



(51) International Patent Classification:

H04L 9/06 (2006.01) G09C 1/00 (2006.01)

(21) International Application Number:

PCT/SG2021/050323

(22) International Filing Date:

04 June 2021 (04.06.2021)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

10202005341W 05 June 2020 (05.06.2020) SG

(71) Applicant: NANYANG TECHNOLOGICAL UNIVERSITY [SG/SG]; 50 Nanyang Avenue, Singapore 639798 (SG).

(72) Inventors: WANG, Haoyang; c/o Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798 (SG). PEYRIN, Thomas, Jérôme, Léonardo, Xavier; c/o Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798 (SG).

(74) Agent: LINDSAY, Jonas, Daniel; Marks & Clerk Singapore LLP, Tanjong Pagar Post Office, P.O. Box 636, Singapore 910816 (SG).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, IT, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW,

(54) Title: METHODS AND SYSTEMS FOR GENERATING A BLOCK CIPHER HAVING BACKDOOR

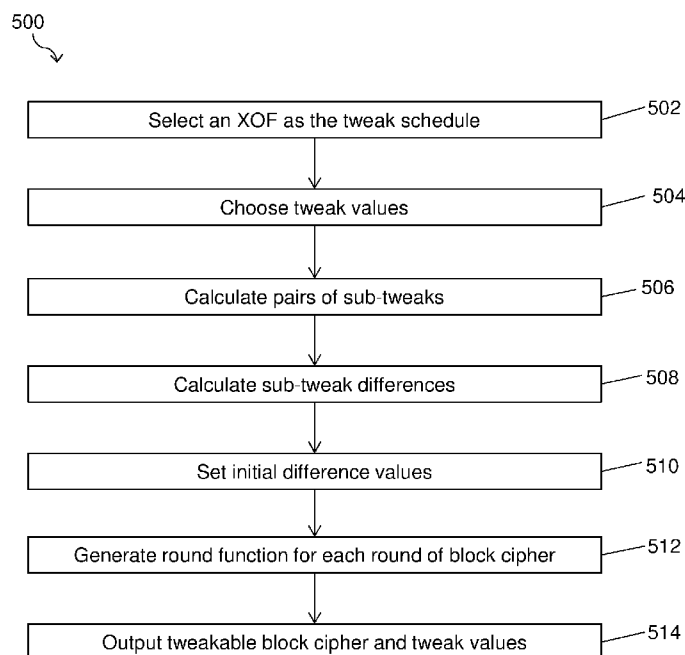


FIG.5

(57) Abstract: Methods and systems for generating a block cipher having a backdoor are disclosed. A method of generating a block cipher having a backdoor comprises: selecting a tweak schedule function as a tweak schedule for the block cipher; choosing a set of tweak values; calculating a set of pairs of sub-tweaks by applying the tweak schedule function to the set of tweak values, each pair of sub-tweaks of the set of pairs of sub-tweaks corresponding to a round of the block cipher; calculating a sub-tweak difference for each round of the block cipher as a difference between the pair of sub-tweaks corresponding to that round of the block cipher; generating a round function for each round of the block cipher, each round function having a layer comprising a non-linear part, wherein the non-linear part is configured with a differential characteristic such the output difference after a respective round of the block cipher is equal to the sub-tweak difference for the respective round of the block cipher; and outputting the block cipher as the round functions for each



SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN,
TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) Designated States (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *of inventorship (Rule 4.17(iv))*

Published:

— *with international search report (Art. 21(3))*

METHODS AND SYSTEMS FOR GENERATING A BLOCK CIPHER HAVING BACKDOOR

TECHNICAL FIELD

[0001] The present disclosure relates to encryption and in particular to the generation of block ciphers having a backdoor.

BACKGROUND

[0002] A backdoor in an encryption algorithm enables an entity who knows the backdoor to circumvent the security guarantees so that they can obtain secret information more efficiently than with a generic black-box attack. There are two categories of backdoors. The first one is the backdoor implemented in a security product at the protocol or key-management level, which is generally considered in practice. The second is a cryptographic backdoor. A cryptographic backdoor is embedded directly during the design phase of a cryptographic primitive and renders the cipher susceptible to some dedicated cryptanalysis. There are very few publicly known backdoored primitives. A concrete example is the pseudorandom number generator Dual_EC_DRBG designed by the United States National Security Agency (NSA), whose backdoor was revealed by Edward Snowden in 2013.

[0003] Embedding backdoors into block ciphers is a challenging problem since block ciphers are deterministic and thus it is complex to exploit randomness in computations. A backdoor should be computationally difficult to retrieve, even if its general form is known. More concretely, the backdoor security (the cost of retrieving the backdoor) should be the same as the security generically provided by the cipher (otherwise the backdoor would naturally reduce the security of the block cipher). Besides, the backdoor should ideally lead to a practical key recovery attack, or at least reduce the brute force search cost for the adversary. For example, if a backdoor could reduce the security of AES-256 to 2^{128} , it would be a great theoretical advance, but would be unusable in practice. Last but not least, the resulting block cipher also has to be secure in the classical sense, that is, it is able to resist state-of-the-art cryptanalysis techniques.

SUMMARY

[0004] The present disclosure provides a new method to generate backdoor encryption algorithms.

[0005] According to a first aspect of the present disclosure, a method of generating a block cipher having a backdoor is provided. The method comprises: selecting a tweak schedule function as a tweak schedule for the block cipher; choosing a set of tweak values; calculating a set of pairs of sub-tweaks by applying the tweak schedule function to the set of tweak values, each pair of sub-tweaks of the set of pairs of sub-tweaks corresponding to a round of the block cipher; calculating a sub-tweak difference for each round of the block cipher as a difference between the pair of sub-tweaks corresponding to that round of the block cipher; generating a round function for each round of the block cipher, each round function having a layer comprising a non-linear part wherein the non-linear part is configured with a differential characteristic such the output difference after a respective round of the block cipher is equal to the sub-tweak difference for the respective round of the block cipher; and outputting the block cipher as the round functions for each round, the tweak schedule function and the set of tweak values, wherein the set of tweak values correspond to the backdoor.

[0006] In an embodiment, the tweak schedule function is an extendable-output function. Embodiments of the present invention bring together tweakable block ciphers (TBC) and Extendable-Output Function (XOF) in a common framework, which enables the designer to embed backdoors into the TBC. The tweak is handled separately by the XOF and the round function has to be partially non-linear.

[0007] The layer comprising a non-linear part may be a partial non-linear layer or a fully non-linear layer.

[0008] The backdoor is based on differential cryptanalysis: due to the non-linear part of the layer of the round function, the designer can embed multiple related-tweak differential characteristics with probability 1 over many rounds. In particular, the sub-

tweak difference employed in an embedded differential characteristic is generated from a specific tweak pair that is chosen in advance by the designer. This malicious tweak pair is the backdoor, and the XOF applied in the tweak schedule is used to protect the malicious tweak pair: even knowing the high-probability related tweak differential characteristic, it will remain computationally difficult to find a tweak pair that triggers it. More importantly, the backdoor security is ensured by the target-difference resistance ability of the chosen XOF. An attacker with the knowledge of the backdoor is able to retrieve the full key with negligible effort under the chosen-tweak scenario.

[0009] In an embodiment, the extendable-output function is based on a secure hash algorithm standard.

[0010] In an embodiment, wherein the layer comprising a non-linear part is a partial non-linear layer and the method further comprises selecting a plaintext difference as a difference in an initial round of the block cipher for a linear part of the non-linear layer and the sub-tweak difference for the initial round of the block cipher as the difference for the non-linear part of the non-linear layer.

[0011] In an embodiment, the block cipher is based on a substitution–permutation network.

[0012] In an embodiment, in each round of the cipher, a non-linear part of an internal state of the cipher is combined with a sub-tweak corresponding to the respective round in a tweak addition operation.

[0013] In an embodiment, the set of tweak values comprises a plurality of pairs of tweak values. In an embodiment, each pair of tweak values of the plurality of pairs of tweak values differs from the each of the remaining pairs of tweak values.

[0014] According a second aspect of the present disclosure, a computer readable carrier medium carrying processor executable instructions which when executed on a processor cause the processor to carry out a method set out above is provided.

[0015] According to a third aspect of the present disclosure a system for generating a block cipher having a backdoor is provided. The system comprises a processor and a data storage device storing computer program instructions operable to cause the processor to: select a tweak schedule function for the block cipher; choose a set of tweak values; calculate a set of pairs of sub-tweaks by applying the tweak schedule function to the set of tweak values, each pair of sub-tweaks of the set of pairs of sub-tweaks corresponding to a round of the block cipher; calculate a sub-tweak difference for each round of the block cipher as a difference between the pair of sub-tweaks corresponding to that round of the block cipher; generate a round function for each round of the block cipher, each round function having a layer comprising a non-linear part, wherein the non-linear part is configured with a differential characteristic such the output difference after a respective round of the block cipher is equal to the sub-tweak difference for the respective round of the block cipher; and output the block cipher as the round functions for each round, the tweak schedule function and the set of tweak values, wherein the set of tweak values correspond to the backdoor.

[0016] According to a fourth aspect of the present disclosure, a communications device is provided. The communications device comprises an encryption module storing a tweakable block cipher and a shared secret key, the encryption module being configured to encrypt a plaintext message using the tweakable block cipher and the shared secret key, wherein the tweakable block cipher comprises a plurality of round functions, each corresponding to a round of the block cipher, each round function having a layer comprising a non-linear part, wherein the non-linear part is configured with a differential characteristic such the output difference after a respective round of the block cipher is equal to a sub-tweak difference between a pair of sub-tweaks for the respective round of the block cipher.

[0017] In an embodiment, the pairs of sub-tweaks for each respective round of the block cipher are obtainable by applying a tweak schedule function to a set of tweak values.

[0018] In an embodiment, the tweak schedule function is an extendable-output function.

BRIEF DESCRIPTION OF THE DRAWINGS

[0019] In the following, embodiments of the present invention will be described as non-limiting examples with reference to the accompanying drawings in which:

[0020] FIG. 1A to FIG. 1C are block diagrams showing an overview the generation and use of tweakable block ciphers having a backdoor according to embodiments of the present invention;

[0021] FIG. 2 depicts a tweakable block cipher framework according to an embodiment of the present invention;

[0022] FIG. 3 shows transitions of state difference in the embedded related-tweak differential characteristic in a framework according to an embodiment of the present invention;

[0023] FIG. 4 is a block diagram showing a block cipher generation system according to an embodiment of the present invention;

[0024] FIG. 5 is a flowchart showing a method of constructing an instance of a tweakable block cipher with a backdoor according to an embodiment of the present invention;

[0025] FIG. 6 shows a variant of the framework in which the tweak differential characteristic is only applied once at the beginning;

[0026] FIG. 7 shows swapping of the order of operations to provide a simplified representation of LowMC block ciphers used in embodiments of the present invention;

[0027] FIG. 8 shows the operations making up a round function in an implementation of a framework for generating a tweakable block cipher according to an embodiment of the present invention;

[0028] FIG.9 shows deterministic differential characteristics embedded into LowMC-M in an embodiment of the present invention;

[0029] FIG.10 is a table showing a range of different parameters sets of LowMC-M instantiations; and

[0030] FIG.11 is a table showing results of backdoor security evaluation.

DETAILED DESCRIPTION

[0031] For classical (tweakable) block ciphers, the attacking scenario considers only two entities: the user (or pair of users) who owns the secret key and the attacker who tries to break the cryptosystem, i.e., to find out the secret key. For (tweakable) block ciphers with a backdoor, another entity has to be involved in the attacking scenario: the designer, who inserts the backdoor into the primitive. Thus, we have in total three entities: the designer (knows the backdoor, but not the secret key), the user (knows the secret key, but not the backdoor) and the attacker (neither backdoor nor key is known).

[0032] One can see that both the user and the attacker have some motivation to find out what is the backdoor. More importantly, in our model the backdoor is independent of the secret key, and therefore the user and the attacker possess the same capability in trying to uncover the backdoor (the cipher specifications are public known, so they can test the cipher with any chosen key they want). For the rest of this disclosure, when considering the recovery of the backdoor, we will simply refer to both of them as the attacker.

