JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 26, 1539-1552 (2010)

Short Paper

A New Variant of 3GPP-MAC With Provable Security and Higher Efficiency*

LI-TING ZHANG[1,2], WEN-LING WU[1] AND PENG WANG[2]
[1]*State Key Laboratory of Information Security*
Institute of Software
Chinese Academy of Sciences
Beijing, 100190 P.R. China
[2]*Graduate University of Chinese Academy of Sciences*
Beijing, 100049 P.R. China

3GPP-MAC, also called $f9$, is an integrity scheme adopted by Universal Mobile Telecommunication System as a standard for the 3rd generation wireless communications. In this paper, we investigate the feasibility of improving its efficiency, with the precondition that its security would not be compromised. As we show, the newly proposed $f9^-$ enjoys provable security under the solo PRP assumption; more importantly, $f9^-$ removes the last block-cipher invocation of $f9$, which means it needs less time and resources to compute MACs than $f9$, offering higher efficiency in practice. Furthermore, $f9^-$ keeps the main structure of $f9$ almost unchanged, making it easy to update $f9$ to $f9^-$ in reality.

Keywords: MAC, 3GPP, integrity, provable security, efficiency

1. INTRODUCTION

MAC: MAC is the abbreviation for Message Authentication Code, which is commonly used to provide data integrity and data origin authentication, *e.g.* in banking applications [1]. The sender and receiver of messages should agree on a secret key K first; to send a message M, the sender computes $T = MAC_K(M)$; then, he/she sends (M, T) to the receiver. On receipt of them, the receiver re-computes $MAC_K(M)$ to verify if the received T' is corresponding to M. If so, M is deemed to be valid.

3GPP-MAC: The 3rd Generation Partnership Project (3GPP) is the body standardizing the next generation of mobile telephony. In the security architecture of 3GPP system there is a standardized MAC algorithm called 3GPP-MAC, with another name $f9$ in the series of algorithms specified by 3GPP. $f9$ is a MAC algorithm based on block ciphers as illustrated in Fig. 1, where COUNT, FRESH, DIRECTION are special parameters, MESSAGE stands for the message to be dealt with, and $1 \| 0 \ldots 0$ denotes the padded bits. For more details, see [2].

Received July 1, 2008; revised January 6 & February 24, 2009; accepted May 3, 2009.
Communicated by Chin-Laung Lei.
* This work was supported by the National High-Tech Research and Development 863 Plan of China (No. 2007AA01Z470), the National Natural Science Foundation of China (No. 60873259 and No. 60903219), the National Grand Fundamental Research 973 Program of China (No. 2004CB318004) and the Knowledge Innovation Project of the Chinese Academy of Sciences

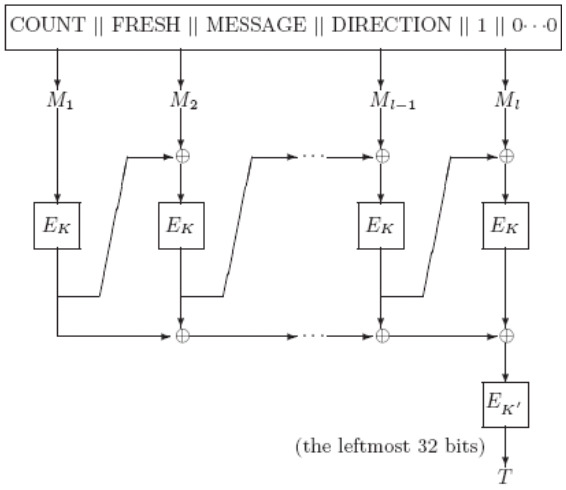

Fig. 1. Illustration of $f9$, where $K' = K \oplus KM$ and $KM = 0 \times AA\ldots AA$.

Algorithm $f9'(K, M)$
$M \leftarrow \mathbf{pad}(M)$
$M_1 M_2 \cdots M_l \leftarrow \mathbf{partition}(M)$
$Y_0 \leftarrow 0^n$
for $i \leftarrow 1$ **to** l **do**
$X_i \leftarrow M_i \oplus Y_{i-1}$
$Y_i \leftarrow E_K(X_i)$
end for
$T \leftarrow E_{K \oplus \Delta}(Y_1 \oplus Y_2 \oplus \cdots \oplus Y_l)$
return the leftmost t bits of T

Algorithm $f9^+(K, M)$
$M \leftarrow \mathbf{pad}(M)$
$M_1 M_2 \cdots M_l \leftarrow \mathbf{partition}(M)$
$Y_0 \leftarrow E_K(0^n)$
for $i \leftarrow 1$ **to** l **do**
$X_i \leftarrow M_i \oplus Y_{i-1}$
$Y_i \leftarrow E_K(X_i)$
end for
$T \leftarrow E_K(Y_1 \oplus Y_2 \oplus \cdots \oplus Y_l \oplus E_K(1^n))$
return the leftmost t bits of T

Fig. 2. Specification of $f9'$ and $f9^+$, where Δ is a non-zero key modifier.

Variants of 3GPP-MAC: Based on the structure of $f9$, researchers have given a generalized version of it, $f9'$ [3-6]. As specified in the left side of Fig. 2, $f9'$ removes COUNT, FRESH, DIRECTION in $f9$, but keeps a key modifier Δ (like the KM in $f9$), which makes it much more difficult to prove its security [3-5].

Then, $f9^+$ was introduced as an enhanced version [6]. As in the right side of Fig. 2, $f9^+$ removes the key modifier Δ from $f9'$ successfully, keeping its provable security with similar bound. What is more, $f9'$ is provably secure under the assumption that the underlying block cipher is a secure PRP-RKA (a PseudoRandom Permutation further secure against a class of Related Key Attacks, here the related key means two different keys having a known XOR-difference, *i.e.* $K_1 \oplus K_2 = \Delta$), while $f9^+$ is provably secure under the solo assumption that the underlying block cipher is a secure PRP (PseudoRandom Permutation). Therefore, the assumption $f9'$ depends on is stronger than that of $f9^+$, *i.e.* $f9'$ is weaker than $f9^+$ in practice.

Regardless of this, $f9$, $f9'$ and $f9^+$ all can be seen as authentication modes of operation for block ciphers, which means the numbers of invocation of their underlying block cipher have great influence on their efficiency. However, we notice that every time authenticating a message M, $f9$, $f9'$ and $f9^+$ all call their underlying block cipher at least

$\lceil |M|/n \rceil + 1$ times, which is 1 or 2 more times than the minimum. Considering their efficiency in practice, it stands to reason to ask, are there some ways to improve $f9$, keeping its security undamaged, to obtain higher efficiency? From an engineering point of view, we further hope the improvement on $f9$ is little, *i.e.* it would not change $f9$ too much, so as to update $f9$ naturally and easily in practice.

