceptible to network-level attacks, where attackers can selectively invalidate packets to inject arbitrary traffic into a data stream.

V. Proposed MAC: RAM-MAC

Since the root cause of this problem is that all the previous MACs, so far, do not consider the collateral damage of packet loss, i.e., the impact of lost packets on the verifiability of surrounding tags. So we instate our MAC, Random Access Message MAC (RAM-MAC), which removes the linear dependency among the packets (as shown in figure 5). Like ProMACS, RAM-MACs extend the procedure of traditional MACs by reducing the tag size and giving a set of recent historical messages instead as input $(m_1, \ldots, m_n)$. However, the dependency is not based on the sliding window. Instead, among the packets in a buffer. For instance, consider that the sender's buffer size is 16 packets. In ProMACs, the dependency of the third packet is on the first and second packets. However, in RAM-MAC, the dependency is generated by a pseudo-random generator function. So third packet might depend on the first and seventh packets.

We ensure synchronization of dependency among packets on both sides of communication using the following algorithm. The algorithm computes the dependency for each message in a buffer space. The buffer size denotes the maximum number of messages stored at a particular instant in the memory of the sender and receiver. We assume the buffer space for both sender and receiver is the same. Both sender and receiver share the same seed value initially. Using the seed value input, the dependency among packets is generated by the algorithm. The dependency is reciprocal. I say, message m_i depends upon m_{i-3} and m_{i+2} besides depending on itself. Then messages m_{i-3} and m_{i+2} will in turn depend on message m_i for their authentication. In each iteration, the seed to the pseudo-random generator function is changed, such that every time there is a new dependency set. The algorithm ensures that each packet in the buffer is dependent on an equal number of packets. This is called the dependency count of the packet. For example, if the buffer size is 16, we can have each packet dependent on 3 other packets. RAM-MAC provides a fixed security level of 128 bit with a constant memory overhead per message stream.

The nonlinear dependency among the packets can help mitigate sandwich attacks, as shown in figure 5. For example, an adversary takes control of the first and fourth packets. Still, packets two and three can be successfully verified, which was impossible in ProMACs or CuMACs due to linear dependency.

VI. Proposed MAC vs Traditional MACs

This work proposes a crucial contribution to addressing the difficulty of low performance in integrity checks on streamed messages. The significant contributions of the paper are listed as follows:

- **Generic Construction**: The proposed MAC extend the notion of standard MACs. Any MAC scheme (traditional [11], truncated [12], aggregated [13] and stateful MACs [14]) can be seen as an instantiation of proposed RAM-MAC by changing message dependency and tag length.
- **Fast packet authentication**: Proposed MAC ensures direct authentication of each packet immediately upon (or at most shortly after) its reception, similar to conventional MACs [4]. The receiver verifies the tag using packets received and internal state. Since the tags are smaller, the authentication is faster than traditional MACs.
- **Low bandwidth overhead**: The traditional MAC tags have low goodput, i.e., the majority of bandwidth is consumed by integrity protection overhead. Even the tiniest messages of only a few bytes require a tag of several bytes (e.g., 16 bytes for 128-bit security). Proposed MAC improves goodput performance by reducing the tag size without affecting the security levels. Similar to the previous case, spreading the proposed MAC over 8 subsequent messages, e.g., allows us to reduce the size of a tag from 16 bytes to 2 bytes and still achieve a security level of 128 bits.
- **High-security guarantee**: Although the truncated MAC satisfies the first two requirements of fast packet authentication and low bandwidth overhead, it comes at the cost of reduced security level or intolerable transmission delays. On the other hand, the proposed MAC provides reduced integrity protection, progressively reinforced by subsequent messages, eventually achieving ”full” security. Spreading a proposed MAC over 8 subsequent messages, e.g., allows one to reduce the size of a tag from 16 bytes to 2 bytes and still achieve a security level of 128 bits.