[0033] FIG.1A to FIG.1C are block diagrams showing an overview the generation and use of tweakable block ciphers having a backdoor according to embodiments of the present invention.

[0034] As shown in FIG.1A, the designer uses a block cipher generation system 100 to generate a tweakable block cipher 102 and a backdoor 104 for the tweakable block

cipher. The backdoor comprises tweaks for the tweakable block cipher 102. As will be described in more detail below, the tweakable block cipher 102 has a partial non-linear layer with embedded tweak related differential characteristics that have probability 1 over many rounds. This allows the tweaks to be used as the backdoor 104.

[0035] As shown in FIG.1B, the tweakable block cipher 102 is provided to users. Tweaks are shared between a sender and a receiver as a parameter of the tweakable block cipher 102. The tweaks shared between the sender and the receiver may be randomly selected. The sender encrypts a plaintext message 210 using a sender device 220 to generate ciphertext 230. The sender device 220 comprises an encryption module 222 which stores the tweakable block cipher 102 and a shared secret key 224. The encryption module 222 stores the shared tweaks as parameters of the tweakable block cipher. The encryption module 222 uses the tweakable block cipher 102 and the shared secret key 224 to encrypt the plaintext message 210 as ciphertext 230. The receiver who has a copy of the shared secret key 224 can decrypt the ciphertext 230 to retrieve the plaintext message 210. A receiver device 240 comprises a decryption module 242 which stores the tweakable block cipher 102 and the shared secret key 224. The decryption module 242 stores the shared tweaks as parameters of the tweakable block cipher. The decryption module 242 uses the tweakable block cipher 102 and the shared secret key 224 to decrypt the ciphertext 230 and thereby retrieves the plaintext message 210.

[0036] As shown in FIG.1C, a party such as the designer in possession of the backdoor 104 can recover the shared secret key 224. In order to do this, the designer requests the sender device 220 to encrypt a specific plaintext message 212. The encryption module 222 of the sender device 220 uses the tweakable block cipher with the tweaks that form the backdoor 104 applied to it and the shared secret key 224 to generate ciphertext 232. From this specific ciphertext 232, the designer can recover the shared secret key 224 using a technique called differential cryptanalysis, in a very short time.

[0037] We introduce below various notions regarding the security and the practicability of a backdoor:

- a. *Undetectability*: this security notion represents the inability for an external entity to realize the existence of the hidden backdoor.
- b. *Undiscoverability*: it represents the inability for an attacker to find the hidden backdoor, even if the general form of the backdoor is known.
- c. *Untraceability*: it states that an attack based on the backdoor should not reveal any information about the backdoor itself.
- d. *Practicability*: this usability notion stipulates that the backdoor is practical, in the sense that it is easy to recover the secret key once the backdoor is known.

[0038] If a cipher is publicly claimed as potentially backdoored, it will naturally increase the watchfulness of users, even if they do not know whether there is indeed backdoored or not embedded in the primitive. In this scenario, the undetectability notion models the incapacity of a user to find any hard evidence that a backdoor indeed exists.

[0039] Given a bit string x , we will denote by $x[i]$ its i -th bit, counting from the least significant bit (LSB). Given two bit strings x and y , $x||y$ will represent the concatenation of x and y . Finally, we denote by k_j (respectively by t_j) the sub-key (respectively sub-tweak) incorporated during the j -th round of the cipher, while k_0 and t_0 are added in as whitening material.

[0040] In this section, we introduce a framework which allows to generate tweakable block ciphers that are embedded with hidden high-probability differential characteristics. This framework is based on partial non-linear layers for the internal state transformation and a tweak schedule based on an extendable output function (XOF).

[0041] Substitution-Permutation Network (SPN) based block ciphers are usually designed to apply linear layers (L_i) and non-linear layers (S_i) to the entire state at every round i . We consider a design in which the non-linear layer is only applied to a subpart of the state at each round with block size n bits and partial non-linear layers of size s ($< n$) bits. Assume, without loss of generality, that the non-linear layer is always applied before the linear layer at every round. Then, we can write $f_i(x) =$

$L_i(S_i(x^{(0)})||x^{(1)})$ the round function f_i that transforms the state x at round i , the state being partitioned into two parts where the non-linear layer only operates on the part $x^{(0)}$ and not on the part $x^{(1)}$.

[0042] Such design allows efficient masking and thus can improve security against side-channel attacks. An example of this design is known as LowMC. Its aim is to minimize the multiplicative complexity and depth of the cipher in order to have performance advantages in certain applications, including multiparty computation (MPC), fully homomorphic encryption (FHE) and zero knowledge proofs (ZK). After a few tweaks due to security concerns, the current version of LowMC remains solid after several third-party analyses.

[0043] Compared to a full non-linear layer, a partial non-linear layer inevitably weakens the security of a cipher. One notable property is that there will exist non-trivial differential characteristics that will not activate any substitution-boxes (Sboxes) over one or more rounds of the cipher. In a single round, by setting the difference on $x^{(0)}$ to be 0, there are 2^{n-s} differences of x that do not differentially activate any Sboxes. Assuming a well-designed linear layer with good mixing properties, one can still expect around 2^{n-2s} differences that will also not differentially activate any Sboxes in the second round. This reasoning can be continued until no difference survives and thus the maximal expected number of rounds that a deterministic differential characteristic can cover is $\left\lfloor \frac{n}{s} \right\rfloor$. Note that this number would of course vary depending on the specificities of the linear layers.

[0044] The signature of a conventional block cipher can be described as $E: \{0,1\}^k \times \{0,1\} \rightarrow \{0,1\}^n$ where an n -bit plaintext is encrypted to an n -bit ciphertext using a k -bit secret key. A tweakable block cipher accepts an additional t -bit public input called tweak, its signature thus being $E: \{0,1\}^k \times \{0,1\}^t \times \{0,1\}^n \rightarrow \{0,1\}^n$. The introduction of a tweak input provides the ability for the user to select a permutation among a family of permutations even when the key is fixed.

[0045] Due to this extra degree of freedom that can potentially be leveraged by the attacker, designing a TBC is not straightforward. Block cipher-based TBC

constructions have been studied, but comes with a non-negligible efficiency penalty. We can mention the TWEAKEY framework, a recent design strategy to build ad-hoc TBCs, has been proposed. In this framework, the key and tweak inputs are treated equivalently in terms of design and this material is called tweakkey: the tweakkey input can be used as key or tweak value, which is up to the choice of the user.

[0046] Unlike the key input, the tweak does not need to be kept secret and therefore one should assume that an adversary has full control over it. Thus, besides the attack models of single-key (no difference in the key or tweak), related-key (difference in the key, but no difference in the tweak), related-tweak (no difference in the key, but difference in the tweak) and related-tweakkey (difference in both the key and tweak), it is reasonable to consider the chosen-tweak model as a meaningful model in practice.

[0047] An extendable-output function (XOF) is a generalization of a hash function, where the output can be extended to any desired length. Similar to a hash function, it should be collision, preimage and second-preimage resistant. A XOF is a natural choice when an application requires a hash function to have nonstandard digest length. Technically, it is also possible to use a XOF as a generic hash function by setting the output length fixed. Besides, it has some other applications, such as key derivation functions and stream ciphers. Currently, there are many instances of XOF, such as SHAKE128 and SHAKE256 (defined in the SHA-3 standard) and the more efficient variant KangarooTwelve.

[0048] Differential and linear cryptanalysis are among the most efficient and well-understood attacks against block ciphers, both in theory and in practice. Even though block ciphers embedding backdoors have been proposed, their design methodologies are usually very dedicated. On the other hand, as the topic of backdoor ciphers has not drawn much attention from the cryptography community, the backdoor security of these ciphers has not been well analyzed yet. Considering the above facts, we introduce a new framework which allows to build efficient backdoors based on differential cryptanalysis. Moreover, we will show that the backdoor security can be reduced to a variation of the collision resistance notion of the XOF used in the tweak schedule.

[0049] FIG.2 depicts a tweakable block cipher framework according to an embodiment of the present invention. The framework 250 is configured to build a tweakable block cipher with n -bit block size, k -bit key and tweak of arbitrary size. It consists of three components: a key expansion algorithm 252 which implements a key schedule, a set of round functions 254 and an extendable-output function (XOF) 256 which implements a tweak schedule. The key expansion algorithm 252 uses a key schedule to expand a secret key K into a set of subkeys k_i . The extendable-output function 256 expands the tweak T into a set of sub tweaks t_i . The round functions 254 are denoted f_i and have a partial non-linear layer, which can be expressed as $f_{i(x)} = L_i(S_i(x^{(0)})||x^{(1)})$. The framework 250 takes plaintext P as an input and the application of the i^{th} round function, the internal state is x_i . The state x_{r+1} after application of the final (r^{th}) is the ciphertext.

[0050] The sub-tweak and sub-key values are XORed only to the non-linear part of the state, but are XORed to full state at the whitening stage. The cipher is composed of r consecutive rounds.

[0051] The backdoor introduced by the framework is implemented by related-tweak differential characteristics with probability 1 (deterministic). With the knowledge of this backdoor, a key recovery attack can be performed using various methods of differential cryptanalysis. It is to be noted that the attack is under the chosen-tweak model: both the designer and the attacker have complete freedom over the tweak values. This model is classical for TBC and realistic in practice.

[0052] We now describe how the backdoor can be embedded in the cipher. The core idea is that the sub-tweak difference of the backdoor chosen tweaks is used to cancel the difference of the non-linear part of the state in each round, so that the resulting differential characteristics will have no differentially active Sbox.

[0053] FIG.3 shows transitions of state difference in the embedded related-tweak differential characteristic in a framework according to an embodiment of the present invention. The rectangles represent the internal states. S_i is the non-linear layer of the round function, and L_i is the linear layer. The differential characteristic propagates

in the following way: The difference of the first internal state 262 is XORed with the sub-tweak Δt_{i-1} , the resultant internal state 264 then goes through the non-linear layer S_i , then the resultant internal state 266 goes through the linear layer L_i , the resultant internal state 268 then goes through another XOR operation with the sub-tweak Δt_i , finally, the resultant internal state 270 goes through another nonlinear layer S_{i+1} . To give the internal state 272. The differences of the hashed blocks can be zero or non-zero, while the differences of the white blocks are necessarily zero.

[0054] In the method 500 described below with reference to FIG.5 , we present the general steps to construct an instance of the framework, in which a deterministic differential characteristic over r_0 ($\leq r$) rounds is embedded.

[0055] FIG.4 is a block diagram showing a block cipher generation system according to an embodiment of the present invention. The block cipher generation system 100 is a computer system with memory that stores computer program modules which implement block cipher generation methods according to embodiments of the present invention.

[0056] The block cipher generation system 100 comprises a processor 110, a working memory 112, an output module 114, and program storage 120. The processor 110 may be implemented as one or more central processing unit (CPU) chips. The program storage 120 is a non-volatile storage device such as a hard disk drive which stores computer program modules. The computer program modules are loaded into the working memory 112 for execution by the processor 110. The output module 114 is an output device which allows output of tweakable block ciphers, indications of tweaks and other data generated by the block cipher generation system 100.

[0057] The program storage 120 stores an extendable-output function (XOF) module 122, a selection module 124, an evaluation module 126 and a round function generation module 128. The computer program modules cause the processor 110 to execute various block cipher generation methods which are described in more detail below. The program storage 120 may be referred to in some contexts as computer readable storage media and/or non-transitory computer readable media. As depicted

in FIG.4, the computer program modules are distinct modules which perform respective functions implemented by the block cipher generation system 100. It will be appreciated that the boundaries between these modules are exemplary only, and that alternative embodiments may merge modules or impose an alternative decomposition of functionality of modules. For example, the modules discussed herein may be decomposed into sub-modules to be executed as multiple computer processes, and, optionally, on multiple computers. Moreover, alternative embodiments may combine multiple instances of a particular module or sub-module. It will also be appreciated that, while a software implementation of the computer program modules is described herein, these may alternatively be implemented as one or more hardware modules (such as field-programmable gate array(s) or application-specific integrated circuit(s)) comprising circuitry which implements equivalent functionality to that implemented in software.

