How Secure is AES under Leakage

Andrey Bogdanov[1] and Takanori Isobe[2]

[1] Technical University of Denmark, Denmark. anbog@dtu.dk
[2] Sony Corporation, Japan. Takanori.Isobe@jp.sony.com

Abstract. While traditionally cryptographic algorithms have been designed with the black-box security in mind, they often have to deal with a much stronger adversary – namely, an attacker that has some access to the execution environment of a cryptographic algorithm. This can happen in such grey-box settings as physical side-channel attacks or digital forensics as well as due to Trojans.
In this paper, we aim to address this challenge for symmetric-key cryptography. We study the security of the Advanced Encryption Standard (AES) in the presence of explicit leakage: We let a part of the internal secret state leak in each operation. We consider a wide spectrum of settings – from adversaries with limited control all the way to the more powerful attacks with more knowledge of the computational platform. To mount key recoveries under leakage, we develop several novel cryptanalytic techniques such as *differential bias attacks*. Moreover, we demonstrate and quantify the effect of uncertainty and implementation countermeasures under such attacks: *black-boxed rounds, space randomization, time randomization, and dummy operations*. We observe that the residual security of AES can be considerable, especially with uncertainty and basic countermeasures in place.

Key words: Grey-box, side-channel attacks, leakage, AES, bitwise multiset attacks, differential bias attacks, malware, mass surveillance

1 Introduction

1.1 Background: black box, grey box and white box

It is symmetric-key algorithms that are in charge of bulk data encryption and authentication in the field. Plenty of multiple wide-spread applications such as mobile networks, access control, banking, content protection, and storage encryption often feature only symmetric-key algorithms, with no public-key cryptography involved.

Traditionally, the security of symmetric-key cryptographic primitives has been analyzed in the *black-box* model, where the adversary is mainly limited to observing and manipulating the inputs and outputs of the algorithm, the related-key model [2] being a notable extension. Multiple techniques have been extensively elaborated upon, such as differential and linear cryptanalysis, integral and algebraic attacks, to call a small subset of the cryptanalytic tools available today. Cryptographers have excelled at preventing those by design [8].

In late 1990s, with the introduction of timing attacks [13] by Kocher, differential fault analysis [1] by Boneh, DeMillo and Lipton, simple power analysis as well as differential power analysis [14] by Kocher, Jaffe and Jun, the research community has become aware of side-channel attacks that operate in the *grey-box* model: Now the attacker has access to the physical parameters of cryptographic implementations or can even inject faults into their execution. Numerous countermeasures have been proposed to hamper those attacks, providing a practical level of security in many cases.

Since mid 2000s, a trend of side-channel analysis has been towards analytical side-channel attacks that assume leakage of fixed values of variables instead of stochastic variables and whose techniques border the black-box cryptanalysis. So, collision attacks [22] by Shramm et al observe an equation within one or several executions of an algorithm. Algebraic side-channel attacks [21] by Renauld and Standaert work under the assumption that the attacker can see the Hamming weight of the internal variables of an algorithm. The attacker uses the techniques of algebraic cryptanalysis to solve the systems of nonlinear equations arising from collisions and algebraic side-channel attacks [5, 19, 20]. Dinur and Shamir [9] apply integral and cube attacks to block ciphers in a setting where a fixed bit after a round is leaked due to physical probing, power analysis or similar. Also differential fault analysis uses elements of differential cryptanalysis.

As an extreme development of the grey-box setting, the *white-box* model [7] by Chow et al poses the assumption that the adversary has full control over the implementation of the cryptographic algorithm. The major goal of white-box cryptography is to protect the *confidentiality of secret keys* in such a white-box environment. However, all published white-box implementations of standard symmetric-key algorithms such as AES to date have been practically broken in this model [18]. The white-box setting may be too strong for standard symmetric-key algorithms such as AES, because such a cipher was designed with the black-box security in mind.

1.2 Leakage and AES

In this paper, we enhance the Dinur-Shamir setting [9] and aim to bridge the gap between the physical side-channel attacks, the techniques of provable leakage resilience [17] and white-box setting (dealing with attackers too hard to protect against). Namely, we let the AES implementation leak some information during its execution which is defined as follows.

Definition 1 (Leakage model) *A malicious agent leaks a part of the intermediate internal secret state (including the key state) of a cryptographic algorithm in each algorithm execution.*

To apply this setting to AES (we will talk about AES-128 most of the time), we make it more concrete and fix several important parameters of the leak:

Frequency: There is a single leak per encryption/decryption. This simplifies complexity estimations in our analysis. If more leaks are available in each execution, the complexities can be adjusted accordingly.

Granularity: A leak can only happen after a full round. This situation corresponds e.g., to a 32-bit serial or round-based hardware implementation of AES or a software implementation using an instruction set extension such as AES-NI available on most Intel/AMD CPUs or the Cryptography Extension on ARMv8.

Knowledge: The attacker does not have any knowledge of the location of leaked bits, i.e., it does not know the bit position and the number of round of leaked bits. He also does not know whether the leak is from the key schedule or data processing part. This circumstance models the limited control of the adversary over the platform.

We let several parameters vary in our analysis[3]:

Time and space: The location of the leak in terms of the round number (time) and bit position within the round (space) can either be fixed or vary.

Known/chosen plaintext/ciphertext: We consider both known and chosen text models. In case of a passive attacker, we talk about the known text setting. Otherwise, the attacker is allowed to choose text.

Alignment: We consider single-bit leaks, byte leaks and multiple-bit leaks. While single-bit leaks are more likely to happen due to physical probing, byte leaks correspond more to software settings.

1.3 Our contributions

The contributions of this paper are as follows. The cryptanalytic results are also summarized in Table 1.

AES under basic leakage and bitwise multiset attacks. We develop a bitwise multiset attack, which exploits relations of sets of plaintexts and internal states, to evaluate the security of AES if the time and space of the leakage is fixed. Our attack utilizes a *bitwise multiset characteristic* which is an extension of Dinur-Shamir integral attacks [9]. Unlike their attacks, our attack is feasible even if an attacker does not have any knowledge of the location of leaked bits. See Section 2 and Table 1 for the details.

AES under leakage with space/time uncertainty and differential bias attacks. We let time, space or both be randomized. The *space randomization* makes the position of leaked bits random in each execution. The *time randomization* makes the round number of leaked bits random in each execution. A

[3] Further models are worth consideration as well. For instance, the *Dinur-Shamir model* of the side-channel cube attacks [9] can be seen as a special case of our leakage model, with the following differences: First, in the Dinur-Shamir model, the adversary knows the location of the leak. Second, the Dinur-Shamir model does not consider leaks of more that a single bit. Third, Dinur-Shamir do not allow for leaks from the key schedule. Finally, the time and location of a leak are fixed, while we allow for time and space uncertainty in our consideration.

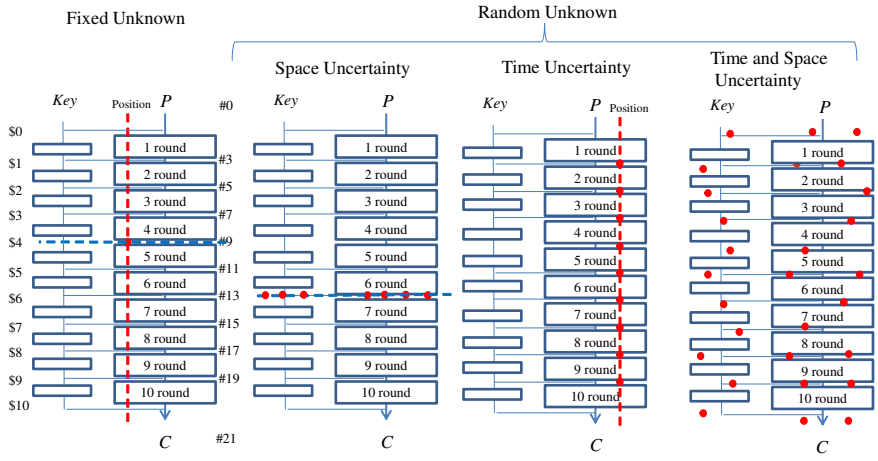

Fig. 1. AES with space and/or time uncertainty

combination of time and space randomization is also an advanced model we consider. See Figure 1 for an illustration.

This setting takes account of a more realistic environment, such as the lack of knowledge of the implementation, and the presence of countermeasures. Here, our multiset attacks are infeasible, as no clean multiset is available. To cope with that, we develop a *differential bias attack* and a *biased state attack* inspired by techniques for distinguishing attacks against stream ciphers [15, 16, 23]. More specifically, by properly choosing differences and values of plaintexts, we create *biased (differential) states*, where the distribution of bitwise differences or value is strongly biased only if the key is correctly guessed. Thus, we are able to distinguish the leak corresponding to the correct key. See Section 3 for the techniques as well as Section 4 and Table 1 for the results.

AES under noisy leakage. We consider leakage with noise, where the attacker does not know exactly if the variable it accesses corresponds to the execution of the algorithm under attack. For example, it can be the case if multiple instances of encryption (with different keys) are run simultaneously or if the implementation uses dummy operations to hide the AES execution. The differential bias attack remains applicable in this setting, with adjusted complexities. See Section 6 and Table 1 for details. To characterize noise, we define π to be the probability that the leak is correctly read. The complexities of our attacks grow quadratically with the increase of $1/\pi$.

Further results. We discuss the applicability of our attacks to AES-192 and AES-256, multiple-bit leakage, and other granularities of the leaks in Section 8.

Our observations and recommendations. To summarize the residual security of AES under leakage in the various settings, we observe the following.