Motivated by the reasons given above, we focus on improving the structure of $f9$, and propose a new variant $f9^-$, as illustrated in Fig. 3. $f9^-$ removes the last block-cipher invocation and the key modifier KM in $f9$, thus it saves one block-cipher computation and one key scheduling every time authenticating a message. At an expense, a secret mask $S \cdot x$ has to be added to the last block to keep $f9^-$ provably secure. However, this additional step does not cost much computation since $S \cdot x$ can be pre-computed only once and then used for all messages. That is, the users of $f9^-$ can compute $S \cdot x$ in advance, and then store it in memory. Every time authenticating a message, the users just call for $S \cdot x$ from the memory without any further computation.

In a short, $f9^-$ makes a relatively high computation save for $f9$ at a low expense, offering higher efficiency than $f9$ and its other two variants in practice. On the other hand, $f9^-$ is provably secure for arbitrary length messages modeling its underlying block cipher as a secure PRP. This implies the security of $f9$ is not damaged after the improvement. Moreover, $f9^-$ keeps the main structure of $f9$ almost unchanged, making it easy to update $f9$ to $f9^-$ in the real world.

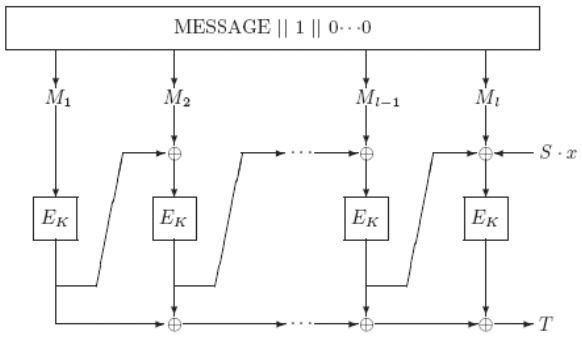

Fig. 3. Illustration of $f9^-$, where $S = E_K(\text{Cst})$, $\text{Cst} \in \{0, 1\}^n$ is an arbitrary constant, and "$\cdot$" denotes the multiplication in $\text{GF}(2^n)$.

2. PRELIMINARIES

Notations: Denote $\{0, 1\}^*$ as the set of all strings with arbitrary lengths, and $\{0, 1\}^n$ as the set of all n-bit strings. If $a, b \in \{0, 1\}^*$ are strings of equal length then $a \oplus b$ is their bitwise XOR. For strings $a, b \in \{0, 1\}^*$, $a \| b$ denotes their concatenation. However, we sometimes write ab for $a \| b$ if there is no confusion. Suppose G is a set, then $\#G$ denotes the size of G, and $x \xleftarrow{\$} G$ denotes that x is chosen from set G uniformly at random. In the rest of this paper, every time we say "random", we mean uniformly random". For an n-bit string $a = a_{n-1}\ldots a_1 a_0 \in \{0, 1\}^n$, let $a << 1 = a_{n-2}\ldots a_1 a_0 0$. For a string $M \in \{0, 1\}^*$, $|M|$ stands for its length in bits, while $\|M\|_n = \max\{1, \lceil |M|/n \rceil\}$. For any string

$M \in \{0, 1\}^*$, we let pad(M) = $M10^{n-1-|M| \bmod n}$; for any string $M \in \{0, 1\}^{ln}$, we let partition(M) = $M_1M_2 \ldots M_l$ such that $M_1 \ldots M_l = M$, and $|M_i| = n$ for $1 \le i \le l$.

The Field with 2^n Elements: GF(2^n): This is a finite field with 2^n elements, and an element a in GF(2^n) can be treated as: (1) an abstract element in the field; (2) an n-bit string $a_{n-1} \ldots a_1a_0 \in \{0, 1\}^n$; (3) a formal polynomial $a(x) = a_{n-1}\, x^{n-1} + \ldots + a_1x + a_0$ with binary coefficients. To add two elements $a, b \in$ GF(2^n), take their bitwise XOR, *i.e.* $a \oplus b$. To multiply these two elements, which we denote $a \cdot b$, regard a and b as polynomials $a(x) = a_{n-1}x^{n-1} + \ldots + a_1x + a_0$ and $b(x) = b_{n-1}x^{n-1} + \ldots + b_1x + b_0$, form their product $c(x)$ where one adds and multiplies coefficients in GF(2^n), then fix some irreducible polynomial $p(x)$ with binary coefficients and degree n, and finally take the remainder when dividing $c(x)$ by $p(x)$. See [7] for more details. It is fairly easy to multiply an element $a \in \{0, 1\}^n$ by x. We give an example for n = 128, where the irreducible polynomial $p(x) = x^{128} + x^7 + x^2 + x + 1$. Then multiplying $a = a_{n-1} \ldots a_1a_0$ by x yields $a \cdot x = a$ <<< 1, if $a_{n-1} = 0$ and $a \cdot x = (a << 1) \oplus 0^{120}10000111$ if $a_{n-1} = 1$.

3. SPECIFICATION OF $f9^-$

$f9^-$ is defined by a block cipher E: $\{0, 1\}^k \times \{0, 1\}^n \to \{0, 1\}^n$, as illustrated in Fig. 3 (with Cst $\in \{0, 1\}^n$) and described in Fig. 4 (with Cst = 0^n). The key space of $f9^-$ is the same as that of its underlying block cipher E. $f9^-$ takes a key $K \in \{0, 1\}^k$ and a message $M \in \{0, 1\}^*$, then returns $T \in \{0, 1\}^n$.

If necessary, T can be truncated to its leftmost t bits in practice; however, the security of $f9^-$ would be damaged to some extent, depending on the size of t. According to the specification of $f9$ [2], t is selected to be 32 and the block length of the underlying block cipher is 64. For simplicity, we would not consider the truncation of T in the rest of this paper.

Algorithm $f9^-(K, M)$
01. $M \leftarrow$ **pad**(M)
02. $M_1M_2 \cdots M_l \leftarrow$ **partition**(M)
03. $S \leftarrow E_K(0^n)$; $Y_0 \leftarrow 0^n$
04. **for** $i \leftarrow 1$ **to** $l - 1$ **do**
05. $\quad X_i \leftarrow M_i \oplus Y_{i-1}$
06. $\quad Y_i \leftarrow E_K(X_i)$
07. **end for**
08. $X_l \leftarrow M_l \oplus Y_{l-1} \oplus S \cdot x$
09. $Y_l \leftarrow E_K(X_l)$
10. $T \leftarrow Y_1 \oplus Y_2 \oplus \cdots \oplus Y_l$
11. **return** T

Fig. 4. Definition of $f9^-$.