[0058] FIG.5 is a flowchart showing a method of constructing an instance of a tweakable block cipher with a backdoor consisting of one deterministic differential characteristic according to an embodiment of the present invention. The method 500 shown in FIG.5 is carried out by the block cipher generation system 100 shown in FIG.4.

[0059] In step 502, the XOF module 122 is executed by the processor 110 of the block cipher generation system 100 to select an extendable-output function (XOF) as the tweak schedule.

[0060] In step 504, the selection module 124 is executed by the processor 110 of the block cipher generation system 100 to choose a pair of tweak values. The pair of tweak values T_1, T_2 are chosen uniformly at random and have arbitrary length.

[0061] In step 506, the XOF module 122 is executed by the processor 110 of the block cipher generation system 100 to calculate pairs of sub-tweaks. A pair of sub-tweaks is calculated for each round of the block cipher by applying the XOF selected in step 502 to the pair of tweak values selected in step 504. Step 506 may be written as follows:

[0062] Calculate $t_0^1 || \dots || t_r^1 \leftarrow XOF(T_1)$ and $t_0^2 || \dots || t_r^2 \leftarrow XOF(T_2)$.

[0063] In step 508, the evaluation module 126 is executed by the processor 100 of the block cipher generation system 100 to calculate sub-tweak differences for each of the round of the block cipher. Step 508 is equivalent to evaluating the differences $\Delta t_i = t_i^1 \oplus t_i^2$ for all $i \in [0, \dots, r]$.

[0064] In step 510, the selection module 124 is executed by the processor 100 of the block cipher generation system 100 to set initial difference values. A plaintext difference $\Delta P = \Delta x_0$ is randomly selected for the linear part $x_0^{(1)}$ and the difference for the non-linear part is set as the sub-tweak difference for the initial round $\Delta x_0^{(0)} = \Delta t_0^{(0)}$.

[0065] In step 512, the round function generation module 128 is executed by the processor 100 of the block cipher generation system 100 to generate a round function for each round of the block cipher. For each layer, a round function f_i is determined with partial non-linear layers such that: given the input difference $(\Delta x_{i-1} \oplus \Delta t_{i-1})$, the output difference after f_i satisfies $\Delta x_i^{(0)} = \Delta t_i$. the non-linear layer of each round function is fixed, while a specific linear layer is generated for each round function.

[0066] In step 514, the output module 114 of the block cipher generation system 100 outputs the tweakable block cipher comprising r_0 -round related-tweak differential characteristic that is embedded into it (with related tweaks T_1 and T_2).

[0067] The tweakable block cipher itself is sent to users. The r_0 -round related tweak differential characteristics, which is a hidden property of the tweakable block cipher, is the backdoor and only the one tweak pair t_1 and t_2 or more tweak pairs, which are only known by the designer, can trigger the backdoor.

[0068] The key of the backdoor is the tweak set generating the particular sub-tweak differences and the plaintext difference used in the embedded differential characteristic. We will use the prefix malicious to denote them. We also note that it is possible to embed multiple differential characteristics simultaneously. Then, the key

recovery complexity will depend on the number of embedded differential characteristics and the cryptanalysis method.

[0069] We emphasize that the framework only focuses on the requirements of the cipher to embed a backdoor. However, a concrete instantiation would also have to take into account many other design principles so that the cipher could resist all state-of-the-art cryptanalysis as well as the attack against the backdoor described in the following section.

[0070] In this section, we will evaluate two particular aspects of the backdoor security: (1) the complexity for the attacker to find the embedded differential characteristics, (2) whether additional backdoors exist in the resulting primitives, and if so, what is the complexity to find them. Firstly, we will discuss the relation between the malicious tweak set and its corresponding plaintext difference. We consider in this article that the number of rounds for the embedded differential characteristic is publicly known. On the one hand, if the malicious tweak set is known to the attacker, then the corresponding sub-tweak differences can of course be computed. From these sub-tweak differences, he can obtain partial information about the state differences expected during the differential characteristic. Note that the embedded differential characteristic being deterministic indicates that the transformations of state differences are linear. Hence, by reversing the linear transformations, the malicious plaintext difference can eventually be recovered. That is, the leakage of the malicious tweak set reveals the malicious plaintext difference. On the other hand, if the malicious plaintext difference is known to the attacker, he can compute its transformation through the linear layer and obtain the required value for the sub-tweak difference such that it cancels the nonlinear layer difference (since the sub-tweak is only XORed to the non-linear part, there is only one such candidate), and continue this process in the following rounds. Eventually, the embedded differential characteristic will be revealed. However, it remains difficult to recover the actual malicious tweak set due to the XOF-based tweak schedule: given an embedded related-tweak differential characteristic, finding a tweak pair that leads to it through the XOF will be difficult. We define this new security notion as target-difference resistance:

[0071] **Definition 1 (Target-difference resistance).** A hash function H is target-difference resistant if it is hard to find two inputs x and y such that $H(x) \oplus H(y) = \Delta$, where Δ is a given non-zero constant.

[0072] To better understand target-difference resistance, we introduce the limited birthday problem.

[0073] **Definition 2 (The limited-birthday problem).** Let H be an n -bit output hash function that can be randomized by some input, such as the initialization vector (IV) or tweak and that processes any input message of fixed size m bits, where $m > n$. Let IN be a set of admissible input differences and OUT be a set of admissible output differences, with the property that IN and OUT are closed sets with respect to $\oplus$ operation. Then, for the limited-birthday problem, the goal of the adversary is to generate a message pair (x, y) such that $x \oplus y \in IN$ and $H(x) \oplus H(y) \in OUT$ for a randomly chosen instance of H .

[0074] Let 2^I and 2^O denote the sizes of IN and OUT respectively. The lower bound on the time complexity to find a solution for the limited-birthday problem is $\max\left(2^{\frac{n-O+1}{2}}, 2^{n-I-O+1}\right)$. The success probability here is about 0.63. If I is small, the complexity is $2^{n-I-O+1}$. However, even if I is very big, the complexity cannot be below $2^{\frac{n-O+1}{2}}$.

[0075] Target-difference resistance can be seen as a special case of the limited-birthday problem (as well as a generalisation of the classical collision resistance) where OUT is limited to a single value ($2^O = 1$) and IN is the full input space. Therefore, target-difference resistance has the same generic complexity as the classical collision resistance notion, that is the birthday bound $O(2^{n/2})$.

[0076] More generally, instead of the exact malicious tweak set, the attacker could try to find another tweak set whose sub-tweak differences are also the desired ones for the embedded differential characteristics. Yet, its complexity is still covered by the expected target-difference resistance of the XOF.

[0077] The above attack can possibly be applied to other plaintext differences. According to the construction of the framework, the size of the input (tweak) to the XOF can be arbitrary long and thus any output of the XOF can potentially be obtained. For instance, if SHAKE128 is used as XOF, it can produce at most 2^b output streams (b being the state size between absorbing and squeezing phases in the sponge construction). Hence the number of possible sub-tweaks values is bounded by 2^b , no matter how many rounds it covers, and the number of sub-tweak differences is accordingly bounded by a greater value $N (\geq 2^b)$. Thus, given a random plaintext difference and a certain number of rounds, if the size of the required sub-tweak differences for the deterministic $\text{rt} \setminus \text{dc}$ does not exceed $\log N$, then there will be a tweak pair matching the differential characteristic. We summarize this finding as follows:

[0078] **Property 1.** In addition to the embedded differential characteristics, there might exist other deterministic differential characteristics that would threaten the cipher security.

[0079] Consequently, we have to evaluate the security of the cipher with respect to all the potential deterministic differential characteristics, not only the planned ones. We consider a framework instance that has a key size of 128 bits and employs SHAKE128 as the tweak schedule. The security strength of SHAKE128 against collision attack is $\min(l/2, 128)$ bits, where l is the output length (or the length of the colliding part). In order to recover an r_0 -round deterministic differential characteristic, the attacker has to find a tweak pair whose sub-tweak differences are the desired ones. The total size of these sub-tweak differences is $n + s \cdot (r_0 - 1)$ bits and thus the generic attack complexity is $2^{\min((n+s \cdot (r_0-1))/2, 128)}$, which becomes 2^{128} when $(n + s \cdot (r_0 - 1))/2 \geq 128$. The analysis is similar for the case where the key size is 256 bits and SHAKE256 is employed. We define r' to represent the value of r_0 that turns this inequality into an equality:

$$(n + s \cdot (r' - 1))/2 = k$$

[0080] All the deterministic related-tweak differential characteristic smaller than r' rounds can be recovered with a complexity smaller than the actual key size. Therefore, in order to prevent these differential characteristics to weaken the cipher, r' must be taken into consideration when determining the number of rounds of the framework instance. Actually, these related-tweak differential characteristics will decay exponentially in the remaining rounds as the corresponding sub-tweak differences are basically random.

[0081] When designing a backdoor for block ciphers, the first question that comes into mind is probably what type of backdoor should be used. While some existing backdoor designs directly insert a backdoor inside Sboxes or some other parts of the round function, we found out that the additional input tweak capability of a tweakable block cipher could be a perfect carrier of the backdoor. Suppose that a tweakable block cipher has a special property only when it is initiated with very specific tweak values, while it performs normally for all the other tweak values, then this property could be used as a backdoor. Moreover, if the tweak size is large enough, finding these special tweak values could be as hard as finding the secret key in the ideal case. One straightforward example of the special property is to build related-tweak differential characteristics using these tweaks. In the following, we provide more in-depth explanations on the design choices in the framework.

[0082] When instantiating the framework, some security notions have to be taken into account. The first and most important one is the undiscoverability: an entity who does not know the backdoor should not have increased chances to break the cipher. This requires that the backdoor security has to be as high as the cipher security. Thus, the framework should provide a valid and solid security evaluation for the backdoor. Another important notion is the practicability of the backdoor, and we will aim to make it as efficient as possible.

[0083] We detail in the following how the components of the framework do follow these principles.

[0084] In the following paragraph the selection of the tweak schedule function is discussed. As the malicious tweak values are the backdoor, the main task of the tweak

schedule is to protect the malicious tweaks. According to the security analysis set out above, the backdoor security relies on the target-difference problem, where the attacker tries to find a tweak set whose sub-tweak differences are the desired ones. This notion is simply a variation of the classical collision resistance for a hash function, so we expect a good cryptographic hash function to naturally provide this resistance.

[0085] Since the framework is a generalized framework, the total number of rounds will vary according to the different instantiations, so does the length of the sub-tweaks. Hence, the output length of the tweak schedule is expected to be flexible. Besides, if the tweak schedule was designed specifically for each framework instantiation, it will render the backdoor evaluation much more difficult. Thus, for sake of simplicity of the analysis, it seems a better idea to make the tweak schedule uniform in the framework.

[0086] For all these reasons, a XOF seemed to be the best choice for our tweak schedule. The security of actual XOF functions such as SHAKE128 or SHAKE256 is rather well-analysed and it can provide many choices in terms of security level. XOF is a special variant of a hash function and it will be appreciated by those of skill in the art that other hash function may be used to build the cipher.

[0087] Next, the use of Partial Non-linear Layers is discussed. The probability of a differential characteristic is determined by the number of differentially active Sboxes. Hence, in order to embed an efficient backdoor based on a differential characteristic, the best case is that the differential characteristic activates no Sbox at all. This is obviously very unlikely to happen in the framework if the round functions are fully non-linear layers. Indeed, unless the related-key model is considered, a non-zero difference inserted in the plaintext would have to be cancelled by the first sub-tweak difference. However, when inserting differences in the tweak input, as the sub-tweak differences produced by the XOF will be random, they will force many active Sboxes in the subsequent rounds. Thus, it is unlikely for the framework to be able to embed a deterministic related tweak differential characteristic that covers more than a few rounds if full nonlinear layers are utilized. Of course, it is possible to construct a differential characteristic with limited number of active Sboxes, and this also leads to a backdoor, even though it is less efficient.