Table 1. Security of AES-128 under leakage in various settings

Time and space of leaked bits	BB round(s)	Best attack	Bit alignment			Byte alignment		
			Time	Data	Section	Time	Data	Section
Fixed time/space	none	MA*2	2^{18}	2^{8} CC	Sec.3	2^{12}	2^{8} CC	Sec. 7.1
	round 9	MA*1	2^{44}	2^{34} CP	Sec.3	2^{42}	2^{34} CP	Sec. 7.1
	rounds 1, 9	MA	2^{47}	2^{8} CP	Sec.3	2^{44}	2^{34} CP	Sec. 7.1
Uncertain space	none	BSA	2^{26}	2^{26} CC	Sec.5.1	2^{23}	2^{23} CC	Sec. 7.2
	round 9	DBA	2^{48}	2^{42} CP	Sec.5.1	2^{41}	2^{39} CP	Sec. 7.2
	rounds 1, 2, 8, 9	DBA	2^{63}	2^{42} CP	Sec.5.1	2^{56}	2^{42} CP	Sec. 7.2
Uncertain time	none	BSA	2^{23}	2^{23} CC	Sec.5.2	2^{23}	2^{23} CC	Sec. 7.2
	round 9	DBA	2^{48}	2^{38} CP	Sec.5.2	2^{44}	2^{38} CP	Sec. 7.2
	rounds 1, 2, 8, 9	DBA	2^{61}	2^{37} CP	Sec.5.2	2^{53}	2^{37} CP	Sec. 7.2
Uncertain space and time	none	BSA	2^{33}	2^{33} CC	Sec.5.3	2^{24}	2^{24} CC	Sec. 7.2
	round 9	DBA	2^{58}	2^{46} CP	Sec.5.3	2^{47}	2^{43} CP	Sec. 7.2
	rounds 1, 2, 8, 9	DBA	2^{72}	2^{45} CP	Sec.5.3	2^{62}	2^{42} CP	Sec. 7.2
Random space and time w/ $\pi = 2^{-10}$	none	BSA	2^{53}	2^{53} CC	Sec.6	2^{44}	2^{44} CC	Sec. 7.2
	round 9	DBA	2^{68}	2^{56} CP	Sec.6	2^{57}	2^{53} CP	Sec. 7.2
	rounds 1, 2, 8, 9	DBA	2^{82}	2^{55} CP	Sec.6	2^{72}	2^{52} CP	Sec. 7.2
Random space and time w/ $\pi = 2^{-20}$	none	BSA	2^{73}	2^{73} CC	Sec.6	2^{64}	2^{64} CC	Sec. 7.2
	round 9	DBA	2^{78}	2^{66} CP	Sec.6	2^{67}	2^{63} CP	Sec. 7.2
	rounds 1, 2, 8, 9	DBA	2^{92}	2^{65} CP	Sec.6	2^{82}	2^{62} CP	Sec. 7.2

*1 : 32-bit partial key recovery attack, *2 : 8-bit partial key recovery attack
BB round(s): Black-boxed round(s), KP: Known Plaintext, CP: Chosen Plaintext
CC: Chosen Ciphertext, MA: Multiset Attack, DBA: Differential Bias Attack
BSA: Biased State Attack, π is the probability to read a correct leak

First, if no rounds are black-boxed and all intermediate internal states can be visible to the attacker, there are practical attacks, even with uncertain time and space. Second, to approach practical infeasibility of attacks in our leakage model without black-boxing, a substantial level of noise are be needed, $\pi = 2^{-10}$ and lower when combined with randomized time and space.

On the other hand, the black-boxing of round 9 is very effective. Indeed, if round 9 is black-boxed[4] (i.e., when the state between round 9 and round 10 is invisible to the attacker), the complexities of our attacks grow beyond 2^{44} even with fixed time and space. Third, if uncertainty in time and space is combined with the black-boxed 9th round, our attacks require more than 2^{58} operations, even with clean leaks. Then, if more rounds (1,2,8, and 9) are black-boxed, the complexities increase to 2^{72}. If noise is applied as countermeasure on top of that, it is possible to attain security levels of 2^{80} and beyond against our attacks.

Thus, black-boxed round 9, noise or both are needed to hamper our attacks at a practical security level under leakage. Note that a high-budget organization can practically afford an attack of complexity 2^{80} and higher [12]. However, the countermeasures considered here may still be effective against a mass surveillance attacker.

[4] E.g., partly unrolled hardware implementations aimed to reduce latency [6] may have this property.

2 Preliminaries

This section fixes AES notations that we will use throughout the paper and describes the leakage attack by Dinur-Shamir on AES as a starting point.

2.1 Notations of AES

AES is a block cipher with a 128-bit internal state and a 128/192/256-bit key K, referred to as AES-128, AES-192 and AES-256, respectively. In most parts of this paper, we refer to AES-128 whenever speaking of AES. The internal state is represented by a 4×4 byte matrix, and the key is represented by a $4\times4/4\times6/4\times8$ matrix. For example, a 4×4 internal state consisting of 16 byte cells is expressed as follows.

$$S = \begin{bmatrix} s_0 & s_4 & s_8 & s_{12} \\ s_1 & s_5 & s_9 & s_{13} \\ s_2 & s_6 & s_{10} & s_{14} \\ s_3 & s_7 & s_{11} & s_{15} \end{bmatrix}$$

AES consists of a data processing part and a key schedule. The data processing part adopts a substitution-permutation network whose round function consists of four layers: SubBytes, ShiftRow, MixColumns and AddRoundKey. SubBytes is a nonlinear transformation applying a 8-bit S-box to each cell. ShiftRow rotates bytes in row r by r positions to the left. MixColumns is a linear transformation applying a 4×4 diffusion matrix with branch number 5 to each column. AddRoundKey adds a 128-bit subkey to a 128-bit state by an XOR operation. Note that AddRoundKey is also performed before the first round as whitening and that MixColumns is omitted in the last round. Subkeys are generated by a key schedule. For the details of the key schedule of AES, we refer to [11].

Two types of internal states in each round of AES-128 are defined as follows: #1 is the state before SubBytes in round 1, #2 is the state after MixColumns in round 1, #3 is the state before SubBytes in round 2,..., #19 is the state before SubBytes in round 10, and #20 is the state after ShiftRow in round 10 (MixColumns is omitted in the last round). The states in the last round of AES-192 are addressed as #23 and #24, and of AES-256 as #27 and #28. We let #0 be a plaintext and #21, #25 and #29 be a ciphertext in AES-128/192/256, respectively. 128-bit subkeys are denoted as \$0 , \$1, ..., and so on. The i-th byte in the state x is denoted as x_i and the j-th bit in x_i is represented as $x_i[j]$.

2.2 Dinur-Shamir chosen-plaintext attack on AES-128 with leakage

As a starting point of our analysis, we outline the leakage attack proposed by Dinur and Shamir in [9]. As explained above, the Dinur-Shamir model is different from our leakage models as the adversary knows the time (round number) and space (bit position inside the round) of the leak, only single-bit leaks are considered there, and no leaks from the key schedule are allowed.

In the attack of [9], one uses the following multiset properties of a byte: In set A, all 2^8 values appear exactly once; In set C, all 2^8 values are fixed to a

constant; In set B, the XOR sum of all 2^8 values is zero; In set U, all 2^8 values is not A, C or B. Let an N-round attack be an attack based on leaked bits after the N-th round function, e.g., a 2-round attack is based on only leaked bits of #5.

In the first step, the attacker guesses 4 bytes of the key \$0, and chooses a set of 2^8 plaintexts, so that #2 consists of Λ-set in which only one byte is A and the other 15 bytes are C. If 4 bytes of \$0 are correctly guessed, #5 consists of 4 bytes of A and 12 bytes of C, while in a wrong key, all bytes in #5 become U. Thus, by checking whether the all 2^8 values of #5 are fixed, an attacker is able to sieve wrong keys after 2^{32} operations. The procedure can be repeated for three times with the three 4-byte sets of the key \$0 depending on the position of the leaked bit. The remaining 4 bytes of \$0 are exhaustively searched. Time complexity is estimated as 2^{42} ($\approx (2^{32} \times 2^8 \times 3)$) encryptions and the required data is 2^{34} ($= 2^{32} \times 4$) chosen plaintexts. The work [9] also proposes other types of 2-round attack using cubes, with a time complexity of 2^{35}. However, the details are not given.

The paper [9] mentioned that 3- and 4-round attacks were possible by using similar techniques but omitted the details. As A expands into all state after 3 rounds even if the key is correctly guessed, at least the 2-round attack has a limited application to 3 and 4 rounds.

3 AES under Leakage with Fixed Time and Space

In this section, we present new key recovery attacks on AES under leakage with fixed time and space. That is, a bit of the internal state is leaked whose location (round and bit position) is unknown but fixed for the entire attack. Our attack is an extension of the Dinur-Shamir integral attacks [9]. While their attack requires the location of leaked bits in advance, our attack is feasible even if an attacker does not have any knowledge of it. First, we describe a technique to detect whether leaked bits come from the key schedule or the data transformation, and show that leaked bits from the key schedule are of very limited use for a key recovery attack in this setting. Then we introduce key recovery attacks based on leaked bits from the data transformation. Our attacks utilize a bitwise multiset characteristic.

Formalization of fixed time and space. The fixed (unknown) location setting assumes that each execution of encryption leaks only *one bit* of the internal state at the fixed location. Specifically, leaked bits are assumed to come from internal states after each round function of the data processing part: $\#3, \#5, \ldots, \#19$ or each state of the key schedule (i.e., subkeys): \$0 , \$1, ..., \$10 at the *fixed* position of the *fixed* rounds in each encryption, e.g., $\#9_{11}[2]$ or $\$5_8[5]$. The adversary is able to access the encryption function with known/chosen plaintexts/ciphertexts and obtain corresponding leaked bits.

