블룸 필터를 이용한 다수의 메시지 인증코드의 표현

손주형, 서승우, 강유*, 최진기*, 문호건*, 이명수*
서울대학교 전기,컴퓨터 공학부
*KT연구소 정보보호단

Representation of Multiple Message Authentication Codes using Bloom Filters

Ju-Hyung Son, Seung-Woo Seo, Yu Kang*, Jin-Gi Choe*, Ho-Kun Moon*, Myuong-Soo Lee*

School of EECS, Seoul National University.

*KT Information Security Center, Korea Telecom.

요 약

Multiple *Message Authentication Codes* can be represented by one of the ***Short MAC, Bloom Filter*** or ***Compressed Bloom Filter*** to reduce communication overheads. However, this will inevitably increase *false positive rate (fpr)* which is a false authentication probability of adversarial messages in trade-off of communication efficiency. While the simple short MAC scheme has the lowest *fpr*, one cannot choose arbitrary authenticator size. Bloom filter, randomized data structure often used for membership queries, can represent multiple MACs more flexibly with slightly higher *fpr*. Furthermore, compressed Bloom filter has the same *fpr* with the short MAC while maintaining its flexibility. Through our detailed analysis, we show that pros and cons of the three schemes are scenario specific. Therefore one can choose appropriate scheme under given parameters to achieve both communication efficiency and security based on our results.

I. Introduction

MAC (Message Authentication Code) is a checksum generated by a message and a shared secret key of two communicating parties. Receiver can check integrity and authenticity of the received message by verifying the attached MAC value. In scenarios where a message is attached by several MACs such as "False report filtering"[6][7] or "Broadcast authentication"[2][5], one should consider communication overheads in addition to security. In energy constrained networks such as *sensor networks*, it is well known that energy dissipated during communication overwhelms computation overheads. In this letter, we consider efficient representation of multiple MACs with small message overhead and adjustable security level. Let's assume that sender attaches n MACs on each messages, resulting authenticator size of $n \times b$ bits. In a simple manner, we can reduce individual MAC size by sending the first, last, or randomly chosen 't' bits out of 'b' bits. By using this Short MAC scheme, sender reduces communication overhead to $n \times t$ bits while increasing *false positive rate(fpr)* from 2^{-b} to 2^{-t}. The *false positive* means that receiver falsely authenticates a certain message although its MAC value was generated by adversary. Main drawback of the short MAC scheme is its inflexibility. Each reduced MAC value should be identical

본 연구는 (주) KT와 대학정보통신연구센터 (ITRC) 프로그램의 지원으로 수행되었습니다.

in size so that receiver can distinguish each MAC values without explicit indexes. We introduce Bloom filters which can represent multiple MAC values more flexibly. While the original *Bloom Filter* [1] has a higher *fpr* than the short MAC, *Compressed Bloom Filter* [3] can reduce the *fpr* equal to that of the short MAC. We establish analysis model of compression of multiple MACs and show comparison results between these mechanisms throughout this letter.

II. Preliminaries

In this section, we briefly explain concepts and mechanisms of MAC and Bloom Filters.

2.1. Message Authentication Code (MAC)

MAC function is a public function of the message and a secret key that produces a fixed-length value which authenticates that the message is from the legitimate sender. Let's assume that sender A and receiver B shares a secret key k. For a variable-length message M to be sent from A to B, A calculates MAC and sends the M with the generated MAC. If we denote the public MAC function as C, this procedures can be described as

$$MAC = C_k(M),\ A \rightarrow B : M \,\|\, C_k(M)$$

Upon receiving the message, receiver B also generates MAC using the received M and the key k. If the received MAC value is the same with the newly generated MAC value, receiver assures that the message is from the legitimate sender A and not been tampered during transmission. Usually MAC function C is many to one function. If b bit MAC is used, total 2^b possible MAC results can be generated. MAC function can be easily implemented using cryptographic hash functions or block ciphers. It is well known that MAC function is very simple to generate and verify even in resource constrained devices. MAC security usually depends on randomness of the MAC function and its output code size 'b'.