[0088] We have also tried to modify the framework such that the sub-tweak addition is not performed every round. For example, an r rounds deterministic related tweak differential characteristic can be realized by applying the tweak addition only once at the beginning, this is shown in FIG.6.

[0089] FIG.6 shows a variant of the framework in which the tweak differential characteristic is only applied once at the beginning. As shown in FIG.6 the sub-tweak difference Δt_0 could neutralize the plaintext input difference Δx_0 and the resulting zero difference would get through the r rounds with probability 1. However, this candidate has a potential disadvantage: for any tweak pair the attacker can always set the plaintext input difference to be equal to Δt_0 .

[0090] The above analysis shows that full non-linear layers seem not suitable for the framework. On the contrary, partial non-linear layers satisfy our requirements. As in that case the Sbox only applies to a part of the internal state, the round function is able to map a non-zero input difference to a non-zero output difference while no active Sbox is activated. In term of building deterministic differential characteristics, we only have to set the difference of the non-linear part of the internal state to be zero rather than the full state. This allows us to choose the linear transformation so that the output difference could satisfy the requirements from described above in relation to step 512 in FIG.5.

[0091] In the following section, we introduce a concrete instantiation of the framework, called LowMC-M, which is based on the family of block ciphers LowMC.

[0092] LowMC is a family of block ciphers based on SPN structure with partial non-linear layers. The parameters are flexible and we denote the block size by n , the key size by k , the number of Sboxes applied each round by m and the maximum allowed data complexity by d (d is the $\log_2$ of the allowable data complexity up to which the cipher is expected to give the claimed security). In order to reach the security claims, the number of rounds r is then derived from all these parameters using a round formula.

[0093] At the beginning of the encryption process, a key whitening is performed. The round function at round i consists of four operations in the following order:

SboxLayer. A 3-bit Sbox is applied in parallel on the $s = 3m$ LSBs of the state, while the transformation for the remaining $n - s$ bits is the identity.

LinearLayer(i). The state is multiplied in GF(2) with an invertible $n \times n$ binary matrix L_i which is chosen independently and uniformly at random.

ConstantAddition(i). The state is XORed with an n -bit round constant C_i which is chosen independently and uniformly at random.

KeyAddition(i). The state is XORed with an n -bit round key k_i . To generate k_i , the master key K is multiplied in GF(2) with an $n \times k$ binary matrix KL_i . This matrix is chosen independently and uniformly at random with rank $\min(n, k)$.

[0094] Round keys and constants in LowMC can be compressed due to the fact that the non-linear layer is partial.

[0095] In the round function, it is possible to exchange the order of consecutive linear operations. This is shown in FIG.7.

[0096] FIG.7 shows swapping of the order of operations to provide a simplified representation of LowMC block ciphers used in embodiments of the present invention. As shown in FIG.7, in an initial order 710 the sequence of operations starts with the KeyAddition operation 711 from the previous round, this is followed by the Sboxlayer 712, this is followed by the LinearLayer operation 713 from the current round, this is followed by the ConstantAddition operation 714 for the current round, and finally, this is followed by the KeyAddition operation 715 from the current round.

[0097] In an exchanged order 720, the KeyAddition operation 715 is moved to before the LinearLayer operation 713 while keeping the ConstantAddition operation 714 as the last step in round i . Then, the equivalent round key can be written as $k'_i = L_i^{-1}(k_i)$. We observe that the Sbox only operates on the first s bits of the state and does not change the rest of the $n - s$ bits. Thus, we split k'_i into $k_i^{(0)}$ and $k_i^{(1)}$, and we can move the addition of $k_i^{(1)}$ to the beginning of the round. Next, we observe that $k_i^{(1)}$ can move further up to be combined with k_{i-1} in the previous round to form a

combined key addition operation 731 which forms part of the final order 730. The remaining key addition operation 735 occurs after the SBoxLayer operation.

[0098] In general, if we start from the last round and iterate this procedure recursively until all the additions to the linear part have been moved to the beginning of the algorithm, we will end up with an equivalent representation where all the round keys are reduced to s bits apart from the whitening key. We remark that the same reasoning can be applied to the round constants. This optimized representation can also reduce the implementation cost of the key schedule. Since all transformations performed during the optimization are linear and since the key schedule is itself linear, these transformations can be composed with the key schedule in order to compute the new $3m$ -bit round keys directly.

[0099] We will directly use the simplified representation of LowMC described above as a starting point in our design, with a further modification: we move LinearLayer behind SboxLayer in every round.

[0100] LowMC-M is a family of tweakable block ciphers built upon LowMC with an additional transformation in each round:

a. ***TweakAddition(i)*** The non-linear part of the state is XORed with an s -bit sub-tweak t_i just after $\backslash\text{lowak}$. t_i is generated from a XOF whose input is the original tweak value T .

[0101] The XOF is based on SHAKE128 or SHAKE256, depending on the key size. All the other transformations of the round function are the same as for LowMC. The operations making up the round function of LowMC-M are shown in FIG.8.

[0102] FIG.8 shows the operations making up a round function in an implementation of a framework for generating a tweakable block cipher according to an embodiment of the present invention. As shown in FIG.8, the internal state of the cipher comprises a non-linear part 802 and a linear part 804. A substitution box (Sbox) layer 806 is applied to the non-linear part 802 of the internal state. Then a linear layer 808 is

applied. Then an XOR operation 810 is applied which corresponds to the tweak addition, key addition, and constant addition, where C_i is the constant, k_i is the sub-key and t_i is the sub-tweak.

[0103] The encryption starts with a key and tweak whitening and the sizes of k_0 and t_0 are both n . derivation formula for the number of rounds r is the same as for LowMC.

[0104] During a differential cryptanalysis, we denote by X_i the i -th round state difference before the LinearLayer transformation. Given a matrix L_i , we denote its j -th row by $L_i[j,*]$, and partition L_i into four sub-matrices:

$$L_i \begin{bmatrix} L_i^{00} & L_i^{01} \\ L_i^{10} & L_i^{11} \end{bmatrix}$$

[0105] where $L_i^{00} \in GF(2)^{s \times s}$, $L_i^{01} \in GF(2)^{s \times (n-s)}$, $L_i^{10} \in GF(2)^{(n-s) \times s}$, $L_i^{11} \in GF(2)^{(n-s) \times (n-s)}$. With this notation, L_i^{00} and L_i^{01} will map $X_i^{(0)}$ and $X_i^{(1)}$ to the non-linear part of the state, respectively. And L_i^{10} and L_i^{11} will map $X_i^{(0)}$ and $X_i^{(1)}$ to the linear part of the state, respectively.

[0106] There are many forms of differential cryptanalysis that can perform a key recovery attack, such as the impossible differential attack, the boomerang attack, etc. For LowMC-M, we use the plain version where the attacker can deduce full or partial information about the r -th round key from a differential characteristic over $r - 1$ rounds.

[0107] Since an $(r - 1)$ -round deterministic differential characteristic can only reveal the s -bit sub-key k_r of the r -th round, more deterministic differential characteristics should be added in order to eventually recover the full key. After k_r has been retrieved, the cipher can be reduced to $r - 1$ rounds and thus another s -bit sub-key k_{r-1} can be recovered from an $(r - 2)$ -round deterministic differential characteristic.

[0108] FIG.9 shows deterministic differential characteristics embedded into LowMC-M in an embodiment of the present invention.

[0109] Finally, assume that there are a total of a such deterministic differential characteristics embedded in LowMC-M (as shown in FIG.9, one on $r - 1$ rounds 902, one on $r - 2$ rounds 904, upto a on $r - 2$ rounds 906), then $a \cdot s$ sub-key bits can be recovered. As the key schedule is fully linear and each matrix inside the key schedule is generated independently and uniformly at random, it implies that one will recover $a \cdot s$ bits of information about the key by solving a system of linear equations. Therefore, at most $a = \lceil k/s \rceil$ deterministic differential characteristics are needed to recover the full key.

[0110] Now, we explain how to embed such differential characteristics into an instantiation of LowMC-M. The general procedure is given in FIG.5 above. The a malicious tweak pairs are chosen by the designer at the very beginning and the corresponding sub-tweak differences are computed. Then, the linear layer matrix L_i is generated along with the generation of the deterministic differential characteristics round by round.

[0111] Firstly, we explain how to generate the linear layer matrices. Note that in order to have a deterministic differential characteristic over i rounds, only the linear layer matrices of the first $i - 1$ rounds have to be specifically designed as the matrix L_i has no impact on the differences of the i -th round Sboxes. Assuming we have already embedded a deterministic differential characteristics over i rounds, then all the linear layer matrices of the first $i - 1$ rounds of LowMC-M have been fixed accordingly. If we plan to extend b ($b \leq a$) of the a deterministic differential characteristics by one more round, the matrix L_i should be specified.

[0112] Denote by SX_i the set of $X_i^{(1)}$ of those deterministic differential characteristics that will be extended in the next round. Here, SX_i refers to the b differential characteristics.

[0113] Since the non-linear state difference $X_i^{(0)}$ equals to zero for all the b differential characteristics, the set SX_i will determine the differential in the following round. Given the difference set SX_i , the output differences after the multiplication by the matrix L_i^{01}

should cancel the following sub-tweak differences so that the b differential characteristics will activate no Sbox in round $i + 1$. The generation of L_i is set out below.

[0114] The set $SX_i = (X_{i,1}^{(1)}, X_{i,2}^{(1)}, \dots, X_{i,b}^{(1)})$ and the sub-tweak differences $(\Delta t_i^1, \Delta t_i^2, \dots, \Delta t_i^b)$ for the b differential characteristics are taken as inputs and the matrix L_i is the output.

[0115] The following system of linear equations is solved for j from 1 to s and a solution of $x = (x_1, x_2, \dots, x_n)$ is randomly picked as $L_i^{01}[j, *]$:

$$\begin{pmatrix} X_{i,1}^{(1)}[1] & X_{i,1}^{(1)}[2] & \dots & X_{i,1}^{(1)}[n-s] \\ X_{i,2}^{(1)}[1] & X_{i,2}^{(1)}[2] & \dots & X_{i,2}^{(1)}[n-s] \\ \vdots & \vdots & \ddots & \vdots \\ X_{i,b}^{(1)}[1] & X_{i,b}^{(1)}[2] & \dots & X_{i,b}^{(1)}[n-s] \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-s} \end{pmatrix} = \begin{pmatrix} \Delta t_i^1[j] \\ \Delta t_i^2[j] \\ \vdots \\ \Delta t_i^b[j] \end{pmatrix}$$

[0116] Then the sub-matrices L_i^{00}, L_i^{10} and L_i^{11} are randomly selected and if L_i is full rank then L_i is returned.

[0117] Denote the $b \times (n - s)$ matrix by MX_i . We emphasize that the rank of MX_i should be $\min(b, n - s)$, otherwise the matrix equation above is likely to have no solution. In practice, b is always smaller than $n - s$ for a normal parameters set of LowMC-M. Thus, this requirement also means that the binary vectors of $X_i^{(1)}$ in SX_i should be linearly independent.

[0118] The complete process of generating an instance of LowMC-M is as follows:

- a. Select a different pairs of tweaks of any desired length and compute the corresponding sub-tweak differences in all rounds for each pair of tweaks.
- b. For each tweak pair, choose an n -bit value of the plaintext difference ΔP as the input difference for the embedded differential characteristics, while setting the first s bits of ΔP to be equal to $\Delta t_0^{(0)}$.

- c. For the a differential characteristics, compute $X_1^{(1)} = \Delta P^{(1)} \oplus \Delta t_0^{(1)}$ and if the binary vectors of SX_1 are not linearly independent, then go back to step (ii).
- d. For round i from 1 to $r - 2$:
- e. Generate the matrix L_i as set out above with SX_i and the corresponding sub-tweak differences as inputs. Starting from round $r - a + 1$, the number of deterministic differential characteristics decrements by 1 at every loop.
- f. Except for the last loop, compute the set of SX_{i+1} through the matrix multiplication of L_i . If the binary vectors of SX_{i+1} are not linearly independent, repeat this loop.
- g. Choose L_{r-1} and L_r independently and uniformly at random from all invertible $n \times n$ binary matrices.
- h. For all rounds i , choose KL_i independently and uniformly at random from all $n \times k$ binary matrices of rank $\min(n, k)$ and the round constants C_i as well.