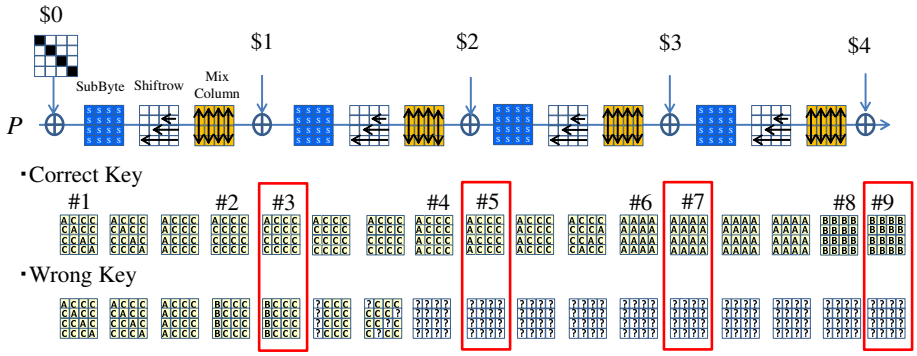

Fig. 2. Bitwise multiset characteristics over 4-round AES-128

Leakage from key schedule. The states in the key schedule, \$0 , \$1, . . ., \$10, are deterministic with respect to the value of the key, i.e., if a key is fixed, all states of the key schedule are fixed independently of the values of plaintexts. On the other hand, the states in the data processing part depend on the values of plaintexts. This difference allows us to detect whether leaked bits come from the key schedule. More specifically, we encrypt N different plaintexts and obtain N leaked bits. If all N bits are the same, they come from the key schedule with probability $(1 - 2^{-N})$.

If the leaked bits come from the key schedule, information theoretically, the attacker is able to get at most one bit of the subkey information, as each encryption leaks the same state information at the fixed location. In addition, since an attacker does not know where leaked bits come from, leaked information from the leaked bits is negligible. Therefore, we will focus on the case where leaked bits come from the data processing part in the following.

3.1 Bitwise multiset characteristic

Our attacks utilize the following bitwise multiset property in the data transform.

Proposition 1 (Bitwise zero-sum property) *If only one byte of* #2 *is* A *and the other 15 bytes are* C *(Λ set), the bitwise XOR-sum of* 2^8 *multiset of any bits in* #3 *to* #10 *is zero.*

Proof. As shown in Fig. 2, if #2 consists of a Λ set, #3 is also a Λ set, and #5 consists of 4 bytes of A and 12 bytes of C. Then, #7 and #9 consist of 16 bytes of A and B, respectively. In the 2^8 multiset of each bit of A, C and B, the XOR sum becomes zero [4]. □

3.2 Chosen-plaintext bitwise multiset attack

The bitwise zero-sum property allows us to develop chosen-plaintext key recovery attacks using leaked bits at a fixed position in #3, #5, #7 or #9. Our attack

firstly guesses 4 bytes of the key \$0, and chooses a set of 2^8 plaintexts resulting in Λ set in #2. If 4 bytes of \$0 are correctly guessed, the bitwise XOR sum of 2^8 leaked bits in any bit position of #3 to #10 is zero (Proposition 1). Otherwise, the probability that the bitwise XOR sum of leaked bits of #5, #7 and #9 is zero is 2^{-1}. If this procedure repeats with N different sets of 2^8 plaintexts, wrong keys can be detected with a probability of $(1 - 2^{-N})$.

First, we prepare a table of 2^{32} plaintexts in which all values of $\#0_0$, $\#0_5$, $\#0_{10}$, $\#0_{15}$ appear once and the other 12 bytes are fixed, and corresponding leaked bits. Assuming that the leaked bits can come from any position of #5, #7 or #9, our attack is performed as follows:

1. Guess $\$0_0$, $\$0_5$, $\$0_{10}$, $\$0_{15}$ (4 bytes) and choose $\#2_1$, $\#2_2$, $\#2_3$ (3 bytes).
2. Compute 2^8 the 4 bytes of $\#0_0$, $\#0_5$, $\#0_{10}$, $\#0_{15}$ backwards with all 2^8 values of $\#2_0$.
3. Get 2^8 leaked bits by accessing the prepared table, and compute the XOR sum of 2^8 leaked bits.
4. Repeat steps 1 to 3 N times with different values of $\#2_1$, $\#2_2$, $\#2_3$. If all N sets of XOR-sums are zero, regard it as a correct key.
5. Repeat steps 1 to 4 with all 2^{32} key candidates for $\$0_0$, $\$0_5$, $\$0_{10}$, $\$0_{15}$.
6. Repeat steps 1 to 5 for three times with the other three 4-byte sets of the key \$0 and corresponding bitwise multiset characteristics and tables.

The number of surviving keys after the above procedure is estimated as $(1 + 2^{-N} \times (2^{32} - 1))^4$. If the remaining key candidates are exhaustively searched, time complexity is estimated as $\{(2^{32} \times 2^8 \times N) \times 4\} + (1 + 2^{-N} \times (2^{32} - 1))^4$ encryptions. When $N = 22$, the time complexity is estimated as $2^{46.46}$ encryptions, the required data is 2^{34} ($= 2^{32} \times 4$) chosen plaintexts and the required memory is 2^{34} bits. This attack is successful if leaked bits come from any bits of #5, #7 and #9 without any knowledge of the location of leaked bits.

3.3 Partial key recovery attack using leaked bits from #3

If leaked bits come from #3, a 32-bit partial key-recovery attack is feasible as AES takes 2 rounds to achieve the full diffusion. If 4 bytes of keys \$0 are guessed correctly, 2^8 multiset in only one byte of #3 is not C as shown in Fig. 2, while for a wrong key, 2^8 multisets in 4 bytes of one column are not C. We exploit the gap of the number of C in #3 between a correct key and a wrong key.

We guess the column in #3 where leaked bits come from and then guess corresponding 4 bytes of \$0. We check whether the 2^8 multiset of leaked bits is fixed with N different sets of 2^8 plaintexts. A correct key can be detected with probability of $(1 - 2^{-8N})$ if leaked bits come from the byte which is C for a correct key and B for a wrong key. We repeat this $4 \times 4/3$ times by guessing different columns and the byte position of leaked bits in #3 and corresponding 4 bytes of \$0. The corresponding 32 bits of the key \$0 can be recovered with about 2^{44} ($\approx 2^{32} \times 2^8 \times 4 \times 4 \times 4/3$) encryptions when $N = 4$, 2^{34} chosen ciphertexts and 2^{34} memory.

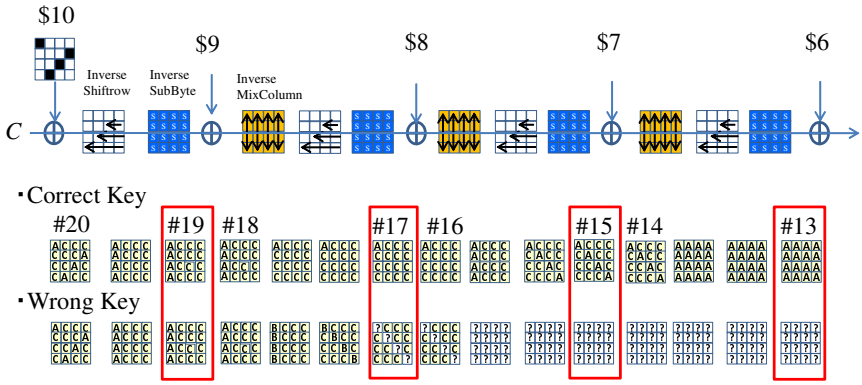

Fig. 3. Bitwise multiset characteristics in 4-round AES-128 in backward direction

3.4 Chosen-ciphertext bitwise multiset attack

In the chosen-ciphertext setting, backward direction attacks are feasible by using leaked bits from #13, #15, #17 or #19. As shown in Fig. 3, if 4 bytes of \$10 are correctly guessed and a set of ciphertexts is properly chosen, the XOR-sum of 2^8 multiset of any bit in #12 to #17 is zero (Proposition 1). Since states #13, #15 and #17 correspond to #7, #5 and #3, respectively, chosen-ciphertext attacks using these bits are feasible in the same manner as for chosen-plaintext attacks.

Also, #19 is affected by only one byte of \$10. Thus, one byte of \$10 can be recovered by the exhaustive search with 8 leaked bits from different ciphertexts after guessing 128 positions of the leaked bit. Time complexity is estimated as 2^{18} ($= 2^8 \times 128 \times 8$) encryptions, the required data is about 2^8 known ciphertexts, and the memory consumption is negligible.

3.5 Combined key recovery attacks on AES

Finally, we introduce a key recovery attack on the full AES-128 by combining the forward and the backward direction attacks. Since we do not know in which round the bits leak, we guess it and then mount each round attack in the following order: #19 $\rightarrow$ #17 $\rightarrow$ #3 $\rightarrow$ #5 $\rightarrow$ #7 $\rightarrow$ #9 $\rightarrow$ #13 $\rightarrow$ #15, i.e., if a correct key is not found by the guessed-round attack, the next round attack is applied in that given order. Our attacks find a correct key successfully except the case where the leaked bits come from #11. Thus the success probability without any knowledge of locations of leaked bits is 0.899 ($= 8/9$).

Time complexity is estimated as 2^{48} ($\approx 2^{18} + 2^{44} + 2^{44} + 2^{46.46} + 2^{46.46}$) encryptions. The required data is about 2^{35} ($= 2^{34} + 2^{34}$) chosen plaintexts and 2^{34} chosen ciphertexts and the required memory is 2^{34} bits. Note that if the leaked bits come from #3, #17, #19, partial key recovery attacks are possible.