4. PROVABLE SECURITY OF $f9^-$

Security Definitions: Denote Perm(n) as the set contains all the permutations over $\{0, 1\}^n$. We say P is a RP (Random Permutation) if $P \xleftarrow{\$}$ Perm(n). Normally, the security of a block cipher E: $\{0, 1\}^k \times \{0, 1\}^n \to \{0, 1\}^n$ is measured in the following

model. There exists an oracle O who selects an RP or E_K ($K \xleftarrow{\$} \{0, 1\}^k$) with equal probability and keeps it secret. Then, adversary A is allowed to query O with any plaintext it chooses, and O returns to it with corresponding ciphertext. At last, A is asked to output a bit d (0 or 1). The security of E is qualified by Adv_E^{prp} which can be seen as the probability difference between different situations (O is a RP or E_K) in which A outputs 1. To be concrete, we let

$$\mathrm{Adv}_E^{prp}(A) \triangleq |\Pr(K \xleftarrow{\$} \{0, 1\}^k: A^{E_K(\cdot)} = 1) - \Pr(P \xleftarrow{\$} \mathrm{Perm}(n): A^{P(\cdot)} = 1)|, \text{ and}$$
$$\mathrm{Adv}_E^{prp}(t, q) \triangleq \max_A \{\mathrm{Adv}_E^{prp}(A)\},$$

We say that E is secure if $\mathrm{Adv}_E^{prp}(t, q)$ is sufficiently small, where all adversaries are allowed to run in time at most t and make at most q queries.

The security of a MAC algorithm F: $\{0, 1\}^k \times \{0, 1\}^* \to \{0, 1\}^n$ (we write $F_K(\cdot)$ for $F(K,\cdot)$ in the rest of this paper) is evaluated by how unforgeable it is. That is, an adversary A can adaptively query $F_K(\cdot)$ with arbitrary messages, obtaining the corresponding outputs. Finally, A is asked to *forge*, *i.e.* to output a pair $(M, F_K(M))$ where M has never been input to $F_K(\cdot)$ by A. Then we let

$$\mathrm{Adv}_F^{mac}(A) \triangleq \Pr(K \xleftarrow{\$} \{0, 1\}^k: A^{F_K(\cdot)} \text{ forges}), \text{ and}$$
$$\mathrm{Adv}_F^{mac}(t, q, \sigma) \triangleq \max_A \{\mathrm{Adv}_F^{mac}(A)\}.$$

We say that F is secure if $\mathrm{Adv}_F^{mac}(t, q, \sigma)$ is sufficiently small, where all adversaries are allowed to run in time at most t, and make at most q queries, having aggregate length of at most σ blocks ($\sigma = \sum_{i=1}^{q} \|M^i\|_n$).

Denote Rand(*, n) as the set contains all functions from $\{0, 1\}^*$ to $\{0, 1\}^n$. We say R is a variable-length input random function if $R \xleftarrow{\$} \mathrm{Rand}(*, n)$. In other words, R behaves as follows, for any string $M \in \{0, 1\}^*$, $R(M)$ is a random string over $\{0, 1\}^n$. An adversary A is asked to distinguish $F_K(\cdot)$ ($K \xleftarrow{\$} \{0, 1\}^k$) from $R \xleftarrow{\$} \mathrm{Rand}(*, n)$. Similar to $\mathrm{Adv}_E^{prp}(t, q)$, we define the advantage of A as

$$\mathrm{Adv}_F^{viprf}(A) \triangleq |\Pr(K \xleftarrow{\$} \{0, 1\}^k: A^{F_K(\cdot)} = 1) - \Pr(R \xleftarrow{\$} \mathrm{Rand}(*, n): A^{R(\cdot)} = 1|),$$
$$\mathrm{Adv}_F^{viprf}(t, q, \sigma) \triangleq \max_A \{\mathrm{Adv}_F^{viprf}(A)\}.$$

We say that F is a *viprf* (Variable-length Input PseudoRandom Function) if $\mathrm{Adv}_F^{viprf}(t, q, \sigma)$ is sufficiently small, where all adversaries are allowed to run in time at most t, and make at most q queries, having aggregate length of at most σ blocks ($\sigma = \sum_{i=1}^{q} \|M^i\|_n$).

Without loss of generality, adversaries are assumed to never ask a query outside the domain of the oracle, and to never repeat a query.

Theorem Statements: First, we prove that $f9_p^-$ is a *viprf* if the underlying block cipher P is a random permutation (information-theoretic result).

Theorem 1 (Main theorem $f9_p^- \approx R$) Suppose $P \xleftarrow{\$} \mathrm{Perm}(n)$ is used in $f9^-$ as the underlying block cipher, and A is an adversary who asks at most q queries, having aggregate length of at most σ blocks, and assume $\sigma < 2^{n/2}$. Then

$$|\Pr(P \xleftarrow{\$} \text{Perm}(n): A^{f9_P^-(\cdot)} = 1) - \Pr(R \xleftarrow{\$} \text{Rand}(*, n): A^{R(\cdot)} = 1)| \le (\sigma^2 + \sigma)/2^n.$$

Before proving the main theorem, we first introduce a basic tool also used in [6, 9, 10]. Lemma 1 is a very useful tool to prove pseudorandomness for both block cipher structures and modes of operation.

Lemma 1 (General Lemma) Define set $M \triangleq \{(M^1, M^2, \ldots, M^q) | M^i \in \{0, 1\}^*\}$, where $M^i \neq M^j$, $1 \le i < j \le q$, if there exists a set $T = \{(T^1, T^2, \ldots, T^q)\}$, where $T = T^i \in \{0, 1\}^n$, $1 \le i \le q$ and a function family F: $\{0, 1\}^k \times \{0, 1\}^* \to \{0, 1\}^n$ such that,

(1) $\#T \ge 2^{qn}(1 - \varepsilon_1)$,
(2) $\forall (M^1, M^2, \ldots, M^q) \in M$, $\forall (T^1, T^2, \ldots, T^q) \in T$,
$\Pr(K \xleftarrow{\$} \{0, 1\}^k: T^i = F_K(M^i) \text{ for } i = 1, 2, \cdots, q) \ge (1 - \varepsilon_2)/2^{qn}$,

then for any computationally unbounded adversary A with q queries,

$$|\Pr(K \xleftarrow{\$} \{0, 1\}^k: A^{F_K(\cdot)} = 1) - \Pr(R \xleftarrow{\$} \text{Rand}(n, n): A^{R(\cdot)} = 1)| \le \varepsilon_1 + \varepsilon_2$$

For the proof of Lemma 1, refer to Appendix A.