[0119] The backdoor is the a malicious tweak pairs and the corresponding plaintext differences. With the knowledge of these related-tweak differential characteristics, the designer can recover the full key in a very short time. To create the a plaintext differences, the designer can firstly choose a random P , then compute $P_i = P \oplus \Delta P_i$ for $i \in \{1, \dots, a\}$.

[0120] We note the fact that for any non-zero probability differential (Δ_1, Δ_2) of LowMC-M Sbox, where $\Delta_1 \neq 0$ and $\Delta_2 \neq 0$, there is only one unordered pair of inputs/outputs of the Sbox satisfying the differential. If each plaintext difference is used only once in the attack, then two sub-key candidates will remain for each Sbox as we cannot determine which order of the input/output pair of the targeted Sbox should be in the attack. The wrong sub-key candidate can be filtered by repeating the attack with another pair of plaintexts of the same difference. By doing so, $a \cdot s$ bits of information of the key can be retrieved in the end. Later, the remaining $(k - a \cdot s)$ key bits, if they exist, can be brute forced. Finally, the key recovery requires $2(a + 1) + \max(k - a \cdot s, 0)$ encryptions and the data complexity is $2(a + 1)$.

[0121] Note that the bit length of $X_i^{(1)}$ is $n - s$. In order to ensure that the matrix equation is solvable, the number of differential characteristics that are embedded in LowMC-M should not be higher than $n - s$. Generally, this bound is much higher than the number of differential characteristics that is actually needed in a concrete instantiation. Last but not least, one may wonder why we chose different malicious tweak pairs for the a related-tweak differential characteristics (indeed using a single malicious tweak pair would work), but we recommend doing so for security reasons as we will explain below.

[0122] The design goal of LowMC-M is to keep the backdoor and the cipher secure, but also to ensure the efficiency of the key recovery using the backdoor. Based on these principles, we selected some instantiation parameters and we present them in FIG.10.

[0123] FIG.10 is a table showing a range of different parameters sets of LowMC-M instantiations. For each instantiation, the malicious tweak pair that triggers each embedded differential characteristic is unique. d is the $\log_2$ of the allowed data complexity, a is the number of differential characteristics embedded. Regarding the performances, we evaluated the corresponding LowMC used in the LowMC-M instances. The LowMC implementations we benchmarked are optimized for AVX2 instructions. Measurements were performed on an AMD EPYC 7401 running Ubuntu 18.04. We tested several instances and we observed that a single encryption generally costs around 10000 to 30000 cycles depending on the parameters, the block size (= key size) ranging from 128 to 256 bits.

[0124] In the following section, we will discuss the backdoor security of LowMC-M with respect to the notions of undetectability, undiscoverability, untraceability and practicability.

[0125] Firstly, we discuss whether a LowMC-M instance containing a backdoor is distinguishable from a random LowMC-M with no backdoor embedded. Since the only

difference between these two cases lies in the way the linear layer matrices are generated, we will investigate the properties of these matrices.

[0126] We now would like to show that all embedded differential characteristics must use distinct tweak pairs in order to maintain undetectability. Assuming there is a backdoored LowMC-M instance that is generated following the steps described in above and a total of a deterministic related-tweak differential are embedded, while only a' ($< a$) different tweak pairs are used during the generation phase. Let c_j denote the number of embedded differential characteristics triggered by the same tweak pair, with $j \in \{1, \dots, a'\}$. We will show that some dependency will exist in the linear layer matrices for the first i ($\leq r - a$) rounds, consequently some additional deterministic related tweak differential characteristics over the first i rounds can be recovered.

[0127] **Definition 3.** For a LowMC-M instance, A_i is the matrix of dimension $(i \cdot s) \times (n - s)$ defined as:

$$\begin{pmatrix} L_1^{01} \\ L_2^{01} \cdot L_1^{11} \\ L_3^{01} \cdot L_2^{11} \cdot L_1^{11} \\ \vdots \\ L_i^{01} \cdot L_{i-1}^{11} \cdot \dots \cdot L_1^{11} \end{pmatrix}$$

[0128] We remark that a malicious plaintext difference ΔP can be retrieved if the corresponding malicious tweak pair is provided: in order to have a deterministic differential characteristics all Sboxes must be differentially inactive (i.e. the input difference of each Sbox should be zero) and thus for a malicious tweak pair that takes any of the a' different values, recovering $\Delta P^{(0)}$ (the non-linear part of ΔP) is straightforward as it is equal to the sub-tweak difference $\Delta t_0^{(0)}$. After that, one just needs to retrieve the remaining part $\Delta P^{(1)}$. In order to have a deterministic differential characteristic over the first two rounds, $L_1^{01}(X_1^{(1)})$ should be equal to Δt_1 , where $X_1^{(1)} = \Delta P^{(1)} \oplus \Delta t_0^{(1)}$. To extend the differential characteristic to the third round, $L_2^{01} \cdot L_1^{11}(X_1^{(1)})$ should be equal to Δt_2 . Continuing this process until the i -th round, we can create a system of linear equations with $n - s$ binary variables:

$$\begin{pmatrix} L_1^{01} \\ L_2^{01} \cdot L_1^{11} \\ L_3^{01} \cdot L_2^{11} \cdot L_1^{11} \\ \vdots \\ L_i^{01} \cdot L_{i-1}^{11} \cdot \dots \cdot L_1^{11} \end{pmatrix} \cdot (X_1^{(1)}) = A_{i-1} \cdot (X_1^{(1)}) = \begin{pmatrix} \Delta t_1 \\ \Delta t_2 \\ \Delta t_3 \\ \vdots \\ \Delta t_{i-1} \end{pmatrix}$$

[0129] Solving the above equation will output the solution of $X_1^{(1)}$, then the remaining part $\Delta P^{(1)}$ can be recovered naturally. However, there may be more solutions as the number of solutions is determined by the rank of A_{i-1} .

[0130] In cases where the number of rounds i is large enough such that $(i - 1) \cdot s \gg (n - s)$, if all the linear layer matrices are chosen independently and uniformly at random, the rank of A_{i-1} will be $n - s$ with very high probability. However, for a LowMC-M instance with backdoor embedded, since the linear layer matrices are specially designed, the rank of A_{i-1} can not be determined similarly.

[0131] Determining the Rank of A_{i-1} : We first introduce the following definition.

[0132] **Definition 4.** If M is an $n \times m$ binary matrix and v is an n -bit vector, the solution space $\text{sol}(M, v)$ is defined as: $\text{sol}(M, v) = \{x^T \in \{0,1\}^m : Mx = v\}$.

[0133] Assume that a special LowMC-M instance is generated with c related-tweak deterministic differential characteristics over i rounds while only one malicious tweak pair is used. During the generation of L_j^{01} , $j \in \{1, \dots, i - 1\}$, could be simplified as:

$$MX_j \cdot x = \vec{1} \text{ or } MX_j \cdot x = \vec{0}$$

[0134] $\vec{0}$ and $\vec{1}$ are c -bit vectors full of zeros and ones, respectively.

[0135] Denote by V the union of $\text{sol}(MX_1, \vec{1})$ and $\text{sol}(MX_1, \vec{0})$, the rows of L_1^{01} are chosen from V . Since the dimensions of $\text{sol}(MX_1, \vec{1})$ and $\text{sol}(MX_1, \vec{0})$ are both $n - s - c$, then the dimension of V is $n - s - c + 1$. When $j = 2$, the above equations can be represented by:

$$MX_1 \cdot (L_1^{11})^T \cdot x = \vec{1} \text{ or } MX_1 \cdot (L_1^{11})^T \cdot x = \vec{0}$$

[0136] because $X_2^{(1)} = L_1^{11} \cdot X_1^{(1)}$. The rows of L_2^{01} are chosen from $\text{sol}(MX_1 \cdot (L_1^{11})^T, \vec{1})$ or $\text{sol}(MX_1 \cdot (L_1^{11})^T, \vec{0})$. Before we continue, we will use the following lemma.

[0137] **Lemma 1.** Let M_1 and M_2 be two binary matrices of dimension $(n \times m)$ and $(m \times m)$ respectively. If $x \in \text{sol}(M_1 \cdot M_2, v)$, then $x \cdot M_2^T \in \text{sol}(M_1, v)$ for any n -bit vector v .

[0138] *Proof.* For any $x \in \text{sol}(M_1 \cdot M_2, v)$, we have $(M_1 \cdot M_2) \cdot x^T = v$. It can be represented by $M_1 \cdot (M_2 \cdot x^T) = v$, thus $(M_2 \cdot x^T)^T = x \cdot M_2^T \in \text{sol}(M_1, v)$.

[0139] According to Lemma 1, if $x \in \text{sol}(MX_1 \cdot (L_1^{11})^T, \vec{1})$, then $x \cdot L_1^{11} \in \text{sol}(MX_1, \vec{1})$ and also if $x \in \text{sol}(MX_1 \cdot (L_1^{11})^T, \vec{0})$, then $x \cdot L_1^{11} \in \text{sol}(MX_1, \vec{0})$. Thus, all the rows of $L_2^{01} \cdot L_1^{11}$ are in the space V . Similarly, we can get the same results for $L_3^{01} \cdot L_2^{11} \cdot L_1^{11}, \dots, L_{i-1}^{01} \cdot L_{i-2}^{11} \cdot \dots \cdot L_1^{11}$. To summarize, all the rows of A_{i-1} for this special LowMC-M instance are chosen from the space V of dimension $n - s - c + 1$. Thus, the rank of A_{i-1} is $n - s - c + 1$. Let us return back to the previous LowMC-M instance mentioned at the beginning of this subsection. We can divide the a differential characteristics into a' sub-groups where each sub-group includes c_j differential characteristics that are triggered with the same tweak pair, $j \in \{1, \dots, a'\}$. Then, the space V will be the intersection of all the spaces that are determined by the a' sub-groups. We summarize the result as follows.

[0140] **Proposition 1.** If there is a total of a' different malicious tweak pairs and each of them is used to build c_j deterministic differential characteristics over i rounds in an instance of LowMC-M, with $(i - 1) \cdot s \gg (n - s)$, then the rank of A_{i-1} will be $n - s - \sum_{j=1}^{a'} (c_j - 1)$.

[0141] As a result, the rank of A_{i-1} is $n - s - \sum_{j=1}^{a'} (c_j - 1)$ and a total of $2^{\sum_{j=1}^{a'} (c_j - 1)}$ deterministic differential characteristics for each of the a' tweak pairs can be recovered by the designer. Note that the rank of A_{i-1} can be easily computed by any entity. Compared to the full rank A_{i-1} for a random LowMC-M with no backdoor embedded, the unusual property of A_{i-1} for the backdoored $\backslash\text{mm}$ will uncover the existence of the backdoor if $a' < a$. However, if $a' = a$, that is, $c_j = 1$ for all $j \in \{1, \dots, a'\}$, then A_{i-1} will be full rank. Therefore, in order to keep the backdoor of LowMC-M undetectable, we recommend to not use the same tweak pair for building more than one differential characteristic in the generation phase.

[0142] In this subsection, we discuss whether the backdoor from a LowMC-M instance can be efficiently recovered by an attacker. Recall that some unknown deterministic related-tweak differential characteristics potentially exist in LowMC-M, according to Property 1. Instead of considering the embedded backdoor exclusively, we evaluate the complexity of finding any useful deterministic related tweak differential characteristics for an attacker. Basically, the complexity is based on the XOF security properties.

[0143] We simply adopt the security analysis for the general framework. For any LowMC-M instance, the bound r' is much smaller compared to the total number of rounds, which poses no threat to the backdoor.

[0144] FIG.11 is a table showing results of backdoor security evaluation. The backdoor security evaluation for LowMC-M- n/s with block size n , key size n , non-linear layer size s and $\log_2$ data complexity 64. r is the actual number of rounds of the instance, r' is in the formula above and the definition of r'' is set out below.