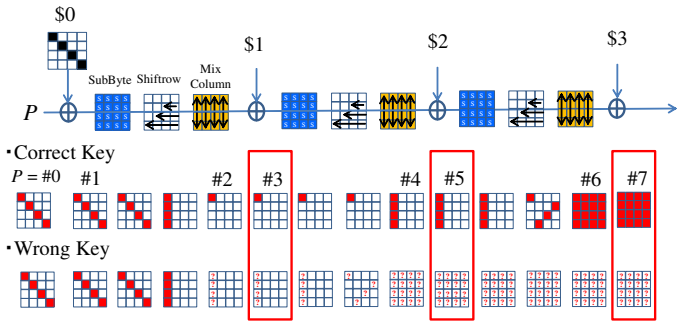

Fig. 4. Truncated differential characteristic over 3-round AES-128

4 Uncertainty and Differential Bias Attacks

The attacker can also have limited control over the execution environment. In particular, the time and space can be uncertain. We assume now that the attacker does not know bit positions and/or the number of rounds of leaked bits. Moreover, the values leaked can be incorrect due to noise or other operations executed in parallel to encryption/decryption. This can happen both for purely technical reasons on a complex multi-process platform and due to countermeasures. This section deals with these uncertainties and develops a cryptanalytic technique that is coined *differential bias attack*.

In a nutshell, the technique works as follows. Let Z_i be a leaked bit from an i-th execution of the encryption function. Our attacks observe a stream of leaked bits $Z_0, Z_1, Z_2, Z_3, \ldots$ and recover the correct key by applying techniques of distinguishing attacks from the domain of stream ciphers [15, 16, 23]. More specifically, we guess a part of the key \$0, and set well-chosen differences for a pair of plaintexts resulting in *biased differential states*, where the distribution of bitwise differences is biased, if the part of key \$0 is correctly guessed. As a leaked bit stream from *biased differential states* is also biased, we are able to detect the bit stream corresponding to the correct key by checking bias on the differences of bits. Also, if leakage after round 9 is available, a more powerful attack, called *biased state attack*, is feasible by using similar techniques.

Formalization of uncertain time and space. We assume a random unknown round (time) and/or bit position (space) within the round of the leak. Again, each execution of encryptions leaks only *one bit* of internal states at the random location. More formally, leaked bits are assumed to randomly come from the target space of internal states. For example, if the target space consists of all states after each round function of the data transform and key schedule, it is the leakage from states $\#0, \#3, \ldots, \#19, \#21$ and states \$0, \$1, ..., \$10. A target space can be a subset of those states if some rounds are black-boxed (and, thus, not visible to the attacker).

4.1 Truncated differential characteristic

Our attacks utilize a bytewise truncated differential characteristic of Fig. 4, where a colored-cell is a probability-one non-zero truncated difference, a blank cell is a probability-one zero truncated difference, and ? is an unknown truncated difference. Define 4 bytes of differences $\{\Delta\#0_0, \Delta\#0_5, \Delta\#0_{10}, \Delta\#0_{15}\}$ in a pair of plaintexts as $(\Delta\#0_0, \Delta\#0_5, \Delta\#0_{10}, \Delta\#0_{15}) = S^{-1}(MC^{-1}(\Delta\#2_0, 0, 0, 0))$, where S^{-1} and MC^{-1} are the inverses of SubBytes and MixColumns in a column, respectively, and $\Delta\#2_0$ is an arbitrary byte difference in $\#2_0$. Given $\{\Delta\#2_0, \#2_0, \ldots, \#2_3\}$ and $\{\$0_0, \$0_5, \$0_{10}, \$0_{15}\}$, $\{\Delta\#0_0, \Delta\#0_5, \Delta\#0_{10}, \Delta\#0_{15}\}$ and $\{\#0_0, \#0_5, \#0_{10}, \#0_{15}\}$ are determined. Let $\#0'$ be a plaintext having differences $\{\Delta\#0_0, \Delta\#0_5, \Delta\#0_{10}, \Delta\#0_{15}\}$, i.e., $\#0'_0 = \#0_0 \oplus \Delta\#0_0, \#0'_5 = \#0_5 \oplus \Delta\#0_5, \#0'_{10} = \#0_{10} \oplus \Delta\#0_{10}, \#0'_{15} = \#0_{15} \oplus \Delta\#0_{15}$. Also, let $\#'1, \ldots, \#'21$ be the corresponding states of $\#0'$, and $Z'_0, Z'_1, Z'_2, Z'_3, \ldots$ be leaked bits of each execution of $\#0'$.

4.2 Biased differential state

Choosing 4-byte differences $\{\Delta\#0_0, \Delta\#0_5, \Delta\#0_{10}, \Delta\#0_{15}\}$ properly and guessing the 4 bytes of $\{\$0_0, \$0_5, \$0_{10}, \$0_{15}\}$ correctly, we are able to create *biased differential states* in $\#3$: consisting of 15 bytes of probability-one zero differences and 1 byte of a probability-one non-zero difference, $\#5$: consisting of 12 bytes of probability-one zero differences and 4 bytes of probability-one non-zero differences, and $\#7$: consisting of 16 bytes of probability-one non-zero differences. As shown in Fig. 4, a correct key has 27 bytes of probability-one zero differences $\#3_1, \ldots, \#3_{15}$ and $\#5_4, \ldots, \#5_{15}$ and 21 bytes of probability-one non-zero differences $\#3_1$, $\#5_0, \ldots, \#5_3$, and $\#7_0, \ldots, \#7_{15}$, while a wrong key has only 12 bytes of probability-one zero differences $\#3_4, \ldots, \#3_{15}$ and does not have any probability-one non-zero difference in the state of the data processing part.

In addition, a pair of plaintexts has 12 bytes of probability-one zero differences and 4 bytes of probability-one non-zero differences for both a correct key and a wrong key. Also, the key schedule has 176 ($= 16 \times 11$) bytes of probability-one zero differences, as the subkeys are always fixed under the same key.

4.3 Bitwise differential bias in biased differential state

For a probability-one zero/non-zero truncated difference, we derive positive and negative bitwise differential biases. Our attack exploits the gap of the number of positive and negative biases between a correct key and a wrong key when a pair of $\#0$ and $\#'0$ is encrypted.

Positive bitwise bias for probability-one zero truncated difference. If a bytewise pair $\#x_y$ and $\#x'_y$ has a probability-one zero truncated difference, a bitwise difference at the same position is also zero with probability one: $Pr(\Delta[\#x_y[j], \#'x_y[j]] = 0) = 1, 0 \leq j \leq 7$, where $\Delta[a, b] = a \oplus b$. A correct key has 1720 ($= 27 \times 8 + 176 \times 8 + 12 \times 8$) *positive* bitwise differential biases, while a wrong key has only 1600 ($= 12 \times 8 + 176 \times 8 + 12 \times 8$) such biases.

Table 2. Bitwise differential biases for truncated differential of Fig. 4

	Positive biases toward zero	Negative biases toward zero
Correct key	$\#3_i[j]$ $(1 \le i \le 15, 0 \le j \le 7)$ $\#5_i[j]$ $(4 \le i \le 15, 0 \le j \le 7)$	$\#3_0[j]$ $(0 \le j \le 7)$ $\#5_i[j]$ $(0 \le i \le 3, 0 \le j \le 7)$ $\#7_i[j]$ $(0 \le i \le 15, 0 \le j \le 7)$
Wrong key	$\#3_i[j]$ $(4 \le i \le 15, 0 \le j \le 7)$	-
Both keys	$\#0_i[j]$ $(i \ne 0, 5, 9, 15 \le j \le 7)$ $\$x_i[j] (0 \le x \le 10, 1 \le i \le 15, 0 \le j \le 7)$	$\#0_i[j]$ $(i = 0, 5, 9, 15, 0 \le j \le 7)$

Negative bitwise bias for probability-one non-zero truncated difference. If a pair $\#x_y$ and $\#x'_y$ has a probability-one non-zero truncated difference, the probability that a bitwise difference at the same bit position is zero is estimated as follows: $Pr(\Delta[\#x_y[j], \#'x_y[j]] = 0) = 127/255 = 1/2 \cdot (1 - 2^{-7.99})$ In experiments with 2^{40} randomly-chosen plaintexts and keys, we confirmed that these negative biases toward zero exist in each bit of the probability-one non-zero truncated difference, where the experimental value is $Pr(\Delta[\#7_i[j], \#'7_i[j]] = 0) = 1/2 \cdot (1 - 2^{-7.92})$.

A correct key has 200 $(= 21 \times 8 + 4 \times 8)$ *negative* bitwise differential biases, while a wrong key has 32 $(= 0 + 4 \times 8)$ ones. The summary of bitwise positive/negative differential biases for the truncated differential of Fig. 4 is shown in Table 2.

4.4 Bitwise differential biases in the stream of leaked bits

Suppose that values of the other bits of the states in the data processing part and the key schedule are randomly distributed, i.e., the probability that differences of other bitwise pairs become zero is 2^{-1}. Let N_{all}, N_{bias_p}, N_{bias_n}, and N_{random} be the number of bitwise pairs in entire space, positive biased space (toward zero), negative biased space (toward zero) and randomly-distributed space, respectively, and x^c and x^w be those of a correct key and a wrong key, respectively (see Fig. 5). The probabilities that a difference of a bitwise pair of randomly-chosen leaked bits is zero ($\Delta[Z_i, Z'_j] = 0$) for a correct key and a wrong key are estimated as follows:

$$Pr^c(\Delta[Z_i, Z'_j] = 0) = 1/2 \cdot (N^c_{random}/N_{all}) + N^c_{bias_n}/N_{all} \cdot (127/255) + N^c_{bias_p}/N_{all},$$

$$Pr^w(\Delta[Z_i, Z'_j] = 0) = 1/2 \cdot (N^w_{random}/N_{all}) + N^w_{bias_n}/N_{all} \cdot (127/255) + N^w_{bias_p}/N_{all}.$$

Our attack observes leaked bits $Z_0, Z_1, Z_2, Z_3, \ldots$ and $Z'_0, Z'_1, Z'_2, Z'_3, \ldots$, and then computes the probability of $\Delta[Z_i, Z'_j] = 0$ in order to distinguish a stream coming from the distribution for a correct key from streams coming from the distribution for a wrong key.