Lemma 2 (Basic Lemma for $f9^-$) Let Rand(n, n) be the set of all functions from $\{0, 1\}^n$ to $\{0, 1\}^n$, define set M as in Lemma 1 and define $T \triangleq \{0, 1\}^{n)q}$ as the set containing all the different q-tuple elements $(T^1, T^2, \ldots, T^q)$, where $T^i \in \{0, 1\}^n$ for $i = 1, 2, \ldots, q$. Let $g \in \text{Rand}(n, n)$ be the underlying function in $f9^-$. Then the number of function g satisfying $\forall (M^1, M^2, \ldots, M^q) \in M$, $\forall (T^1, T^2, \ldots, T^q) \in T$, $T^i = f9_g^-(M^i)$, $1 \le i \le q$ is at least $N_g \ge (1 - (\sigma^2 + \sigma)/2^{n+1}) \times (2^n)^{2^n - q}$, where $\sigma = \sum_{i=1}^{q} \|M^i\|_n$.

The proof for Lemma 2 is given in Appendix B.

Based on the two lemmas given above, we can prove Theorem 1. According to Lemma 1, to prove $f9^-$ is a *viprf* we have to (1) construct a set $T = \{(T^1, T^2, \ldots, T^q)\}$ whose size is large enough (then ε_1 would be sufficiently small) and (2) show that $\forall (M^1, M^2, \ldots, M^q) \in M$, $\forall (T^1, T^2, \ldots, T^q) \in T$,

$\Pr(K \xleftarrow{\$} \{0, 1\}^k: T^i = F_K(M^i) \text{ for } i = 1, 2, \ldots, q) \ge (1 - \varepsilon_2)/2^{qn}$ holds, where ε_2 is sufficiently small. In Lemma 2, we have almost solved these two problems except that function g is not limited to permutations, and we fix this flaw in the end by the "PRP and PRF Switching Lemma" [8].

***Proof*:** (of Theorem 1) According to Lemma 2, we get

$$\Pr(g \xleftarrow{\$} \text{Rand}(n, n): T^i = f9_g^-(M^i), 1 \le i \le q) \ge \frac{(1 - (\sigma^2 + \sigma)/2^{n+1}) \times (2^n)^{2^n - q}}{(2^n)^{2^n}} = \frac{1 - (\sigma^2 + \sigma)/2^{n+1}}{2^{qn}},$$

which indicates $\varepsilon_2 = (\sigma^2 + \sigma)/2^{n+1}$ in Lemma 1.

Furthermore, applying Lemma 1, and noticing that $\#T = 2^{qn}$ which means $\varepsilon_1 = 0$, it

stands to reason that

$$\begin{aligned}&|\Pr(g \xleftarrow{\$} \text{Rand}(n, n): A^{f9_g^-(\cdot)} = 1) - \Pr(R \xleftarrow{\$} \text{Rand}(*, n): A^{R(\cdot)} = 1)| \\ &\le \varepsilon_1 + \varepsilon_2 = (\sigma^2 + \sigma)/2^{n+1}.\end{aligned} \tag{1}$$

Moreover, we claim that for any adversary A, it follows that,

$$\begin{aligned}&|\Pr(g \xleftarrow{\$} \text{Rand}(n, n): A^{f9_g^-(\cdot)} = 1) - \Pr(P \xleftarrow{\$} \text{Perm}(n): A^{f9_g^-(\cdot)} = 1)| \\ &\le |\Pr(g \xleftarrow{\$} \text{Rand}(n, n): B_A^{g(\cdot)} = 1) - \Pr(P \xleftarrow{\$} \text{Perm}(n): B_A^{P(\cdot)} = 1)| \qquad (2) \\ &\le (\sigma^2 + \sigma)/2^{n+1}, \qquad (3)\end{aligned}$$

where $\sigma + 1$ denotes the number of queries to the oracle ($g(\cdot)$ or $P(\cdot)$).

Inequality (2) holds because for any adversary A we can construct another adversary B_A run as in Fig. 5. Adversary B_A always returns the same bit as adversary A does. That is, no matter when adversary A outputs "1", adversary B_A outputs "1". Furthermore, notice that adversary B_A may have some other occasions (not by running adversary A) to output "1". Consequently, inequality (2) follows.

Adversary B_A:
When A asks to its oracle O_A with $M^i \in \{0,1\}^*$,
B_A returns to A by $f9^-_{O_B}(M^i)$,
Until A stops and returns a bit d,
Then B_A returns d.

Fig. 5. Construct adversary B_A by adversary A.

Following the "PRP and PRF Switching Lemma" introduced in [8], we get inequality (3) holds.

In the end, combining inequalities (1) and (3) by the triangle inequality, we get

$$\begin{aligned}&|\Pr(P \xleftarrow{\$} \text{Perm}(n): A^{f9_P^-(\cdot)} = 1) - \Pr(R \xleftarrow{\$} \text{Rand}(*, n): A^{R(\cdot)} = 1)| \\ &\le |\Pr(g \xleftarrow{\$} \text{Rand}(n, n): A^{f9_g^-(\cdot)} = 1) - \Pr(P \xleftarrow{\$} \text{Perm}(n): A^{f9_P^-(\cdot)} = 1)| \\ &+ |\Pr(g \xleftarrow{\$} \text{Rand}(n, n): A^{f9_g^-(\cdot)} = 1) - \Pr(R \xleftarrow{\$} \text{Rand}(*, n): A^{R(\cdot)} = 1)| \\ &\le \frac{\sigma^2 + \sigma}{2^{n+1}} + \frac{\sigma^2 + \sigma}{2^{n+1}} \\ &= \frac{\sigma^2 + \sigma}{2^n}.\end{aligned}$$

Thus, we conclude the proof of Theorem 1.

Next, we pass this information-theoretic result to the complexity-theoretic result (the following Theorem 2) in the usual way [8], and we claim that $f9_E^-$ is a *viprf* if the underlying block cipher E is secure.

Theorem 2 ($f9^-_E \approx R$) Suppose block cipher E: $\{0, 1\}^k \times \{0, 1\}^n \to \{0, 1\}^n$ is used in $f9^-$, then we have $\mathrm{Adv}^{viprf}_{f9^-_E}(t, q, \sigma) \le (\sigma^2 + \sigma)/2^n + \mathrm{Adv}^{prp}_E(t', q')$, where $t' = t + O(\sigma)$, $q' = \sigma + 1$, and $\sigma = \sum_{i=1}^{q} \| M^i \|_n$.

At last, we say that $f9^-_E$ is a secure MAC algorithm by Theorem 2, since pseudo-random functions are secure MACs [8].