[0145] We can examine the undiscoverability security from another perspective. Note that deterministic related-tweak differential characteristics can be derived as long as $X_1^{(1)}$ is solvable. The requirement for the equation to be solvable is that the ranks of the coefficient matrix A_{i-1} and the augmented matrix are equal, which means that the vector on the right side of the equation, denoted as v , has to be a combination of the columns of A_{i-1} . Observe that the number of such combinations is 2^α , α being the rank

of A_{i-1} and it can be computed according to Proposition 1. As for vector v , it is random due to the XOF and its size is $s \cdot (i - 1)$. In conclusion, $X_1^{(1)}$ is solvable with probability $2^{\alpha - s \cdot (i-1)}$, that is, the complexity of finding an i -round deterministic related-tweak differential characteristics is $2^{s \cdot (i-1) - \alpha}$. We define r'' to represent the value of i that turns the complexity to be equal to the key space size:

$$r'' = \frac{k + \alpha}{s} + 1$$

[0146] The maximal value is $r'' = \frac{k+n}{s}$ when A_{i-1} is full rank of $n - s$. Still, r'' is much smaller than the number of rounds of any LowMC-M instance as can be seen from FIG.11.

[0147] To summarize, the backdoor and the other potential deterministic related-tweak differential characteristics of the same length are fully protected by the XOF, and its recovery is as hard as brute forcing the key.

[0148] As for practicability, only negligible data and computation are required to launch a full key recovery attack with the knowledge of the backdoor, as explained above. Thus, the full key can be recovered within seconds. Since the usage of the backdoor requires chosen tweaks, the malicious tweaks can be detected by the user once the designer makes queries to attack him, which means the backdoor is traceable. Besides, as only a few queries are needed to launch an attack with the knowledge of the backdoor, the user is able to quickly brute force the queries to find out the malicious tweak pairs.

[0149] In the following section, we study the security of LowMC-M as a tweakable block cipher.

[0150] In comparison to LowMC, an additional tweak addition is introduced in LowMC-M. Theoretically, this feature will provide extra degrees of freedom for the attacker and might naturally weaken LowMC-M when compared to LowMC. However, since the tweak schedule is an XOF, the attacker cannot control its output. Even if the attacker

could brute force some structures on the sub-tweaks for a few rounds, this will result in the remaining rounds containing completely random structure, which consequently prevent the attacker utilizing these remaining rounds for what should have been the best attack on LowMC. Hence, we believe that the extra degrees of freedom provided by the tweak is not easily usable and will not lead to any important improvement over classical attack, including the existing cryptanalysis on LowMC.

[0151] All the current attacks on LowMC have been conducted under the assumption that the linear layer matrices of LowMC are chosen independently and uniformly at random. Except the tweak addition, LowMC-M has the equivalent specification to LowMC. The only difference lies in the way the linear layer matrices L_i are chosen during the generation phase. In order to prove that the security of LowMC-M is on par with that of LowMC, we need to show that the linear layer matrices of LowMC-M are indistinguishable from those of LowMC-M from the perspective of the attacker. We will evaluate this with respect to the randomness and independence.

[0152] The randomness of the linear layer matrix L_i is analyzed by scrutinizing its four sub-matrices one by one.

[0153] The two sub-matrices L_i^{00} and L_i^{10} of L_i are chosen independently and uniformly at random for each round. Even though L_i^{11} is chosen randomly, there is a supplementary requirement during the generation phase. That is, the binary vectors of SX_{i+1} have to be linearly independent, which adds an extra constraint to L_i^{11} since each binary vector of SX_{i+1} is obtained by:

$$X_{i+1}^{(1)} = L_i^{11} \cdot X_i^{(1)}$$

[0154] and thus the transformation of L_i^{11} should map a set of linearly independent vectors to another set of linearly independent vectors. Since L_i^{11} is chosen randomly and all the $X_i^{(1)}$ involved are linearly independent, every $X_{i+1}^{(1)}$ in SX_{i+1} can be regarded as random binary vectors and are independent from each other. On the other hand, note that at most $a = \lfloor k/s \rfloor$ differential characteristics are embedded in LowMC-M, which means that the size of SX_{i+1} is $\lfloor k/s \rfloor$ at most. For any reasonable parameter

set, we will have $[k/s] \ll (n - s)$. Based on Lemma 2 below, we can compute the probability that the set SX_{i+1} is linearly independent. As a result, the probability is almost 1, which is also verified from our experiments.

[0155] Hence, the constraint on L_i^{11} is very loose. The final selection of L_i^{11} will not introduce any special property.

[0156] **Lemma 2.** For m random n -bit vectors over F_2 ($m \leq n$), the probability that they are linearly independent is $p(m) = \prod_{k=0}^{m-1} (1 - 2^{k-n})$. In particular, $p(n) > 0.2887$.

[0157] L_i^{01} is the essential part for embedding backdoors, and thus it is the one specially designed. The row length of L_i^{01} is $n - s$ bits, while in the generation phase each row is chosen from a sub-space of dimension $n - s - b$ which is determined by the corresponding Equation above, b being the size of SX_i . However, we will show that for the attacker this special chosen L_i^{01} is still indistinguishable from a randomly chosen one.

[0158] Observe that both MX_i and the sub-tweak difference vector are unknown for the attacker, thus the solution space is unidentified. Moreover, the solution space for each row of L_i^{01} could be different due to the sub-tweak difference. Therefore, it is impossible for the attacker to trace some rows of L_i^{01} to the targeted hidden sub-space.

[0159] To summarize, the four sub-matrices are indistinguishable from random matrices for the attacker. The only connection between these four sub-matrices is that the combined matrix L_i should be invertible, which is also the same for LowMC, so it reveals no additional information. Hence, we conclude that for the attacker the matrix L_i is indistinguishable from a random matrix.

[0160] The definition of A_r captures partial information of the matrices that includes L_i^{10} and L_i^{11} over r consecutive rounds. If the linear layer matrices are chosen independently and uniformly at random, the resultant A_r should be random, thus the rank of A_r will be $n - s$ when $r \cdot s \gg (n - s)$. If the rank for a LowMC-M instance is smaller than $n - s$, it will imply a connection between these matrices.

[0161] As suggested in Proposition 1, the rank of A_r can be computed by $n - s - \sum_{i=1}^{a'} (c_i - 1)$. In order to eliminate the connections, each c_i should equal to 1, that is, different malicious tweak pairs should be used to build different differential characteristics during the generation phase.

[0162] The two sub-matrices L_i^{00} and L_i^{10} are chosen randomly and independently, so it will not impose any connection between the matrices.

[0163] We remark that even if there is some dependence existing between the linear layer matrices, the cipher security is still unlikely to be threatened. Yet, we conservatively recommend to avoid such dependency in a LowMC-M instance.

[0164] As described above, a framework for embedding backdoors into tweakable block ciphers is provided. The backdoor is a set of related-tweak differential characteristics with probability 1, from which the secret key can be recovered fully and efficiently. Besides, the backdoor security of our proposal is reduced to the target-difference resistance (a variant of the classical collision resistance, with the same generic complexity) of the XOF employed in the cipher.

[0165] Whilst the foregoing description has described exemplary embodiments, it will be understood by those skilled in the art that many variations of the embodiments can be made within the scope and spirit of the present invention.

CLAIMS

1. A method of generating a block cipher having a backdoor, the method comprising:
 - selecting a tweak schedule function as a tweak schedule for the block cipher;
 - choosing a set of tweak values;
 - calculating a set of pairs of sub-tweaks by applying the tweak schedule function to the set of tweak values, each pair of sub-tweaks of the set of pairs of sub-tweaks corresponding to a round of the block cipher;
 - calculating a sub-tweak difference for each round of the block cipher as a difference between the pair of sub-tweaks corresponding to that round of the block cipher;
 - generating a round function for each round of the block cipher, each round function having a layer comprising a non-linear part, wherein the non-linear part is configured with a differential characteristic such the output difference after a respective round of the block cipher is equal to the sub-tweak difference for the respective round of the block cipher; and
 - outputting the block cipher as the round functions for each round, the tweak schedule function and the set of tweak values, wherein the set of tweak values correspond to the backdoor.
2. A method according to claim 1, wherein the tweak schedule function is an extendable-output function.
3. A method according to claim 2, wherein the extendable-output function is based on a secure hash algorithm standard.
4. A method according to any preceding claim, wherein the layer comprising a non-linear part is a partial non-linear layer and the method further comprises selecting a plaintext difference as a difference in an initial round of the block cipher for a linear part of the non-linear layer and the sub-tweak difference for the initial round of the block cipher as the difference for the non-linear part of the non-linear layer.

5. A method according to any preceding claim, wherein the block cipher is based on a substitution–permutation network.
6. A method according to any preceding claim, wherein in each round of the cipher, a non-linear part of an internal state of the cipher is combined with a sub-tweak corresponding to the respective round in a tweak addition operation.
7. A method according to any preceding claim, wherein the set of tweak values comprises a plurality of pairs of tweak values.
8. A method according to claim 7, wherein each pair of tweak values of the plurality of pairs of tweak values differs from the each of the remaining pairs of tweak values.
9. A computer readable carrier medium carrying processor executable instructions which when executed on a processor cause the processor to carry out a method according to any one of claims 1 to 8.
10. A system for generating a block cipher having a backdoor, the system comprising a processor and a data storage device storing computer program instructions operable to cause the processor to:
 - select a tweak schedule function as a tweak schedule for the block cipher;
 - choose a set of tweak values;
 - calculate a set of pairs of sub-tweaks by applying the tweak schedule function to the set of tweak values, each pair of sub-tweaks of the set of pairs of sub-tweaks corresponding to a round of the block cipher;
 - calculate a sub-tweak difference for each round of the block cipher as a difference between the pair of sub-tweaks corresponding to that round of the block cipher;
 - generate a round function for each round of the block cipher, each round function having a layer comprising a non-linear part, wherein the non-linear part is configured with a differential characteristic such the output difference after a respective round of the block cipher is equal to the sub-tweak difference for the respective round of the block cipher; and

output the block cipher as the round functions for each round, the tweak schedule function and the set of tweak values, wherein the set of tweak values correspond to the backdoor.

11. A system according to claim 10, wherein the tweak schedule function is an extendable-output function.

12. A system according to claim 11, wherein the extendable-output function is based on a secure hash algorithm standard.

13. A system according to any one of claims 10 to 12, wherein the data storage device further stores computer program instructions operable to cause the processor to: select a plaintext difference as a difference in an initial round of the block cipher for a linear part of the non-linear layer and the sub-tweak difference for the initial round of the block cipher as the difference for the non-linear part of the non-linear layer.

14. A system according to any one of claims 10 to 13, wherein the block cipher is based on a substitution–permutation network.

15. A system according to any one of claims 10 to 14, wherein in each round of the cipher, a non-linear part of an internal state of the cipher is combined with a sub-tweak corresponding to the respective round in a tweak addition operation.

16. A system according to any one of claims 10 to 15, wherein the set of tweak values comprises a plurality of pairs of tweak values.

17. A system according to claim 16, wherein each pair of tweak values of the plurality of pairs of tweak values differs from the each of the remaining pairs of tweak values.

18. A communications device comprising an encryption module storing a tweakable block cipher and a shared secret key, the encryption module being configured to encrypt a plaintext message using the tweakable block cipher and the shared secret key, wherein the tweakable block cipher comprises a plurality of round functions, each

corresponding to a round of the block cipher, each round function having a layer comprising a non-linear part, wherein the non-linear part is configured with a differential characteristic such the output difference after a respective round of the block cipher is equal to a sub-tweak difference between a pair of sub-tweaks for the respective round of the block cipher.

19. A communications device according to claim 18, wherein the pairs of sub-tweaks for each respective round of the block cipher are obtainable by applying a tweak schedule function to a set of tweak values.