The number of required samples for distinguishing the two distributions with probability of $1 - \alpha$ is given by the following lemmata.

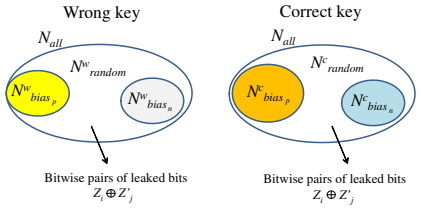

Fig. 5. Bias in leaked stream

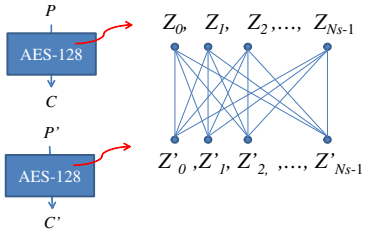

Fig. 6. Bitwise pairs of leaked bits

Lemma 1 [15, 16] *Let X and Y be two distributions and suppose that the independent events E occur with probabilities $Pr_X(E) = p$ in X and $Pr_Y(E) = (1+q)\cdot p$ in Y. Then the discrimination D of the distributions is $p \cdot q^2$.*

Lemma 2 [15] *The number of samples N_{sample} that is required for distinguishing two distributions that have discrimination D with success probability $1-\alpha$ is $(1/D)\cdot(1-2\alpha)\cdot log_2\frac{1-\alpha}{\alpha}$.*

Assuming that the target event E is $\Delta[Z_i, Z'_j] = 0$, X is the distribution for a wrong key, and Y is the distribution for a correct key, p and q are estimated as $p = Pr^w(\Delta[Z_i, Z'_j] = 0)$ and

$$q = \frac{-N^c_{bias_n} + N^w_{bias_n} + 255(N^c_{bias_p} - N^w_{bias_p})}{255N_{all} - N^w_{bias_n} + 255N^w_{bias_p}}.$$

For success probability $1 - 2^{-32}$, the estimated number of required samples is:

$$N_{sample} = (pq^2)^{-1}\cdot(1 - 2\cdot 2^{-32})\cdot log_2\frac{1-2^{-32}}{2^{-32}} \approx 2\cdot 32\cdot q^{-2} = 2^6\cdot q^{-2}.$$

4.5 Chosen-plaintext differential bias attack

First, this attack prepares 2^{32} chosen plaintexts in which all 2^{32} values of $\#0_0$, $\#0_5$, $\#0_{10}$, $\#0_{15}$ appear once and the other 12 bytes are fixed, and obtains N_s leaked bits in each plaintext, i.e., each plaintext is encrypted N_s times. Given a pair of P and P', N_s^2 $(= N_s \times N_s)$ pairs of leaked bits are obtained as shown in Fig. 6. After we make a table of the values of $\{\#0_0, \#0_5, \#0_{10}, \#0_{15}\}$ and corresponding N_s leaked bits, our attack is performed as follows:

1. Guess the 4 bytes of key $\$0_0, \$0_5, \$0_{10}, \$0_{15}$, and choose $\Delta\#2_0, \#2_0, \ldots, \#2_3$.
2. Compute a pair of 4 bytes of plaintexts, $\#0_0, \#0_5, \#0_{10}, \#0_{15}$ and $\#0'_0, \#0'_5, \#0'_{10}, \#0'_{15}$, resulting in biased #3, #5 and #7 states if a key is correctly guessed.
3. Get N_s^2 pairs of leaked bits $\Delta[Z_i, Z'_j], 0 \le i, j < N_s$ by accessing the prepared table.

4. Repeat steps 2-3 N_{sample}/N_s^2 times with different values of #2.
5. Check whether a distribution of N_{sample} pairs is the one for a correct key. If so, regard it as a candidate for the correct key.
6. Repeat steps 1 to 5 with all 2^{32} candidates of keys $\$0_0$, $\$0_5$, $\$0_{10}$, $\$0_{15}$.
7. Repeat steps 1 to 6 for three times with the other three 4-byte sets of the key \$0, corresponding truncated differential characteristics, and the tables of plaintexts and leaked bits.

In steps 3 to 5, we check N_{sample} pairs to detect a stream coming from the biased distribution for a correct key. In the step 3, we count the number of the events $\Delta[Z_i, Z'_j] = 0$, and estimate the probability $Pr(\Delta[Z_i, Z'_j] = 0)$. The straight forward method requires N_s^2 operations to check all N_s^2 pairs. To improve it, we first calculate the number of $Z_i = 0$, $0 \leq i < N_s$, defined as N_{zero}. Then the number of $\Delta[Z_i, Z'_j] = 0$ is estimated as

$$\big(N_{zero} \times (\overline{Z'_0} + \ldots, +\overline{Z'}_{N_s-1}) + ((N_s - N_{zero}) \times (Z'_0 + \ldots, +Z'_{N_s-1})\big)/N_{all},$$

where $\overline{a}$ is the complement of a. These costs are estimated as $N_s + (N_s + N_s)$ additions and multiplications. It is assumed to be less than N_s one-round encryptions. The number of surviving keys after the above procedure is estimated as $(1+2^{-\alpha} \times (2^{32}-1))^4$. If the remaining key candidates are exhaustively searched, the entire time complexity is estimated as $(2^{32} \times 4 \times N_{sample}/N_s \times 1/10) + (1 + 2^{-32} \times (2^{32} - 1))^4 \approx 2^{31} \times N_{sample}/N_s$ encryptions and the required data is $2^{34} \times N_s$ $(= 4 \times 2^{32} \times N_s)$ chosen plaintexts with leaked bits. The memory requirement is $2^{34} \times N_s$ bits.

4.6 Chosen-ciphertext differential bias attack

If the decryption function is accessible, chosen-ciphertext attacks are applicable. Similarly to the setting of bitwise mutiset attacks before, the chosen-ciphertext attacks are more efficient and it makes sense to black-box the output of round 9 also in the cases with time and space uncertainty.

As shown in Fig. 7, the states #13, #15 and #17 correspond to #7, #5 and #3, respectively. Since the state #19 consists of 12 probability-one zero truncated differences and 4 probability-one non-zero truncated differences, both for a correct key and a wrong key, one additionally has 96 positive and 32 negative bitwise differential biases in the chosen-ciphertext attack.

Biased state attack of #19: Leakage after round 9. If leaked bits from #19 are obtained, a more powerful attack is feasible. Each byte in #19 can be controlled by one byte of \$10 and one byte of a ciphertext. Thus, we are able to create a *biased state* in #19 whose one byte (8 bits) is fixed to 0, if the corresponding byte of \$10 is correctly guessed and the respective byte of the ciphertext is property chosen. Suppose that the values of the other bits of the states are randomly distributed. The probabilities that each leaked bit is zero $(Z_i = 0)$ for a correct key is estimated as $Pr^c(Z_i = 0) = 1/2 \cdot (N'^c_{random}/N'_{all}) +$

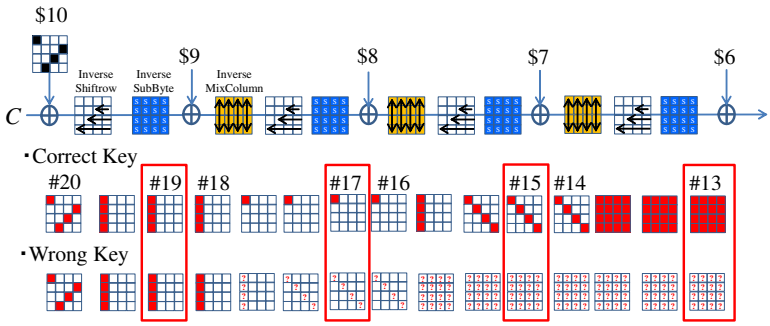

Fig. 7. Differential characteristic over 4-round AES-128 in backward direction

$N'^{c}_{bias_p}/N'_{all}$, where N'_{all}, N'_{bias_p}, and N'_{random} are the numbers of bits in entire space, positive biased space and randomly-distributed space, respectively. Also, $Pr^w(Z_i = 0)$ is assumed to be $1/2$.

Assuming that the target event E is $Z_i = 0$, p and q are estimated as $p = 1/2$ and $q = N'^{c}_{bias_p}/N'_{all}$. For the success rate of $1 - 2^{-8}$ ($\alpha = 2^{-8}$), the sample requirement is estimated as $N'_{sample} \approx 2 \cdot 8 \cdot (q)^{-2} = 2^4 \cdot (q)^{-2}$. We repeat the procedure for all 16 bytes of \$10. Therefore, time complexity is estimated as $2^{12} \times N'_{sample}$ ($= 16 \times 2^8 \times N'_{sample}$) encryptions and the required data is $2^{12} \times N'_{sample}$ ($= 16 \times 2^8 \times N'_{sample}$) chosen ciphertexts. The memory requirement is negligible.

4.7 Known-plaintext differential bias attack

Finally, we introduce a known-plaintext differential bias attack using a truncated differential characteristic of Fig. 8. For a correct key, one has 24 ($= 3 \times 8$) positive bitwise differential biases toward zero and 8 negative bitwise differential biases in #3, while for a wrong key, there are not such biases. The key schedule has the same number of positive biases of chosen-plaintext attacks and the plaintext has 32 ($= 4 \times 8$) negative biases in both of a correct and a wrong key.