Theorem 3 ($f9^-_E$ is a Secure MAC) Suppose block cipher E: $\{0, 1\}^k \times \{0, 1\}^n \to \{0, 1\}^n$ is used in $f9^-$, then we have $\mathrm{Adv}^{mac}_{f9^-_E}(t, q, \sigma) \le (\sigma^2 + \sigma + 1)/2^n + \mathrm{Adv}^{prp}_E(t', q')$, where $t' = t + O(\sigma)$, $q' = \sigma + 1(\sigma)$, and $\sigma = \sum_{i=1}^{q} \| M^i \|_n$.

5. FURTHER CONSIDERATIONS

In fact, $f9^-$ could be improved to be a more efficient version, still keeping provable security with the same security bound. That is, eliminate the bottom line in the structure of $f9^-$, and we denote this new version as $f9^-_2$. Obviously, $f9^-_2$ is a completely Cipher Block Chaining mode, offering higher efficiency by saving multiple XOR operations compared with $f9^-$. What is more, we can modify the proof of $f9^-$ only a little (modify line 31 to $Y^i_{l_i} \leftarrow T^i$ in Fig. 6 to show that $f9^-_2$ is also provably secure with the same bound as $f9^-$. Alternatively, $f9^-_2$ can be viewed as a special case in OMAC-family [9] (the padded messages input to $f9^-_2$ is treated as the messages need not to be padded in OMAC-family); since OMAC-family is provably secure under the PRP assumption, so is $f9^-_2$.

```
01.  c^i_r ← 1 for i = 1, 2, ···, q and r = 2, 3, ..., l_i
02.  Domain ← G_0 ∪ G_1 ∪ G_2
11.  for r ← 2 to max{l_1, l_2, ..., l_q} do
12.    for i ← 1 to q do
13.      if (c^i_r = 1 && r ≤ l_i)
14.      then X ← {X^i_r}
15.        for j ← i + 1 to q do
16.          if (r ≤ l_j && D^i_1||D^i_2||...||D^i_{r-1} = D^j_1||D^j_2||...||D^j_{r-1})
17.          then c^i_r ← c^i_r + 1; X ← X ∪ {X^j_r}; c^j_r ← 0
18.          end if
19.        end for
20.        Y^i_{r-1} ←$ {0,1}^n; X^i_r ← Y^i_{r-1} ⊕ D^i_r
21.        for all X^j_r ∈ X, X^j_r ← X^i_r ⊕ D^i_r ⊕ D^j_r
22.        while (X ∩ Domain ≠ φ) do
23.          Y^i_{r-1} ←$ {0,1}^n; X^i_r ← Y^i_{r-1} ⊕ D^i_r
24.          for all X^j_r ∈ X, X^j_r ← X^i_r ⊕ D^i_r ⊕ D^j_r
25.        end while
26.        Domain ← Domain ∪ X
27.      end if
28.    end for
29.  end for
30.  for i ← 1 to q do
31.    Y^i_{l_i} ← Σ_{r=1}^{l_i-1} Y^i_r ⊕ T^i
32.  end for
```

Fig. 6. Evaluation of X^i_r (r = 2, 3, …, l_i) and Y^i_r (r = 1, 2, …, l_i) for i = 1, 2, …, q.

However, we prefer $f9^{-}$ other than $f9_2^{-}$ because the aim of our work is not only to improve $f9$, based on the original scheme, to be a more efficient and provably secure version, but also to make sure that the modification is small, so as to update $f9$ easily in practice. As a result, $f9^{-}$ successfully achieves all these goals, keeping the main $f9$ structure almost unchanged; while $f9_2^{-}$ destroys the main structure of $f9$, although it offers higher efficiency. From an engineering point of view, it is much easier to update $f9$ to $f9^{-}$ in practice, other than $f9_2^{-}$.

6. EFFICIENCY COMPARISON

Note that all the $f9$ variants $f9'$, $f9^{+}$ and $f9^{-}$ can be seen as authentication modes of operation for block ciphers. In this point of view, the number of block-cipher invocation to generate a MAC (#E invo.) has great influence on the efficiency of mode of operation. Moreover, the number of block-cipher key schedulings (#K sche.) and the number of block-cipher invocations during the preprocessing time (#E pre.) are also important parameters to evaluate the efficiency of different modes of operation.

In this section, we compare $f9^{-}$ with $f9'$ and $f9^{+}$ in aspect of efficiency, and show the detailed results in Table 1.

Table 1. Efficiency comparison among $f9^{-}$, $f9'$ and $f9^{+}$.

	$f9^{-}$	$f9'$	$f9^{+}$
#K sche.	1	2	1
#E pre.	1	0	2
#E invo.	$\lceil (\|M\| + 1)/n \rceil$	$\lceil (\|M\| + 1)/n \rceil + 1$	

By Table 1, it is clear that $f9'$ has one more "#K sche." than $f9^{-}$ and $f9^{+}$, because there is an extra key scheduling $K \oplus \Delta$ in the last block-cipher invocation of $f9'$; furthermore, there is one less "#E pre." in $f9^{-}$ than $f9^{+}$, because $f9^{-}$ has to pre-compute $S \cdot x = E_K(0^n) \cdot x$ only while $f9^{+}$ has to pre-compute $E_K(0^n)$ and $E_K(1^n)$. Notice that $E_K(\text{Cst}) \cdot x$ can be done by $S = E_K(\text{Cst})$ and $S \cdot x$, where the latter one is much faster than a block- cipher invocation.

More importantly, $f9^{-}$ saves one "#E invo." compared with $f9'$ and $f9^{+}$, which implies $f9^{-}$ can save one block-cipher invocation for every message. This improvement is more effective when lots of short messages have to be processed, *e.g.* in cellphone communications where $f9$ is used.

Overall, Table 1 tells us $f9^{-}$ offers higher efficiency than the others.

7. CONCLUSION

In summary, we introduce a new variant of 3GPP-MAC, $f9^{-}$, in this paper. $f9^{-}$ is provably secure for arbitrary length messages, modeling its underlying block cipher as a secure PRP (PseudoRandom Permutation). Compared with $f9$, $f9'$ and $f9^{+}$, $f9^{-}$ removes the last block-cipher invocation, offering higher efficiency in practice. What is more, $f9^{-}$ keeps the main structure of $f9$ almost unchanged, making it easy to update $f9$ to $f9^{-}$ if necessary.