20. A communications device according to claim 19, wherein the tweak schedule function is an extendable-output function.

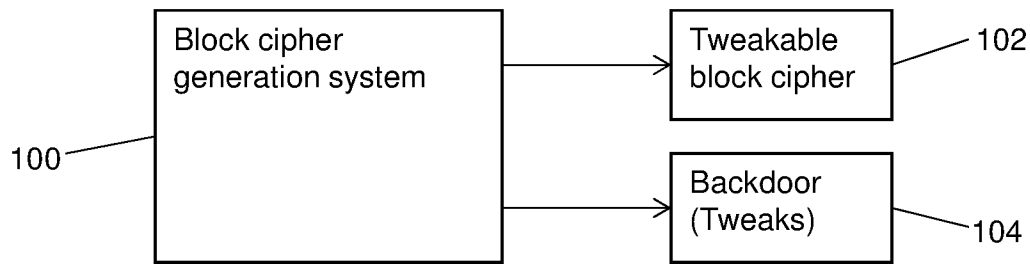


FIG. 1A

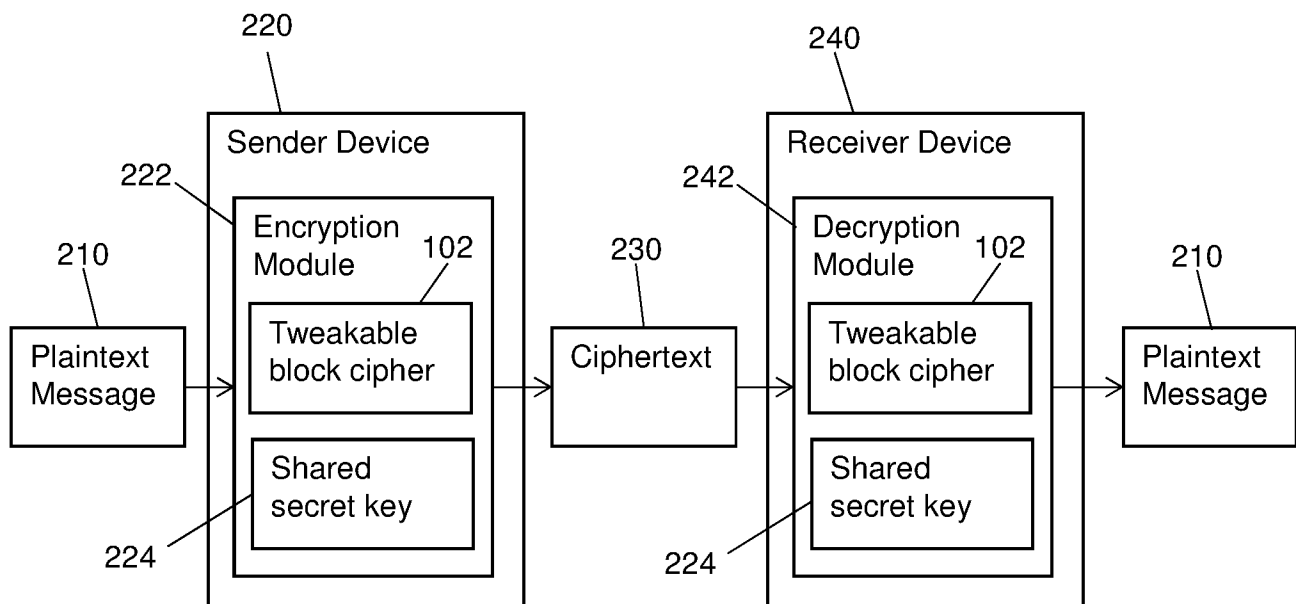


FIG. 1B

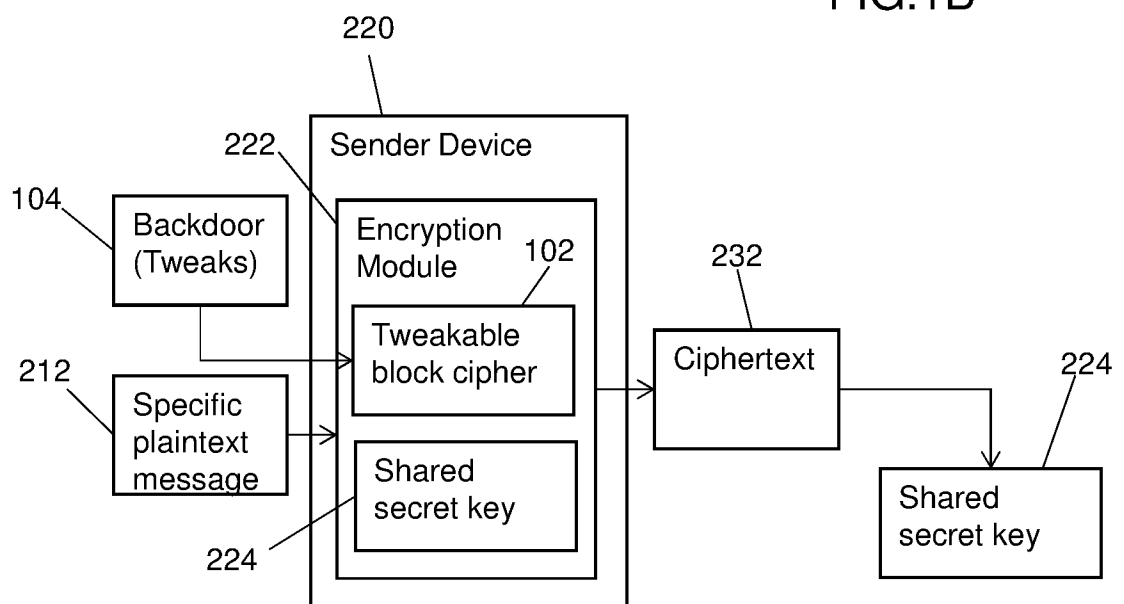


FIG. 1C

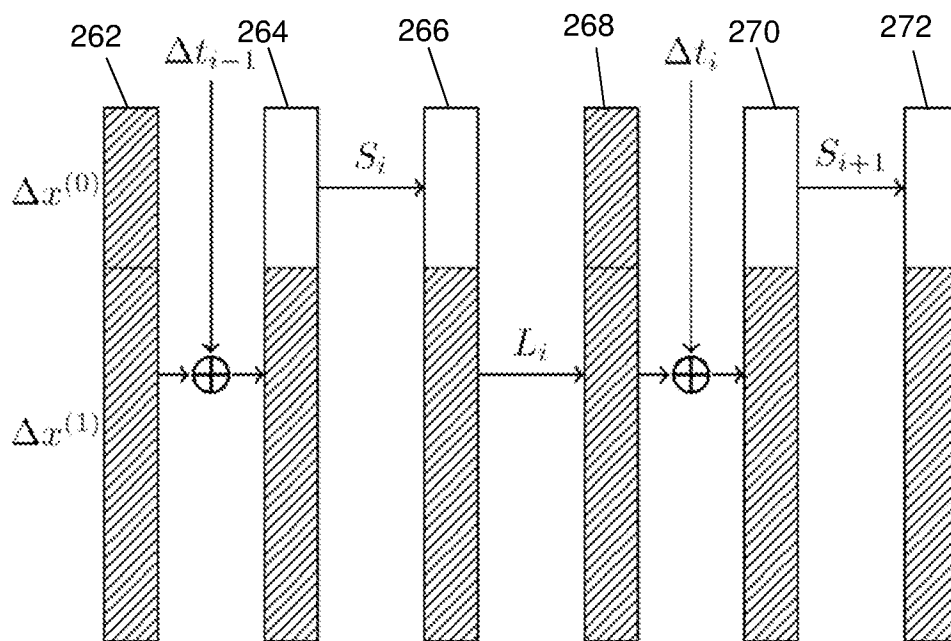
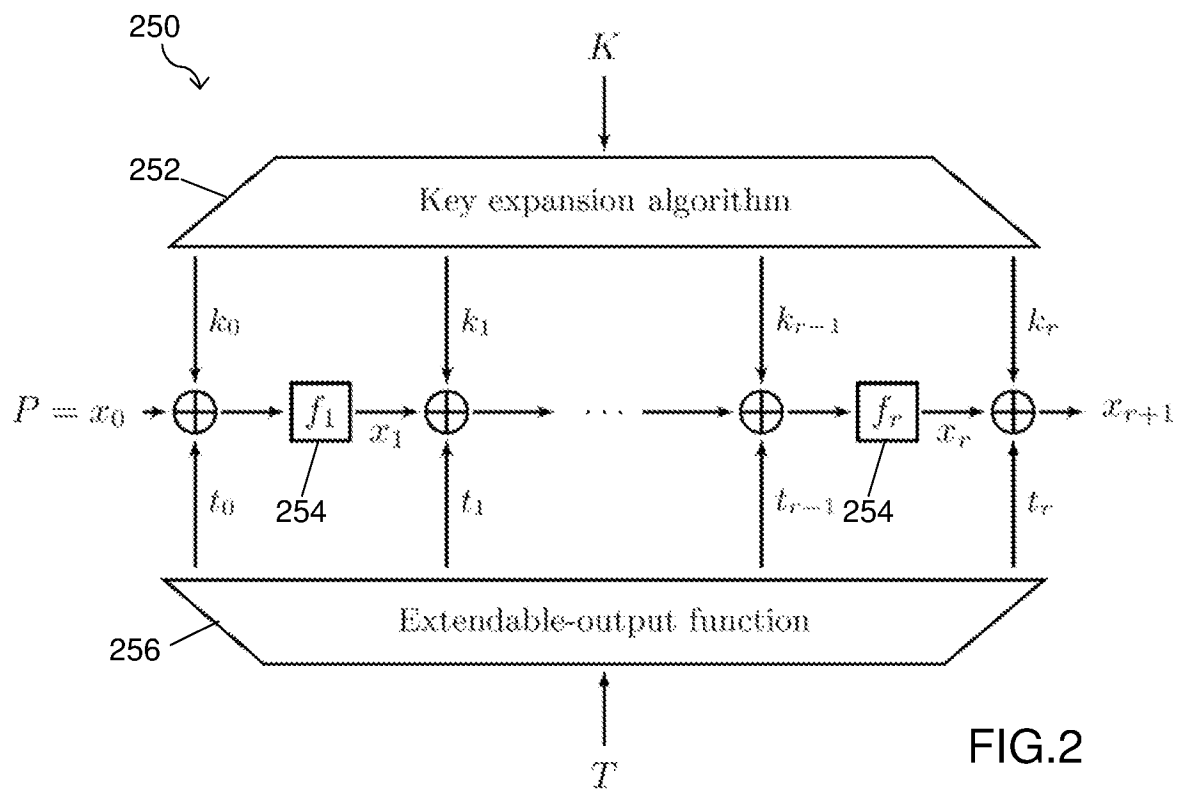