This attack prepares 2^{33} known plaintexts and makes a table of $\#0_0$, $\#0_5$, $\#0_{10}$, $\#0_{15}$ and the corresponding N_s leaked bits. The expected number of the entries of each value of $\#0_0$, $\#0_5$, $\#0_{10}$, $\#0_{15}$ is more than 1. We mount key recovery attacks for $\$0_0$, $\$0_5$, $\$0_{10}$, $\$0_{15}$ in the same manner as in the chosen-plaintext attack. In step 3, the prepared table contains the corresponding values of $\#0_0$, $\#0_5$, $\#0_{10}$, $\#0_{15}$ with high probability. Thus, time complexity is estimated as $2^{31} \times N_{sample}/N_s$ encryptions and the required data is $2^{35} \times N_s$ ($= 4 \times 2^{33} \times N_s$) known plaintexts with leaked bits and the required memory is about $2^{35} \times N_s$ bits.

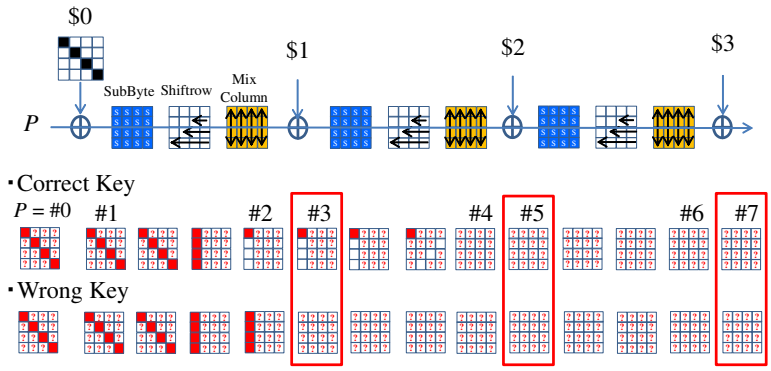

Fig. 8. Differential characteristic over 3-round AES-128 for known plaintext attack

5 AES under Leakage with Uncertainty in Time/Space

This section evaluates the security of AES if the attacker is uncertain about time and space, that is, if the round of leak and/or the bit position of leak within the round are randomized. Since the multiset of leaked bits at the fixed location is not available in the random unknown setting, our bitwise multiset attacks are not applicable to these variants. Thus, we estimate the costs of differential (state) bias attacks on each variant of AES with countermeasures as shown in Fig. 1.

Formalization of time/space uncertainty for AES. We speak of *randomized time*, when one bit of the state information is leaked at a *fixed* bit position after a *random* number of rounds, e.g., $\#(2x+1)_{10}[7]$ $(0 \le x \le 10)$ or $\$x_3[4]$ $(0 \le x \le 10)$. We speak of *randomized space*, when one bit of the state information is leaked at a *random* bit position after a *fixed* number of rounds, e.g., $\{\#17_i[j], \$8_i[j]\}$ $(0 \le i \le 15,\ 0 \le j \le 7)$. *Randomized time and space* occur, when one bit of state information is leaked at a *random* bit position after a *random* number of rounds, e.g., $\#(2x+1)_i[j]$ $(0 \le x \le 10,\ 0 \le i \le 15$ and $0 \le j \le 7)$ or $\$x_i[j]$ $(0 \le x \le 10,\ 0 \le i \le 15$ and $0 \le j \le 7)$.

5.1 Uncertainty in space

The space randomization makes the bit position of leaked bits random in each execution, i.e., Z_i and Z'_i randomly come from two 256-bit spaces consisting of a 128-bit state in the data processing part and a 128-bit state in the key schedule at the unknown fixed round, assuming encryptions are executed with a 256-bit working memory for a internal state and a subkey.

Assuming that leaked bits come from the states after round 2, i.e., $\{\#5_i[j]$ and $\$2_i[j]\}$ and $\{\#'5_i[j]$ and $\$'2_i[j]\}$ $(0 \le i \le 15,\ 0 \le j \le 7)$, the parameters of our differential bias attacks are chosen as $N_{all} = (256)^2$, $N^{(c)}_{bias_p} = 224$

$(= 96 + 128)$, $N^{(w)}_{bias_p} = 128$ $(= 0 + 128)$, $N^{(c)}_{bias_n} = 32$ and $N^{(w)}_{bias_n} = 0$ (see Table 2). Then, $Pr^c(\Delta[Z_i, Z'_j] = 0)$ and $Pr^w(\Delta[Z_i, Z'_j] = 0)$ are estimated as $1/2 \cdot (1 + 2^{-8.192})$, and $1/2 \cdot (1 + 2^{-9.000})$, respectively, and $q = 2^{-9.42}$. In our experiment with 2^{40} randomly-chosen correct and wrong pairs of keys and plaintexts, $Pr_c(\Delta[Z_i, Z'_j] = 0)$ and $Pr_w(\Delta[Z_i, Z'_j] = 0)$ are $1/2 \cdot (1 + 2^{-8.191})$ and $1/2 \cdot (1 + 2^{-9.001})$, respectively, and $q = 2^{-9.42}$. The number of required samples to detect a stream for a correct key is estimated as $N_{sample} = 2^{24.84}$ $(= 2^6 \times 2^{9.42 \times 2})$. We experimentally confirmed that this number of samples is enough for a successful attack. With $N_s = (N_{all})^{1/2}$, time complexity is estimated as $2^{47.84}$ $(= (2^{31} \times 2^{24.84})/(2^8))$ encryptions and the required data is 2^{42} $(= 2^{34} \times 2^8)$ chosen plaintexts and corresponding leaked bits with 2^{42} bits of prepared tables.

The details of attacks for $N_s = (N_{all})^{1/2}$ are provided in Table 3, where $q^{(e)}$ is our experimental value with 2^{40} randomly-chosen correct and wrong pairs of keys and plaintexts/ciphertexts, and T and D are time complexity and the amount of the required data, respectively. Our theoretical values closely approximate the experimental data in all cases. Since an attacker does not know the round number of leaked bits, he firstly guesses the round of leaked bits and then mounts an attack similar to the combined attack of the bitwise multiset attacks. If the decryption is accessible, our attacks are successful except the case where leaked bits come from states after 4 or 5 round only. Also, a known plaintext attack is possible if leaked bits from #3 are available.

5.2 Uncertainty in time

The time randomization makes the round number of leaked bits random in each execution, i.e., Z_i and Z'_i come from the fixed bit position at a random round of the data processing part. Additionally, we take into account the leaked bits from plaintexts #0 or ciphertexts #21 in the data processing part. For instance, assuming that leaked bits come from 33-th bits of the data processing part, i.e., $\#0_4[1]$, $\#3_4[1]$, …, $\#19_4[1]$ or $\#21_4[1]$, the attack parameters are given as $N_{all} = 11^2$, $N^{(c)}_{bias_p} = 3$, $N^{(w)}_{bias_p} = 2$, $N^{(c)}_{bias_n} = 1$, $N^{(w)}_{bias_n} = 0$. Then $q = 2^{-6.45}$, $N_{sample} = 2^{19.9}$, $T = 2^{47.44}$ and $D = 2^{37.46}$.

The details of our attacks using leaked bits from the data processing part are provided in Table 4, where the attack parameters of chosen-plaintext differential bias attacks depend on the positions of leaked bits, but time and data complexities are almost same for each position. We also evaluate a chosen-plaintext attack when round 9 and round 1, 2, 8 and 9 rounds are black-boxed. i.e., {#19, \$9} and {#3,#5,#17, #19, \$1, \$2, \$8, \$9} are not available, respectively. Other black-boxed variants are also evaluated by properly choosing attack parameters. Since an attacker does not know the bit position of leaked bits, he firstly guesses it and then mounts an attack. If the decryption is accessible, our attacks are feasible as long as leaked bits after round 1, 2, 3, 6, 7, 8 or 9 in the data processing part are available. A known plaintext attack is applicable if leaked bits from #3

are obtained. However, it is a 32-bit key recovery attack, because a bit of #3 is affected by 32 bits of \$0.

5.3 Uncertainty in both space and time

The space and time randomization makes the both the bit position and the round number of leaked bits random in each execution, i.e., Z_i and Z'_i randomly come from any bit of any states in the data processing part {#0, #3, #5, ..., #19, #21} and in the key schedule {\$0, ..., \$10}. The parameters of the chosen-plaintext differential bias attacks are estimated as $N_{all} = (256 \times 11)^2$, $N^{(c)}_{bias_p} = 1720$ $(= 27 \times 8 + 176 \times 8 + 12 \times 8)$, $N^{(w)}_{bias_p} = 1600$ $(= 12 \times 8 + 176 \times 8 + 12 \times 8)$, $N^{(c)}_{bias_n} = 200$ $(= 21 \times 8 + 0 + 4 \times 8)$ and $N^{(w)}_{bias_n} = 32$ $(= 0 + 0 + 4 \times 8)$.

The details of our attacks are given in Table 5. We also provide a chosen-plaintext attack when round 9 and round 1, 2, 8 and 9 are black-boxed. If the decryption is accessible, our attacks work as long as leaked bits after round 1, 2, 3, 6, 7, 8 or 9 of the data processing part are available. Also, a known-plaintext attack is applicable if leaked bits from #3 are observable.