REFERENCES

1. ANSI X9.19: Financial Institution Retail Message Authentication, 1986.
2. ETSI TS 35.201 V7.0.0: Specification of the 3GPP confidentiality and integrity algorithms; document 1: f8 and f9 specification, http://www.3gpp.org/ftp/Specs/html-info/35201.htm.
3. D. Hong, J. S. Kang, B. Preneel, and H. Ryu, *A Concrete Security Analysis for 3GPP-MAC*, LNCS, Springer, Heidelberg, Vol. 2887, 2003, pp. 154-169.
4. T. Iwata and K. Kurosawa, *On the Correctness of Security Proofs for the 3GPP Confidentiality and Integrity Algorithms*, LNCS, Springer, Heidelberg, Vol. 2898, 2003, pp. 306-318.
5. T. Iwata and T. Kohno, *New Security Proofs for the 3GPP Confidentiality and Integrity Algorithms*, B. K. Roy, and W. Meier, eds., LNCS, Springer, Heidelberg, Vol. 3017, 2004, pp. 427-445.
6. T. Iwata and K. Kurosawa, *How to Enhance the Security of the 3GPP Confidentiality and Integrity Algorithms*, H. Gilbert, and H. Handschuh, eds., LNCS, Springer, Heidelberg, Vol. 3557, 2005, pp. 268-283.
7. R. Lidl and H. Niederreiter, *Introduction to Finite Fields and Their Applications*, Cambridge University Press, New York, 1986.
8. M. Bellare, J. Kilian, and P. Rogaway, *The Security of Cipher Block Chaining*, Y. Desmedt, ed., LNCS, Springer, Heidelberg, Vol. 839, 1994, pp. 341-358.
9. T. Iwata and K. Kurosawa, *Omac*: *One-Key CBC MAC*, T. Johansson, ed., LNCS, Springer, Heidelberg, Vol. 2887, 2003, pp. 129-153.
10. T. Iwata and K. Kurosawa, *Stronger Security Bounds for OMAC, TMAC, and XCBC*, T. Johansson and S. Maitra, eds., LNCS, Springer, Heidelberg, Vol. 2904, 2003, pp. 402-415.
11. J. Patarin, *Luby-Rackoff*: *7 Rounds are Enough for* $2^{n(1-\varepsilon)}$ *Ecurity*, D. Boneh, ed., LNCS, Springer, Heidelberg, Vol. 2729, 2003, pp. 513-529.
12. S. Vaudenay, *Decorrelation over Infite Domains*: *the Encrypted CBC-MAC Case*, D. R. Stinson and S. E. Tavares, eds., LNCS, Springer, Heidelberg, Vol. 2012, 2000, pp. 189-201.

APPENDIX A

Lemma 1 is not a new tool in discussion of block cipher structures and modes of operation. The "Coefficients H Technique" developed by Jacques Patarin [11] and the decorrelation theory developed by Serge Vaudenay have similar forms [12].

***Proof*:** Adversary A has access to oracle O which is either $F_K(\cdot)$ or $R(\cdot)$. We run A with oracle O, and suppose A makes q queries $M^1, M^2, \cdots, M^q$, $M^i \in \{0, 1\}^*$ to O, where these q queries are pairwise distinct. Since A is computationally unbounded, without loss of generality we can assume that A is deterministic. Therefore, the ith query $M^i \in \{0, 1\}^*$ A makes is fully determined by the previous $i - 1$ answers $T^1, T^2, \ldots, T^{i-1} \in \{0, 1\}^n$ from oracle O. Thus, A's queries are uniquely determined, q is uniquely determined, and the final output of A (0 or 1) is uniquely determined. In other words, the final output of A

fully depends on what the q-tuple $(T^1, T^2, \ldots, T^q)$ is. Then, we let $V_{\text{one}} = \{V = (V^1, V^2, \ldots, V^q) \mid V^i \in \{0, 1\}^n, i = 1, 2, \ldots, q\}$ be the set of all q-tuple $V = (V^1, V^2, \ldots, V^q)$ such that A outputs 1 when given $V = (V^1, V^2, \ldots, V^q)$ by O, and we let $N_{\text{one}} = \#V_{\text{one}}$. Noticing that $\#T \geq 2^{qn}(1 - \varepsilon_1)$, *i.e.* condition 1) in this Lemma, we have $\#\{V|V \in V_{\text{one}} \cap T\} \geq N_{\text{one}} - \varepsilon_1 2^{qn}$.

$$P_{rand} \triangleq \Pr(R \xleftarrow{\$} \text{Rand}(*, n): A^{R(\cdot)} = 1)$$

Evaluation of p_{rand}:

$$= \frac{\#\{R|A^{R(\cdot)} = 1\}}{\#\text{Rand}(*, n)}$$

$$= \sum_{V \in V_{\text{one}}} \frac{\#\{R|R \text{ satisfying } V^i = R(M^i) \text{ for } i = 1, 2, \cdots, q\}}{\#\text{Rand}(*, n)}.$$

Notice that $|\Pr(K \xleftarrow{\$} \text{Rand}(*, n): T^i = R(M^i) \text{ for } i = 1, 2, \ldots, q) = (1/2^n)^q$, so

$$p_{rand} = \sum_{V \in V_{\text{one}}} \frac{1}{2^{qn}} = \frac{N_{\text{one}}}{2^{qn}}.$$

Evaluation of p_{real}:

$$P_{real} \triangleq \Pr(K \xleftarrow{\$} \{0, 1\}^k: A^{F_K(\cdot)} = 1)$$

$$= \frac{\#\{K|A^{F_K(\cdot)} = 1\}}{2^k}$$

$$= \sum_{V \in V_{\text{one}}} \frac{\#\{K|F_K \text{ satisfying } V^i = F_K(M^i) \text{ for } i = 1, 2, \cdots, q\}}{2^k}$$

$$\geq \sum_{V \in (V_{\text{one}} \cap T)} \frac{\#\{K|F_K \text{ satisfying } V^i = F_K(M^i) \text{ for } i = 1, 2, \cdots, q\}}{2^k}$$

By condition (2), we get $p_{real} \geq \sum_{V \in (V_{\text{one}} \cap T)} \frac{1 - \varepsilon_2}{2^{qn}} \geq (N_{\text{one}} - \varepsilon_1 2^{qn}) \times \frac{1 - \varepsilon_2}{2^{qn}}$. Then, it is not hard to obtain $p_{real} = p_{rand} - \varepsilon_2 p_{rand} - \varepsilon_1 + \varepsilon_1 \varepsilon_2$.