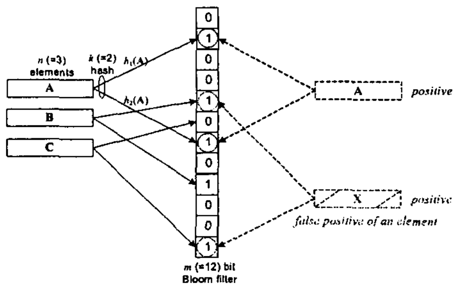

Figure 1: An example of a Bloom filter. False element "X" results from *false positive rate.*

2.2. Bloom Filter (BF)

Bloom filter is a space-efficient data structure often used to represent a group of elements with a small *false positive rate.* That is, Bloom filter saves space at the cost of representing a slightly larger group. Fig.1 shows the typical generation and verification process of Bloom filter. In this example, there are three elements represented in the Bloom filter. Their values are hashed k times using k different hash functions, and the values are mapped into a k different bit positions of an m-bit Bloom filter. Note that it is possible that a certain element's hashed values being mapped into an already marked bits. Bloom filter maintain this collision as "1" without any change. This is how Bloom filter reduces spaces. In the example depicted in Fig.1, an element X which is not a legitimate element of the group falsely succeeds in Bloom filter verification and therefore a false positive event happens.

Given a group of n elements and an m-bit Bloom filter, the probability of a certain bit being unmarked after all n elements being mapped into the Bloom filter is

$$p = (1 - 1/m)^{km} \approx e^{-kn/m}. \qquad (1)$$

Then the false positive rate of an element, which is equal to the probability that k hash results of an element all goes to bits that are already marked is:

$$fpr_{elmt} = \left(1-(1-1/m)^{kn}\right)^k \approx (1-e^{-kn/m})^k \quad (2)$$

The 'optimal k' that minimizes the above equation can be obtained by differentiation as $k_{opt} = \ln 2(m/n)$. With this k value $fpr_{elmt}^{k_{opt}}$ which is minimal value of false positive rate of an element becomes

$$fpr_{elmt}^{k_{opt}} = (0.5)^{\ln 2(m/n)} \approx (0.6185)^{(m/n)}. \quad (3)$$

2.3. Compressed Bloom Filter (CBF)

If Bloom filter should be exchanged between communicating parties, size of Bloom filter becomes important to minimize communication overheads. M. Mitzenmacher introduced *Compressed Bloom filters* [3] which reduces size of Bloom filter using arithmetic compression [4]. The design uses arbitrary large Bloom filter (m_0 bit) to represent n elements with fewer number of hash functions. Then compress the large Bloom filter using arithmetic encoding before transmission. In the original Bloom filter design, optimal k produces the p in Eq.(1), a probability that a certain bit position in Bloom filter is '0' as 1/2. Since this 'p' makes it difficult to have compression gain, CBF uses small number of hash functions and large Bloom filter size to generate long consecutive '0's for compression efficiency. On the same Bloom filter size, it can reduce false positive rate in cost of compression/decompression overhead at sender/receiver.

III. Performance Analysis

In this section, we provide formal analysis of representing multiple MACs using short MAC and two Bloom filter schemes.

3.1. System Model

In our model, there are two communicating parties; *Sender group* and *Receiver group*. The sender group and receiver group is composed of n and l nodes respectively where $(1 \le l \le n)$. We assume that both group shares n pairwise secret keys before communication. Sender group transmit a message attached with n MACs generated by the n keys and a public MAC function. These n MAC values are represented by our short MAC or Bloom filters for communication efficiency. In general, this model generalizes conventional 1-to-1 communication scenario to 1-to-many, many-to-1, and many-to-many communication scenarios.

We can think of many specific scenarios where our model holds. Here we introduce two examples. First, 'false report filtering' is one scenario. In [6][7], authors proposed a filtering scheme of a false report from a malicious or malfunctioning node in wireless sensor network. A newly generated sensor report is certified by n neighbors using their contributed MAC values generated using the report message with neighbors' own secret keys. During hop-by-hop forwarding of that report, every intermediate nodes verify validity of MAC codes using their shared keys with the n neighbors. In this case, sender group is a report-generating node plus its neighbors and receiver group is several nodes along the path to sink plus sink node itself which holds all secret keys in network. In second example, multiple MACs are also used in 'broadcast authentication'. One sender can broadcast a message M attached by n MACs for many receivers who each shares l $(l \le n)$ keys with the sender [2][5]. Each receiver verifies the message using their own l keys. In this case, sender group is one sender and receiver group is multiple receivers.