VII. Future scope

The suggested method’s future scope is fairly broad since several enhancements may be done. The suggested technique may be tweaked in various ways, such as by varying message length, using time stamping, and so on. The approach may also be used in medical information, security, military applications, and a variety of peer-to-peer applications. Because this study gives a theoretical examination of the model, a real application scenario might highlight the approach’s advantages and disadvantages. The work may be improved further by considering the type of the packet, the length of the header, the source node, and the destination node.

VIII. Conclusion

The overhead of integrity protection is one of the significant challenges of secure communication in resource-constrained environments. Message authentication codes have been used to ensure message integrity during the communication. However, the traditional message authentication code (MAC) schemes incur a high overhead and thus cause latency. The impact of MAC becomes more prominent in IoT networks where the HMAC takes up the majority of the total payload. Over the years, many researchers have worked on designing novel MAC solutions such as CuMACs, Mini-MACs, and ProMACs that improve the efficiency of MACs, but each has its limitation. This work proposed a novel MAC technique (RAM-MAC) that addresses the issues of previous MACs while yet maintaining efficiency. RAM-MAC employs state chaining and tag truncation for improved performance and security.

IX. Acknowledgement

This work was supported by the Ministry of Education, Singapore under grant T2EP20121-0036.

References

[1] G. Bansal and B. Sikdar, “S-maps: Scalable mutual authentication protocol for dynamic uav swarms,” *IEEE Transactions on Vehicular Technology*, vol. 70, no. 11, pp. 12 088–12 100, 2021.

[2] G. Bansal and V. Chamola, “Lightweight authentication protocol for inter base station communication in heterogeneous networks,” in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2020, pp. 871–876.

[3] T. St Denis, *Cryptography for developers*. Elsevier, 2006.

[4] M. Bellare, R. Canetti, and H. Krawczyk, “Keying hash functions for message authentication,” in *Annual international cryptology conference*. Springer, 1996, pp. 1–15.

[5] P. Wang, D. Feng, C. Lin, and W. Wu, “Security of truncated macs,” in *International Conference on Information Security and Cryptology*. Springer, 2008, pp. 96–114.

[6] J. Schmandt, A. T. Sherman, and N. Banerjee, “Mini-mac: Raising the bar for vehicular security with a lightweight message authentication protocol,” *Vehicular Communications*, vol. 9, pp. 188–196, 2017.

[7] J. Katz and A. Y. Lindell, “Aggregate message authentication codes,” in *Cryptographers’ Track at the RSA Conference*. Springer, 2008, pp. 155–169.

[8] H. Li, V. Kumar, J.-M. J. Park, and Y. Yang, “Cumulative message authentication codes for resource-constrained networks,” in *2020 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2020, pp. 1–9.

[9] H. Li, V. Kumar, J.-M. Park, and Y. Yang, “Cumulative message authentication codes for resource-constrained iot networks,” *IEEE Internet of Things Journal*, 2021.

[10] F. Armknecht, P. Walther, G. Tsudik, M. Beck, and T. Strufe, “Promacs: progressive and resynchronizing macs for continuous efficient authentication of message streams,” in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 211–223.

[11] M. Bellare, J. Kilian, and P. Rogaway, “The security of the cipher block chaining message authentication code,” *Journal of Computer and System Sciences*, vol. 61, no. 3, pp. 362–399, 2000.

[12] D. McGrew, E. Rescorla *et al.*, “Datagram transport layer security (dtls) extension to establish keys for the secure real-time transport protocol (srtp),” 2010.

[13] M. Bellare, R. Guérin, and P. Rogaway, “Xor macs: New methods for message authentication using finite pseudorandom functions,” in *Annual International Cryptology Conference*. Springer, 1995, pp. 15–28.

[14] C. Boyd, B. Hale, S. F. Mjølsnes, and D. Stebila, “From stateless to stateful: Generic authentication and authenticated encryption constructions with application to tls,” in *Cryptographers’ Track at the RSA Conference*. Springer, 2016, pp. 55–71.