Now, it is easy to see that,

$$P_{rand} - P_{real} \leq \varepsilon_2 P_{rand} + \varepsilon_1 - \varepsilon_1 \varepsilon_2 \leq \varepsilon_1 + \varepsilon_2; \tag{4}$$

in the same way, if we define a set V_{zero} just as V_{one}, and apply similar arguments, we will get

$$(1 - p_{rand}) - (1 - p_{real}) \leq \varepsilon_1 + \varepsilon_2. \tag{5}$$

By inequalities Eqs. (4) and (5), we know that $|P_{rand} - P_{real}| \leq \varepsilon_1 + \varepsilon_2$. In conclusion, for any computationally unbounded adversary A, it follows that,

$$|\Pr(K \xleftarrow{\$} \{0, 1\}^k: A^{F_K(\cdot)} = 1) - \Pr(R \xleftarrow{\$} \text{Rand}(*, n): A^{R(\cdot)} = 1)| \leq \varepsilon_1 + \varepsilon_2.$$

APPENDIX B

***Proof*:** (of Lemma 2) For generality, we prove Lemma 2 for $f9^-$ with any fixed Cst $\in \{0, 1\}^n$ (see Fig. 3). Notice that $f9^-$ with Cst = 0^n is a special case of $f9^-$ with any fixed Cst, so our proof applies to $f9^-$ with Cst = 0^n naturally.

Step 1: We pad the q-tuple $(M^1, M^2, \ldots, M^q)$ by the method given in $f9^-$ algorithm, and then partition M^i into $M^i_1, M^i_2, \ldots, M^i_{l_i}$, where $|M^i_r| = n$ for $i = 1, 2, \ldots, q$ and $r = 1, 2, \ldots, l_i$.

Step 2: Let $D^i_r = M^i_r$ (for $i = 1, 2, \ldots, q$ and $r = 1, 2, \ldots, l_i - 1$) and $D^i_{l_i} = M^i_{l_i} \oplus (S \cdot x)$. Notice that variable $S = g(\text{Cst})$ has not been evaluated, so $D^i_{l_i}$ is a monomial of variable S.

Step 3: We define X^i_r for $i = 1, 2, \ldots q$ and $i = 1, 2, \ldots, l_i$ as the inputs to function g and Y^i_r as their corresponding outputs. According to the algorithm of $f9^-$, it stands to reason that, for $i = 1, 2, \ldots, q$,

$$\begin{cases} X^i_1 = D^i_1, \\ X^i_r = g(X^i_{r-1}) \oplus D^i_r = Y^i_{r-1} \oplus D^i_r, \text{ for } r = 2, 3, \cdots, l_i, \\ T^i = \sum_{r=1}^{l_i} Y^i_r. \end{cases} \tag{8}$$

What is more, notice that if there exists M^i such that $|M^i| \geq 2$ and M^i_1 = Cst, then $X^i_1 = D^i_1 = M^i_1 =$ Cst and $X^i_2 = g(X^i_2) \oplus D^i_2 = g(\text{Cst}) \oplus D^i_2 = S \oplus D^i_2$. In such a case, we write $X^i_2 = S \oplus D^i_2$ directly. Finally, we let $X^i = X^i_1 \| X^i_2 \| \ldots \| X^i_{l_i}$.

Step 4: Define some sets $G_0 = \{\text{Cst}\}$, $G_1 = \{X^i_1, i = 1, 2, \ldots, q\}$, $G_2 = \{X^i_2 | X^i_1 = \text{Cst}, i = 1, 2, \ldots, q\}$, $G_3 = \{(X^i_{l_i}, X^j_{l_j}) | 1 \leq i < j \leq q, l_i = l_j \geq 2, X^i_1 \| X^i_2 \| \ldots \| X^i_{l_i - 1} = X^j_1 \| X^j_2 \| \ldots \| X^i_{l_j - 1}\}$, and $Domain = G_0 \cup G_1 \cup G_2$. Then, we evaluate $S = g(\text{Cst}) \in \{0, 1\}^n$ such that

$$\begin{cases} \text{the elements in Domain are pairwise distinct,} \\ X^i_{l_i} \neq X^j_{l_j}, \forall (X^i_{l_i}, X^j_{l_j}) \in G_3. \end{cases} \tag{9}$$

Noticing the fact that $\#Domain + G_3 \leq q + 1$, it is not hard to see inequality system (9) contains at most $\binom{q+1}{2}$ inequalities with variable S. According to the properties of ope ration in GF(2^n), inequality system (9) is resolvable (refer to [7] for more information). We fix any S satisfying inequality system (9), and the number of choices of such S is $N_S \geq 2^n - \binom{q+1}{2}$.

Step 5: In this step, X^i_r ($r = 2, \ldots, l_i$) and Y^i_r ($r = 1, 2, \ldots, l_i$) for $i = 1, 2, \ldots, q$ are going to be evaluated by a pseudocode program (see Fig. 6). In the pseudocode program, set X contains all the X^j_r related to X^i_r, by "related" we mean condition $X^i_r \oplus X^j_r = D^i_r \oplus D^j_r$ holds (because of $D^i_1 \| D^i_2 \| \ldots \| D^i_{r-1} = D^j_1 \| D^j_2 \| \ldots \| D^j_{r-1}$). In or-

der to count the number of X_r^j related to X_r^i, we define a counter c_r^i for X_r^i. The counter c_r^i works as follows: once condition $X_r^i \oplus X_r^j = D_r^i \oplus D_r^j$ holds, we let c_r^i +1, $X \leftarrow X \cup \{X_r^j\}$ and $c_r^j \leftarrow 0$ (line 17). We let $c_r^j \leftarrow 0$ because we will not consider the evaluation of X_r^j independently. In other words, we will consider the evaluation of X_r^i first and then let $X_r^j = D_r^i \oplus D_r^j \oplus X_r^i$.

Set Domain defines the domain of function g and it increases with our evaluation for X_r^i (i = 1, 2, …, q and r = 2, …, l_i) step by step. At the beginning of the program, we let $Domain \leftarrow G_0 \cup G_1 \cup G_2$ (line 02), because the elements in sets G_0, G_1 and G_2 are exactly the inputs has been (it is Cst) or are going to be (those except Cst) input to function g.

As long as we have found all the X_r^j related to X_r^i we start the evaluation of X_r^i by line 20 in the program. That is, we randomly select Y_{r-1}^i, and then let $X_r^i \leftarrow Y_{r-1}^i \oplus D_r^i$. In the meanwhile, we can obtain $X_r^j = D_r^i \oplus D_r^j \oplus X_r^i$ for all $X_r^j \in X$ (line 21). Then, we check whether condition $X \cap Domain \neq \phi$ holds or not (line 22). If it holds, we randomly select Y_{r-1}^i again (line 23) until condition $X \cap Domain \neq \phi$ does not hold. Obviously, there are at least $2^n - (\#Domain + (\#Domain + 1) + \ldots + (\#Domain + \#X - 1)) \geq 2^n - (\#Domain + (\#Domain + 1) + \ldots + (\#Domain + c_r^j - 1)$ choices of Y_{r-1}^i satisfying the condition in line 22. Once we have found a satisfying Y_{r-1}^i, we let $Domain \leftarrow Domain \cup X$ (line 26) which also means we have finished the evaluation of all the X_r^j in set X.