6 AES under Noisy Leakage

This section studies the effect of additional noise on top of the time and space randomization. The noise can be due to the limited knowledge of the platform by the adversary or due to the implemented countermeasures such as insertion of dummy operations. In the differential bias attack, this reduces the rate of positive/negative biased bits by adding noise bits into the space of the actually leaked bits. To quantify the amount of noise present in the attack, we define π as the probability that an observed bit is not a noise bit. Suppose that the values of the noise bits are randomly distributed, the bias of a leaked bit stream of the correct key with noise bits is estimated as $q' = q \times \pi$, and the required number of sample bits to distinguish a stream for a correct key increases by the multiple of $(\pi^2)^{-1}$ to $N'_{sample} = N_{sample} \times (\pi^2)^{-1}$. With $N_s = (N_{all})^{1/2} \times \pi^{-1}$, the time and data complexities of our known/chosen plaintext differential bias attacks increase by the multiple of $(\pi)^{-1}$ as $T \approx 2^{31} \times (N_{sample} \times \pi^{-2})/N_s \times \pi^{-1} = 2^{31} \times (N_{sample} \times \pi^{-1})/N_s$ encryptions and $D \approx 2^{34}(2^{35}) \times (N_s \times \pi^{-1})$ chosen/known-plaintexts with leaked bits. Also, the time and data complexities of chosen-ciphertext biased state attacks increase by the multiple of $(\pi)^{-2}$. The detailed evaluations for each values of π are shown in Table 6.

7 Towards More Alignment: Bytewise Leakage

Here we deal with the case where each execution leaks *one byte of a byte-aligned* state. In other words, now we let aligned bytes of internal states leak. Such leaks

Table 3. Differential bias and biased state attacks for space randomization

Chosen-Plaintext(Ciphertext) Differential Bias Attack										
Round	N_{all}	$N^c_{bias_p}$	$N^w_{bias_p}$	$N^c_{bias_n}$	$N^w_{bias_n}$	q	$q^{(e)}$	N_{sample}	T	D
1 (8)	256^2	248	224	8	0	$2^{-11.42}$	$2^{-11.38}$	$2^{28.84}$	$2^{51.84}$	$2^{42.00}$ CP(CC)
2 (7)	256^2	224	128	32	0	$2^{-9.42}$	$2^{-9.42}$	$2^{24.84}$	$2^{47.84}$	$2^{42.00}$ CP(CC)
3 (6)	256^2	128	128	128	0	$2^{-16.99}$	$2^{-16.84}$	$2^{39.98}$	$2^{62.98}$	$2^{42.00}$ CP(CC)
Known-Plaintext Differential Bias Attack										
1	256^2	152	128	8	0	$2^{-11.42}$	$2^{-11.10}$	$2^{28.84}$	$2^{51.84}$	$2^{43.00}$ KP
Chosen-Ciphertext Biased State Attack										
Round	N'_{all}	$N'^c_{bias_p}$	$N'^w_{bias_p}$	-	-	q	$q^{(e)}$	N'_{sample}	T	D
9	256	8	0	-	-	$2^{-5.00}$	$2^{-5.00}$	2^{14}	$2^{26.00}$	$2^{26.00}$ CC

Table 4. Differential bias and biased state attacks for time randomization

Chosen-Plaintext Differential Bias Attack										
BB round	N_{all}	$N^c_{bias_p}$	$N^w_{bias_p}$	$N^c_{bias_n}$	$N^w_{bias_n}$	q	$q^{(e)}$	N_{sample}	T	D
None	11^2	3	2	1	0	$2^{-6.95}$	$2^{-6.94}$	$2^{19.90}$	$2^{47.44}$	$2^{37.46}$ CP
9	10^2	3	2	1	0	$2^{-6.68}$	$2^{-6.68}$	$2^{19.36}$	$2^{47.04}$	$2^{37.32}$ CP
1, 2, 8, 9	7^2	0	0	1	0	$2^{-13.61}$	$2^{-13.23}$	$2^{33.22}$	$2^{60.41}$	$2^{36.81}$ CP
Known-Plaintext Differential Bias Attack										
None	11^2	1	0	0	0	$2^{-6.92}$	$2^{-7.30}$	$2^{19.84}$	$2^{47.38}$	$2^{38.46}$ KP
Chosen-Ciphertext Biased State Attack										
BB round	N'_{all}	$N'^c_{bias_p}$	$N'^w_{bias_p}$	-	-	q	$q^{(e)}$	N'_{sample}	T	D
None	11	1	0	-	-	$2^{-3.46}$	$2^{-3.45}$	$2^{10.92}$	$2^{22.92}$	$2^{22.92}$ CC

Table 5. Differential bias and biased state attacks for space and time randomization

Chosen-Plaintext Differential Bias Attack										
BB round	N_{all}	$N^c_{bias_p}$	$N^w_{bias_p}$	$N^c_{bias_n}$	$N^w_{bias_n}$	q	$q^{(e)}$	N_{sample}	T	D
None	$(256 \cdot 11)^2$	1720	1600	200	32	$2^{-16.02}$	$2^{-15.92}$	$2^{38.04}$	$2^{57.58}$	$2^{45.46}$ CP
9	$(256 \cdot 10)^2$	1592	1472	200	32	$2^{-15.75}$	$2^{-15.70}$	$2^{37.49}$	$2^{57.17}$	$2^{45.32}$ CP
1, 2 8, 9	$(256 \cdot 7)^2$	896	896	128	0	$2^{-22.61}$	$2^{-23.07}$	$2^{51.22}$	$2^{71.41}$	$2^{44.81}$ CP
Known-Plaintext Differential Bias Attack										
None	$(256 \cdot 11)^2$	1440	1408	40	32	$2^{-17.92}$	$2^{-17.69}$	$2^{41.84}$	$2^{61.38}$	$2^{46.46}$ KP
Chosen-Ciphertext Biased State Attack										
BB round	N'_{all}	$N'^c_{bias_p}$	$N'^w_{bias_p}$	-	-	q	$q^{(e)}$	N'_{sample}	T	D
None	$(256 \cdot 11)$	8	0	-	-	$2^{-8.46}$	$2^{-8.44}$	$2^{20.92}$	$2^{32.92}$	$2^{32.92}$ CC

Table 6. Differential bias and biased state attacks for leakage with noise

BB round	Time	Data	Time	Data	Time	Data	Time	Data
	$\pi = 1$		$\pi = 2^{-10}$		$\pi = 2^{-20}$		$\pi = 2^{-30}$	
Chosen-Plaintext Differential Bias Attack								
None	$2^{57.58}$	$2^{45.46}$ CP	$2^{67.58}$	$2^{55.46}$ CP	$2^{77.58}$	$2^{65.46}$ CP	$2^{87.58}$	$2^{75.46}$ CP
1, 2, 8, 9	$2^{71.41}$	$2^{44.81}$ CP	$2^{81.41}$	$2^{54.81}$ CP	$2^{91.41}$	$2^{64.81}$ CP	$2^{101.41}$	$2^{74.81}$ CP
Known-Plaintext Differential Bias Attack								
None	$2^{61.38}$	$2^{46.46}$ KP	$2^{71.38}$	$2^{56.46}$ KP	$2^{81.38}$	$2^{66.46}$ KP	$2^{91.38}$	$2^{76.46}$ KP
Chosen-Ciphertext Biased State Attack								
None	$2^{32.92}$	$2^{32.92}$ CC	$2^{52.92}$	$2^{52.92}$ CC	$2^{72.92}$	$2^{72.92}$ CC	$2^{92.92}$	$2^{92.92}$ CC

reflect the realities of a byte-oriented software implementation better. [5] In both settings – leakage with fixed and uncertain time/space – our techniques still apply. However, some adjustments are needed, see below.

7.1 Fixed time/space: Bytewise multiset attack

Our bitwise multiset attacks naturally extend to *bytewise* multiset attacks, because the multiset characteristics are based on the bytewise XOR-sum property. The success probability for detecting wrong keys increases from $(1 - 2^{-1})$ to $(1 - 2^{-8})$ by using the bytewise zero-sum property. Then the time complexities of 2, 3, 4, 6 and 7-round attacks are estimated as $\{(2^{32} \times 2^8 \times N) \times 4\} + (1 + 2^{-8N} \times (2^{32} - 1))^4$ encryptions. With $N = 4$, it is about 2^{44}. The time complexities of 1 and 8-round attacks and the 9-round attack also improve to 2^{42} ($\approx 2^{32} \times 2^8 \times 4 \times 4/3$) and 2^{12} ($= 2^8 \times 16$) encryptions, respectively. The time complexity of the combined attack is 2^{45} ($\approx 2^{12}+2^{42}+2^{42}+2^{44}+2^{44}$) encryptions and the required data is 2^{35} chosen plaintexts and 2^{34} chosen ciphertexts.

7.2 Uncertain time/space: Differential bias attack

Our differential bias attacks also extend to *bytewise* attacks using bytewise differential biases of truncated differential characteristics of Fig. 4, 7 and 8.

Chosen/known-plaintext differential bias attack. Let a leaked byte from the i-th execution be Z_i^*, and N_{all}^*, $N_{bias_p}^*$, $N_{bias_n}^*$, N_{random}^* be the number of bytewise pairs in the entire space, positive biased space, negative biased space and randomly-distributed space, respectively. The probabilities that a difference of a bytewise pair of randomly chosen leaked bytes is zero ($\Delta[Z_i^*, Z_j'^*] = 0$) for a correct key and a wrong key are estimated as follows.

$$Pr^c(\Delta[Z_i^*, Z_j'^*] = 0) = 1/2^8 \cdot (N_{random}^{*c}/N_{all}^*) + N_{bias_p}^{*c}/N_{all}^*,$$

[5] The stream cipher LEX can be regarded as a bytewise leakage model at the fixed space [3] but the locations of leaked bytes are known for the attacker. Thus, the attack against LEX [10] is not directly applicable to our unknown location model.