3.2. Analysis of false positive rate

3.2.1. Short MAC

In Short MAC, sender group simply shorten size of each MAC from the original 'b' bits to 't' bits ($t < b$). Since there are total n keys, total transmitted authentication information size is reduced from $n \times b$ to $n \times t$ bits. Every reduced MAC code should have the same size so that each receiver find their intended MAC value in concatenated bit strings. Therefore, total authentication

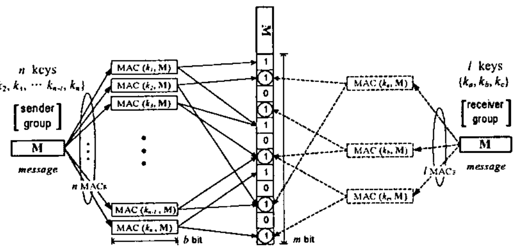

Figure 2: Bloom filter generation and verification.

information size has a granularity of n bits. In other words, reducing '1' bits in each MAC code will result in reduction of n bits in total MAC codes. This will limit flexibility of choosing authenticator sizes especially when n is large.

The *false positive rate* of *Short MAC* against blind attack is:

$$fpr_{SM} = (0.5)^{t \cdot l} = (0.5)^{\lfloor m/n \rfloor l} \qquad (4)$$

3.2.2. Bloom Filter

In Fig.2, sender group represents n MACs within a m bit Bloom filter and this is verified by receiver group using l keys. If any bit is marked as '0' during verification, the message M will not be accepted. Unfortunately, adversaries can generate a false Bloom filter which randomly marked as '1's on many bit positions. Since legitimate sender and receiver group agree that legitimate Bloom filter cannot have more than $n \times k$ '1's, attacker's best strategy is to randomly mark nk bits. Therefore, the probability that a randomly generated Bloom filter is falsely authenticated even after k hash functions of l keys is $(nk/m)^{kl}$. Therefore false positive rate of Bloom Filter, fprBF becomes:

$$fpr_{BF} = (nk/m)^{kl}. \qquad (5)$$

Like the fpr_{elmt} in Eq.(3), we can find optimal value of k which minimizes the fpr_{BF} given m and n. Differentiating Eq.(5) results $k_{opt} = \frac{1}{e}\left(\frac{m}{n}\right)$. Then with this optimal k value, fpr_{BF} has minimal value given m and n

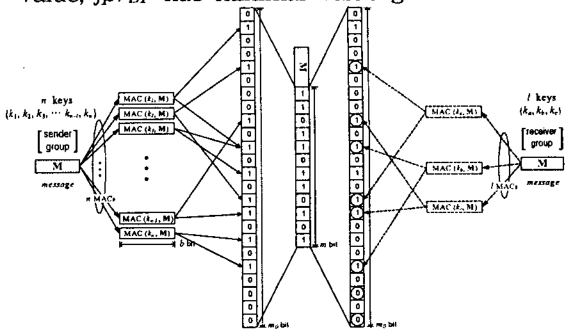

Figure 3: Compressed Bloom filter generation and verification.

as follows.

$$fpr_{BF}^{k_{opt}} = \left(\frac{n \cdot k_{opt}}{m}\right)^{k_{opt} \cdot l} \approx (0.6922)^{(\frac{m}{n})l} \qquad (6)$$

3.2.3. Compressed Bloom Filter

When we use Compressed Bloom Filter for representing multiple MACs, sender group first map n MACs into m_0 bit Bloom filter. Then they compress Bloom filter using arithmetic coding such as [4] before transmission as depicted in Fig.3. Theoretically, we can compress m_0 bits into $m = m_0 H(p)$ bits in arithmetic coding. The $H(p)$ is a binary entropy function with p as a probability that a certain bit is '0' in original bit array. If we set m first, then m_0 can be described as