Run the program until line 29, now we have obtained that $X_{l_i}^i$ are pairwise distinct and $Y_{l_i}^i = g(X_{l_i}^i)$ have not been evaluated. Next, we let $Y_{l_i}^i \leftarrow \sum_{r=1}^{l_i-1} Y_r^i \oplus T^i$ (line 31). Then the program finishes, all the X_r^i (r = 2, …, l_i) and Y_r^i (r = 1, 2, …, l_i) for i = 1, 2, …, q have been evaluated. Next, by the Eq. (9) in step 3, we can establish a bridge between M^i and T^i for i = 1, 2, …, q. Finally, we successfully find a class of function g satisfying $T^i = f_{9_g^-}(M^i)$ for i = 1, 2, …, q.

Step 6: In this step, we count the number of function g found in the previous steps. Before the program starts, notice the facts that $N_0 = \#Domain \leq q + 1$ and $\sum_{i=1}^{q} \sum_{r=2}^{l_i} c_r^i = \sigma - q$, where $\sigma = \sum_{i=1}^{q} l_i$. When the program runs to line 29, suppose that the number of c_r^i satisfying $c_r^i \geq 1$ is z. We write these c_r^i as $(c_r^i)_1$, $(c_r^i)_2$, …, $(c_r^i)_z$ in the order of their first appearance in the program. By the lines 01 and 17 in the program, we know that $\sum_{w=1}^{z} c_r^i = \sum_{i=1}^{q} \sum_{r=2}^{l_i} c_r^i = \sigma - q$.

Furthermore, we let $N_1 = N_0 + (c_r^i)_1$, $N_2 = N_1 + (c_r^i)_2$, …, $N_z = N_{z-1} + (c_r^i)_z$. Here, N_0, N_1, …, N_z can be seen as $\#Domain$ in different stages when the program is running. Obviously, $N_z = N_0 + \sum_{w=1}^{z} c_r^i = N_0 + \sigma - q \geq \sigma + 1$. What is more, we have $\#Domain = N_z \leq \sigma + 1$ after line 29. Then the program continues to run from line 30. Since we let $Y_{l_i}^i \leftarrow \sum_{r=1}^{l_i-1} Y_r^i \oplus T^i$ (line 31), each $Y_{l_i}^i$ has a unique choice in $\{0, 1\}^n$.

Now we review the process from the beginning to the end. First, we fix S satisfying some inequalities (9) and the number of choices of S is $N_S \geq 2^n - \binom{q+1}{2}$. Then, we evaluate X_r^i for i = 1, 2, …, q and r = 2, …, l_i by a program (Fig. 6). Notice the fact that $(c_r^i)_w \geq 1$ (w = 1, 2, …, z) indicates its corresponding X_r^i has $(c_r^i)_w$ related X_r^j (including

itself); consequently, the wth $X_r^i (w = 1, 2, \ldots, z)$ has at least $2^n - (N_{w-1} + (N_{w-1} + 1) + \ldots + (N_{w-1} + \#X - 1)) \geq 2^n - (N_{w-1} + (N_{w-1} + 1) + \ldots + (N_{w-1} + (c_r^i)_w - 1))$ choices satisfying the condition $X \cap Domain = \phi$. Next, we fix $Y_{l_i}^i$ to $\sum_{r=1}^{l_i-1} Y_r^i \oplus T^i$ for $i = 1, 2, \ldots, q$; that is, each $Y_{l_i}^i$ has a unique choice in $\{0, 1\}^n$. Finally, according to the Eq. (8), we establish a bridge between M^i and T^i for $i = 1, 2, \ldots, q$, leaving function g $2^n - (z + q + 1)$ elements in its domain undefined. Since these $2^n - (z + q + 1)$ elements have nothing to do with $T^i = f9_g^-(M^i)$ for $i = 1, 2, \ldots, q$, we randomly select a value from $\{0, 1\}^n$ for each of them. In other words, these $2^n - (z + q + 1)$ elements have 2^n choices each.

Denote N_g as the number of function g found in the previous steps. Based on the former analysis, we know that $N_g = N_S \times N_{11-19} \times N_{31-33} \times (2^n)^{2^n-(z+q+1)}$, where N_{11-29} is the number of choices of function g at some points ($g(X_r^i)$ for $i = 1, 2, \ldots, q$ and $r = 2, \ldots, l_i - 1$) from line 11 to 29 in Fig. 6, and

$$\begin{aligned} N_{11-29} &= [2^n - (N_0 + (N_0 + 1) + \ldots + (N_0 + (c_r^i)_1 - 1))] \\ &\quad \times [2^n - (N_1 + (N_1 + 1) + \ldots + (N_1 + (c_r^i)_2 - 1))] \times \ldots \\ &\quad \times [2^n - (N_{z-1} + (N_{z-1} + 1) + \ldots + (N_{z-1} + (c_r^i)_z - 1))] \\ &\quad > 2^{zn} - (N_0 + (N_0 + 1) + \ldots + N_z - 1))2^{(z-1)n}, \end{aligned}$$

$N_{31-33} = 1^q$ is the number of choices of function g at some points ($g(X_{l_i}^i)$ for $i = 1, 2, \ldots, q$) from line 30 to 32 in Fig. 6. Consequently,

$$\begin{aligned} N_g &= N_S \times N_{11-19} \times N_{31-33} \times (2^n)^{2^n-(z+q+1)} \\ &\geq 2^n - \binom{q+1}{2} \times [2^{zn} - (N_0 + (N_0 + 1) + \ldots + (N_z - 1))2^{(z-1)n}] \times 1^q \times (2^n)^{2^n-(z+q+1)} \\ &\geq [2^{(z+1)n} - (1 + 2 + \ldots + (N_z - 1))2^{zn}] \times (2^n)^{2^n-(z+q+1)} \\ &= (1 - \frac{\sigma^2 + \sigma}{2^{n+1}}) \times (2^n)^{2^n-q}. \end{aligned}$$

Li-Ting Zhang (張立廷) is currently a Ph.D. student in Graduate School of Chinese Academy of Sciences and Institute of Software, Chinese Academy of Sciences. His current research area includes provable security on block ciphers, message authentication codes and hash functions.

Wen-Ling Wu (吳文玲) is now a Professor at the State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences. She received her Ph.D. degree in Cryptography from Xidian University in 1997. Her current research interests include theory of cryptography, modes of operation, block ciphers, stream ciphers and hash functions.

Peng Wang (王鹏) is a Lecturer in Graduate University of Chinese Academy of Sciences (GUCAS). He received his Ph.D. degree from GUCAS in 2006. His current research interests include theory of cryptography, especially about symmetric cipher.