FIG.3

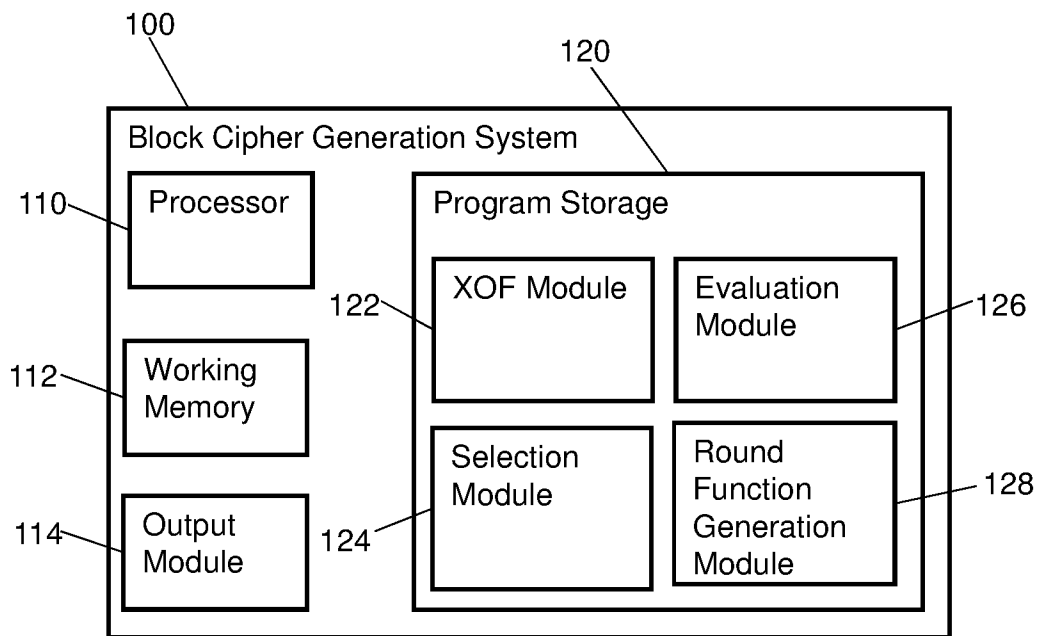


FIG.4

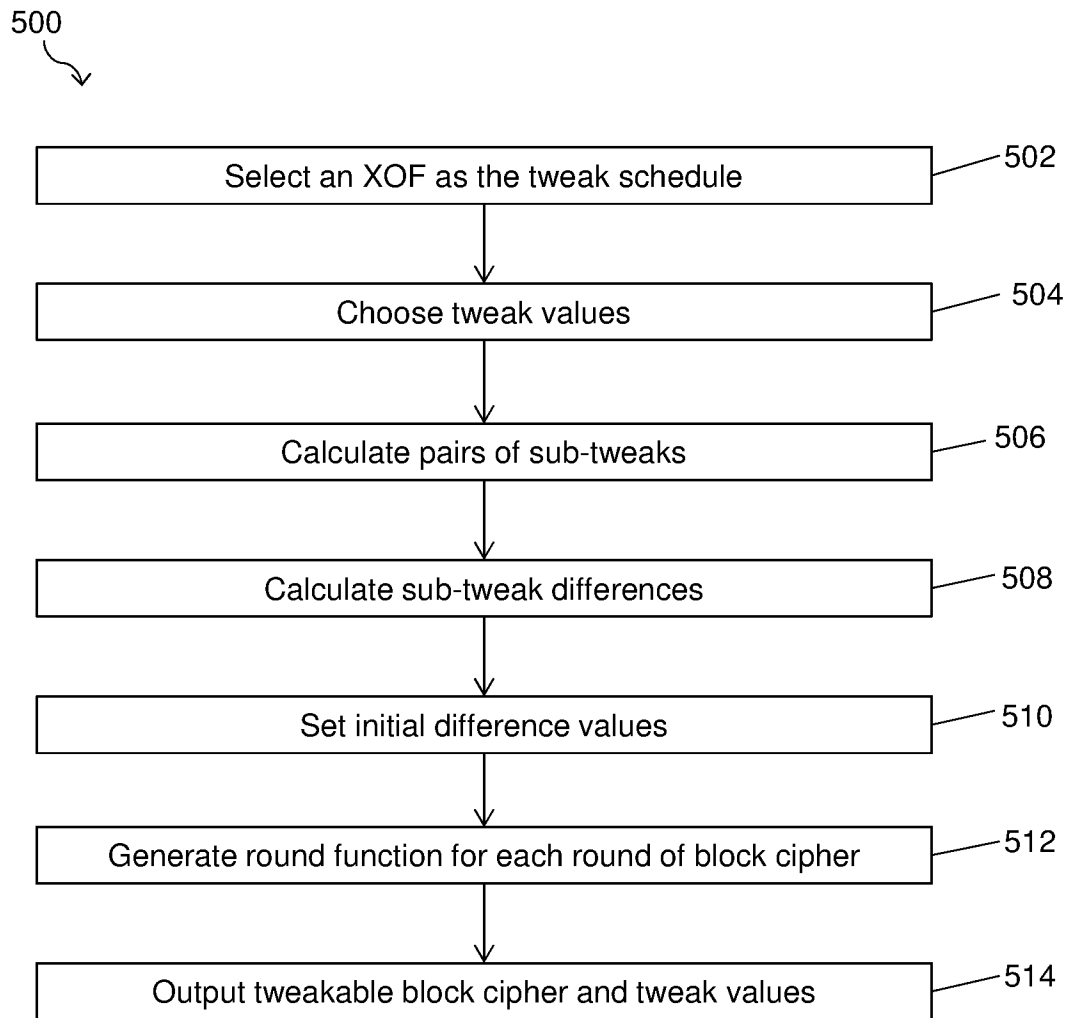


FIG.5

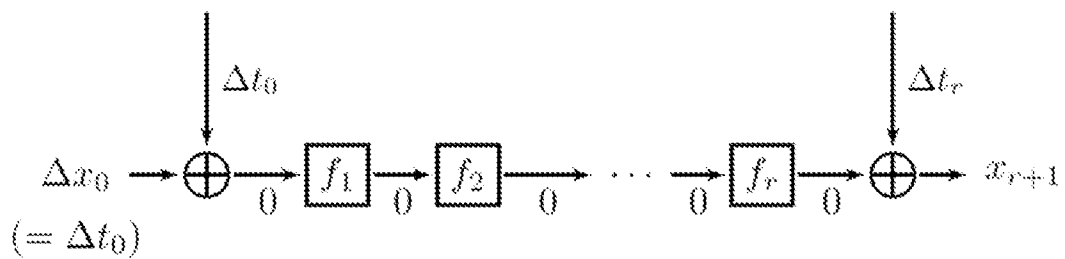


FIG.6

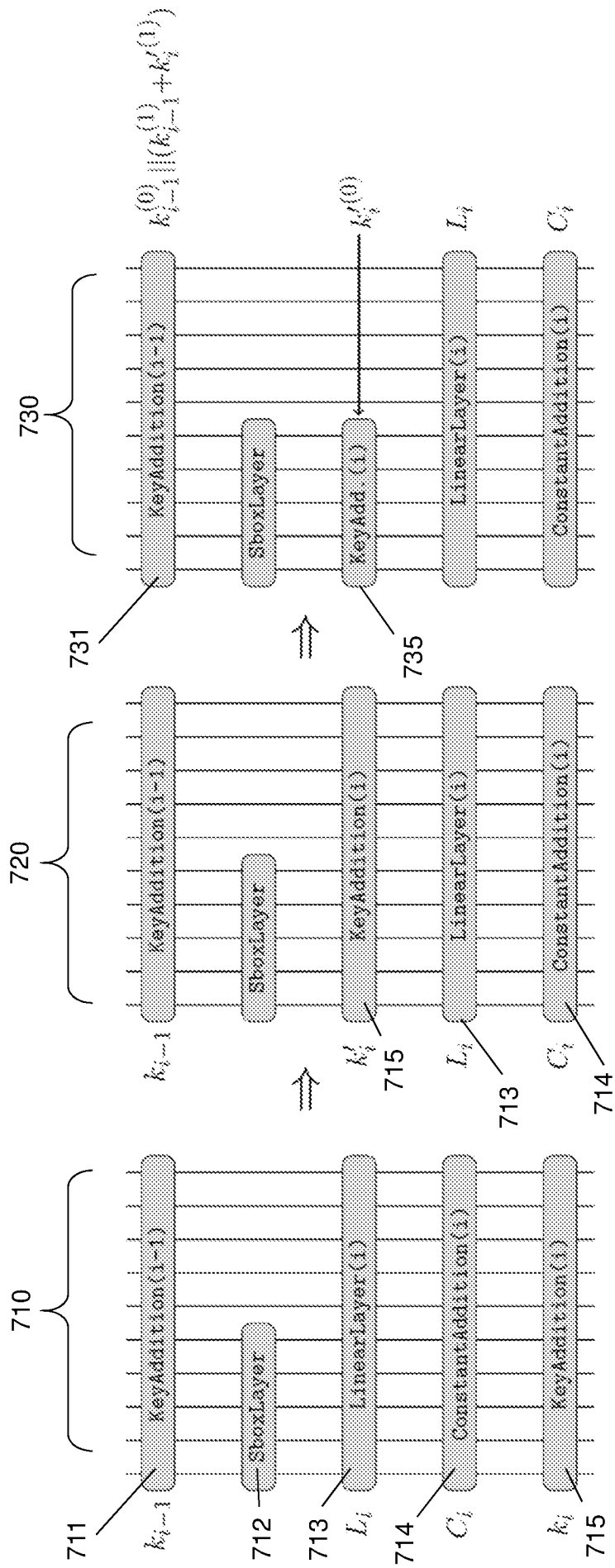


FIG.7

block size n	non-linear s	key size k	data d	rounds r	#differentials α	XOF
128	3	128	64	208	43	SHAKE128
	6	128	64	104	21	SHAKE128
	9	128	64	70	14	SHAKE128
	30	128	64	23	5	SHAKE128
	90	128	64	14	2	SHAKE128
256	3	256	64	384	85	SHAKE256
	9	256	64	129	28	SHAKE256
	60	256	64	21	5	SHAKE256
	120	256	64	14	3	SHAKE256

FIG.10

Parameters	r	r'	r''
LowMC-M-128/3	208	44	86
LowMC-M-128/6	104	23	43
LowMC-M-128/9	70	16	29
LowMC-M-128/30	23	6	9
LowMC-M-128/90	14	3	3
LowMC-M-256/3	384	87	170
LowMC-M-256/9	129	30	57
LowMC-M-256/60	21	6	9
LowMC-M-256/120	14	4	5

FIG.11

INTERNATIONAL SEARCH REPORT

International application No.

PCT/SG2021/050323

A. CLASSIFICATION OF SUBJECT MATTER

H04L 9/06 (2006.01) G09C 1/00 (2006.01)

According to International Patent Classification (IPC)

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

H04L, G09C, G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

Fampat, Internet: Tweakable block cipher, 可调分组加密, LowMC, backdoor, 后门, XOF, extendable-output function, Tweak schedule, sub tweak difference, state difference, differential, non-linear, 非线性 and related terms

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	CHAKRABORTI A. ET AL., Elastic-Tweak: A Framework for Short Tweak Tweakable Block Cipher. <i>IACR Cryptol. ePrint Arch.</i> , 30 April 2019, pages 1-36 [Retrieved on 2021-08-25] Abstract, Fig. 3.2, Sections 1.3, 3, 3.1, 3.2, 4.2	1-20
X	BOSSERT J. ET AL., Pholcos – Efficient Large-state Tweakable Block Ciphers from the AES Round Function. <i>IACR Cryptol. ePrint Arch.</i> , 2 March 2020, pages 1-31 [Retrieved on 2021-08-25] Abstract, Sections 2, 3, 5.4	1-20
A	BANNIER A. ET AL., Mathematical Backdoors in Symmetric Encryption System-Proposal for a Backdoored AES-like Block Cipher. <i>1st International Workshop on FORMal methods for Security Engineering</i> , 21 February 2017, pages 1-12 [Retrieved on 2021-08-25] Whole document particularly Section 1	-

☒ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

*Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"D" document cited by the applicant in the international application

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

26/08/2021

(day/month/year)

Date of mailing of the international search report

31/08/2021

(day/month/year)

Name and mailing address of the ISA/SG

Intellectual Property Office of Singapore
1 Paya Lebar Link, #11-03
PLQ 1, Paya Lebar Quarter
Singapore 408533

Email: pct@ipos.gov.sg

Authorized officer

Mao Xiaohong (Dr)

IPOS Customer Service Tel. No.: (+65) 6339 8616

INTERNATIONAL SEARCH REPORT

International application No.

PCT/SG2021/050323

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	DOBRAUNIG C. ET AL., Rasta: A cipher with low ANDdepth and few ANDs per bit. <i>Advances in Cryptology – CRYPTO 2018</i> , 19 August 2018, pages 1-36 [Retrieved on 2021-08-25] Whole document particularly Section 2.2	-
A	ALBRECHT M. R. ET AL., Ciphers for MPC and FHE. <i>Advances in Cryptology - EUROCRYPT 2015</i> , 14 April 2015, pages 430-454 [Retrieved on 2021-08-25] Whole document	-
A	WO 2019/163032 A1 (NEC CORP) 29 August 2019 & US 2021/021406 A1 Whole document of the patent family member	-

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/SG2021/050323

Note: This Annex lists known patent family members relating to the patent documents cited in this International Search Report. This Authority is in no way liable for these particulars which are merely given for the purpose of information.

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2019/163032 A1	29/08/2019	US 2021/0021406 A1	21/01/2021