$$m_0 = \frac{m}{H(p)} = \frac{m}{-p\log_2 p - (1-p)\log_2(1-p)} \qquad (7)$$

where $p = (1 - 1/m_0)^{kn} \approx e^{-nk/m_0}$ using Eq.(1). If we replace k as k_{opt} which minimizes false positive rate in Bloom filter before compression, the p would be $p = e^{-1/e} \approx 0.6922$. Then $m_0 = (e \ln 2) \cdot m \approx (1.8842) \cdot m$, it means that optimal uncompressed BF size should be 1.8842 times larger than actually transmitting Bloom filter size in order to minimize false positive. Finally, *false positive rate* of *Compressed Bloom Filter* is

$$\left.\begin{aligned} fpr_{CBF}^{k_{opt}} &= (n \cdot k_{opt}/m_0) \approx (0.6922)^{(m_0/n)l} \\ &= (0.5)^{(m/n)l} \end{aligned}\right\} \quad (8)$$

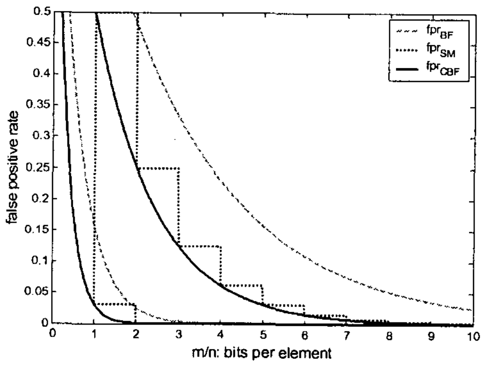

Figure 4: *false positive rate* of Short MAC(SM), Bloom Filter(BF) and Compressed Bloom Filter(CBF) with n=100 and l=1 (upper 3 graphs), l=5 (lower 3 graphs).

Note that Eq.(8) is the same as Eq.(4) except that there is no restrictions that bits per element (m/n) should be integers unlike short MAC.

3.3. Comparisons between Short MAC and Bloom Filters

Fig.4 compares false positive rate of Short MAC, Bloom Filter and Compressed Bloom Filter when there are total 100 keys at sender group while receiver group verifies the message using 1 or 5 keys out of total key pool. We can see that CBF achieves the smallest *fpr* in most settings where differences getting smaller by increasing l and m/n. In Fig.4, by increasing l from 1 to 5, *fpr* of three schemes notably decrease altogether. When l and m/n, which are number of keys for receiver group and number of bits per one MAC respectively, false positive rate of the three schemes become quite similar with each other. Therefore, when we afford enough l and m/n, we might achieve both flexibility and security by using normal BF without compression overhead.

IV. Conclusions

If we want to compress multiple Message Authentication Codes with small error rates, we can use Short MAC (SM), Bloom Filter (BF) or Compressed Bloom Filter (CBF). While *false positive rate* of SM is smaller than BF, BF's representation is more flexible than SM. We show that *fpr* of CBF is the same with SM whereas it needs compression (decompression) overheads. In conclusion, one can choose freely one of the three mentioned schemes according to their system requirements to compress multiple Message Authentication Codes.

[References]

[1] B. Bloom, "Space/time trade-offs in hash coding with allowable erros," Communications of the ACM, vol. 13, no. 7, 1970.

[2] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and Pinkas, "Multicast security: A taxonomy and some efficient constructions," in IEEE INFOCOM, March 1999.

[3] M. Mitzenmacher, "Compressed bloom filters," IEEE/ACM Transactions on Networking, vol. 10, no. 5, pp. 604 - 612, Oct. 2002.

[4] A. Moffat, R. M. Neal, and I. H. Witten, "Arithmetic coding revisited," ACM Transactions on Information Systems, vol. 16, no. 3, pp. 1237 - 1239, 1998.

[5] J.-H. Son, H. Luo, and S.-W. Seo, "Authenticated flooding in large-scale sensor networks." in IEEE MASS, November, 2005.

[6] F. Ye, H. Luo, S. Lu, and L. Zhang, "Statistical en-route filtering of injected false data in sensor networks," in IEEE INFOCOM, March 2004.

[7] S. Zhu, S. Setia, S. Jajodia, and P. Ning, "An interleaved hop-by-hop authentication scheme for filtering false data injection in sensor networks," in IEEE Security and Privacy, May 2004.