Table 7. Evaluation for byte-aligned space randomization ($N_s = (N_{all})^{1/2}$)

Chosen-Plaintext(Ciphertext) Differential Bias Attack										
Round	N_{all}	$N^c_{bias_p}$	$N^w_{bias_p}$	$N^c_{bias_n}$	$N^w_{bias_n}$	q	$q^{(e)}$	N_{sample}	T	D
1 (8)	32^2	31	28	1	1	$2^{-3.41}$	$2^{-3.42}$	$2^{19.84}$	$2^{45.84}$	$2^{39.00}$ CP(CC)
2 (7)	32^2	28	16	4	4	$2^{-0.74}$	$2^{-0.74}$	$2^{14.48}$	$2^{40.48}$	$2^{39.00}$ CP(CC)
3 (6)	32^2	16	16	16	0	$2^{-8.32}$	$2^{-8.38}$	$2^{29.64}$	$2^{55.64}$	$2^{42.00}$ CP
Known-Plaintext Differential Bias Attack										
1	32^2	19	16	1	0	$2^{-2.74}$	$2^{-2.74}$	$2^{18.48}$	$2^{44.48}$	$2^{40.00}$ KP
Chosen-Ciphertext Biased State Attack										
Round	N'_{all}	$N'^c_{bias_p}$	$N'^w_{bias_p}$	-	-	q	$q^{(e)}$	N_{sample}	T	D
9	32	1	0	-	-	$2^{2.99}$	$2^{2.99}$	$2^{11.00}$	$2^{23.00}$	$2^{23.00}$ CC

$$Pr^w(\Delta[Z_i^*, Z_j'^*] = 0) = 1/2^8 \cdot (N^{*w}_{random}/N^*_{all}) + N^{*w}_{bias_p}/N^*_{all}.$$

Assuming that the target event E is $\Delta[Z_i^*, Z_j^*] = 0$, X is a distribution for a wrong key, and Y is a distribution for a correct key, p and q are estimated as $p = Pr^w(\Delta[Z_i^*, Z_j'^*] = 0)$ and $q = \frac{-N^c_{bias_n}+N^w_{bias_n}+255(N^c_{bias_p}-N^w_{bias_p})}{N_{all}-N^w_{bias_n}+255N^w_{bias_p}}$. For the success probability of $1 - 2^{-32}$, the required sample size is estimated as $N^*_{sample} \approx 32 \cdot 256 \cdot q^2 = 2^{13} \cdot q^2$. Time complexity is estimated as $2^{31} \times N_{sample}/N_s$ encryptions and the required data is $2^{34}(2^{35}) \times N_s$ chosen/known plaintexts with leaked bits.

Chosen-ciphertext biased-state attack. Assuming that the target event E is $Z_i = 0$, p and q are estimated as $p = 1/2^8$ and $q = (255 \times N^c_{bias_p})/N_{all}$. The number of required samples is estimated as $N_{sample} \approx 8 \cdot 2^8 \cdot (q)^{-2}$. We repeat the procedure for all 16 byte of \$10. Therefore, time complexity is estimated as $2^{12} \times N_{sample}$ $(= 16 \times 2^8 \times N_{sample})$ encryptions and the number of required data is $2^{12} \times N_{sample}$ $(= 16 \times 2^8 \times N_{sample})$ chosen ciphertexts.

Security under time and space randomization and with leakage noise. The results of security evaluations under time and space randomization with noisy leakage are provided in Tables 7 to 10. [6] In all cases, time complexity and data requirements are improved compared to the bit-aligned attacks.

8 Some Extensions

8.1 AES-192 and 256

Bitwise multiset attacks and differential bias attacks on AES-128 are directly applicable to AES-192 and AES-256 in both fixed and random settings. In the

[6] If q is not small, Lemmata 1 and 2 are not applicable [16]. In this case we estimate $N_{sample} = 2^{11}$ and 2^{13} for known-plaintext differential bias attacks and chosen-ciphertext biased state attacks, respectively. We confirmed experimentally that these numbers of samples were enough for successful attacks.

Table 8. Evaluation for byte-aligned time randomization ($N_s = (N_{all})^{1/2}$)

Chosen-Plaintext Differential Bias Attack										
BB round	N_{all}	$N^c_{bias_p}$	$N^w_{bias_p}$	$N^c_{bias_n}$	$N^w_{bias_n}$	q	$q^{(e)}$	N_{sample}	T	D
None	11^2	3	2	1	0	$2^{-1.31}$	$2^{-1.31}$	$2^{15.62}$	$2^{43.16}$	$2^{37.46}$ CP
9	10^2	3	2	1	0	$2^{-1.26}$	$2^{-1.26}$	$2^{15.52}$	$2^{43.19}$	$2^{37.32}$ CP
1, 2, 8, 9	7^2	0	0	1	0	$2^{-5.61}$	$2^{-5.50}$	$2^{24.22}$	$2^{52.41}$	$2^{36.81}$ CP
Known-Plaintext Differential Bias Attack										
None	$(11)^2$	1	0	0	0	$2^{1.08}$	$2^{1.08}$	$2^{13.00}$	$2^{40.54}$	$2^{38.46}$ KP
Chosen-Ciphertext Biased state Attack										
BB round	N'_{all}	$N'^c_{bias_p}$	$N'^w_{bias_p}$	-	-	q	$q^{(e)}$	N_{sample}	T	D
None	11	1	0	-	-	$2^{4.50}$	$2^{4.50}$	$2^{11.00}$	$2^{23.00}$	$2^{23.00}$ CP

Table 9. Evaluation for byte-aligned space and time randomization ($N_s = (N_{all})^{1/2}$)

Chosen-Plaintext Differential Bias Attack										
BB round	N_{all}	$N^c_{bias_p}$	$N^w_{bias_p}$	$N^c_{bias_n}$	$N^w_{bias_n}$	q	$q^{(e)}$	N_{sample}	T	D
None	$(32 \cdot 11)^2$	215	200	25	4	$2^{-5.52}$	$2^{-5.53}$	$2^{24.04}$	$2^{46.58}$	$2^{42.45}$ CP
9	$(32 \cdot 10)^2$	199	184	25	4	$2^{-5.29}$	$2^{-5.29}$	$2^{23.58}$	$2^{46.26}$	$2^{42.32}$ CP
1, 2, 8, 9	$(32 \cdot 8)^2$	112	112	16	0	$2^{-12.52}$	$2^{-12.58}$	$2^{38.04}$	$2^{61.23}$	$2^{41.80}$ CP
Known-Plaintext Differential Bias Attack										
None	$(32 \cdot 11)^2$	179	176	5	4	$2^{-7.79}$	$2^{-7.74}$	$2^{28.58}$	$2^{52.12}$	$2^{43.45}$ KP
Chosen-Ciphertext Biased state Attack										
BB round	N'_{all}	$N'^c_{bias_p}$	$N'^w_{bias_p}$	-	-	q	$q^{(e)}$	N_{sample}	T	D
None	$(32 \cdot 11)$	1	0	-	-	$2^{-0.46}$	$2^{-0.46}$	$2^{11.92}$	$2^{23.92}$	$2^{23.92}$ CC

Table 10. Evaluation for byte-aligned leakage with noise ($N_s = (N_{all})^{1/2} \times \pi^{-1}$)

BB round	Time	Data	Time	Data	Time	Data	Time	Data
	$\pi = 1$		$\pi = 2^{-10}$		$\pi = 2^{-20}$		$\pi = 2^{-30}$	
Chosen-Plaintext Differential Bias Attack								
None	$2^{46.58}$	$2^{42.45}$ CP	$2^{56.58}$	$2^{52.45}$ CP	$2^{66.58}$	$2^{62.45}$ CP	$2^{76.58}$	$2^{72.45}$ CP
1, 2, 8, 9	$2^{61.23}$	$2^{41.80}$ CP	$2^{71.23}$	$2^{51.80}$ CP	$2^{81.23}$	$2^{61.80}$ CP	$2^{91.23}$	$2^{71.80}$ CP
Known-Plaintext Differential Bias Attack								
None	$2^{50.28}$	$2^{43.45}$ KP	$2^{60.28}$	$2^{53.45}$ KP	$2^{70.28}$	$2^{63.45}$ KP	$2^{80.28}$	$2^{73.45}$ KP
Chosen-Ciphertext Biased state Attack								
None	$2^{23.92}$	$2^{23.92}$ CC	$2^{43.92}$	$2^{43.92}$ CC	$2^{63.92}$	$2^{63.92}$ CC	$2^{83.92}$	$2^{83.92}$ CC

backward direction, 6- to 9- round attacks on AES-128 are corresponded to 8- to 11-round ones on AES-192 and 10- to 13- round ones on AES-256, respectively.

8.2 Multiple-bit leakage

Here we consider the case where M bits of the bit-aligned state information leak in each execution for a small M. Let $Z_1^i, Z_2^i, \ldots, Z_M^i$ be M leaked bits of the i-th execution.

Bitwise Multiset Attack: Assume that $Z_0^i, Z_1^i, \ldots, Z_{M-1}^i$ come from different but fixed locations of the state. If the XOR sum of 2^8 multiset of each location is zero, the XOR-sum of all set of $2^8 \times M$ bits is also zero. Thus, bitwise multiset attacks are feasible as long as leaked bits come from space where each XOR sum is zero only in a correct key. Time and date complexities are almost the same.

Differential Bias Attack: Assume that $Z_1^i, Z_2^i, \ldots, Z_M^i$ come from randomly-chosen different locations of the state. Since the attacker is able to obtain M bits in each execution, the required data reduces by a factor of M.

8.3 Other Granularities

So far, we have assumed that a leak can only occur after a full round. However, in other granularities such as leaks after SubBytes or MixColumns, our bitwise multiset attacks and differential bias attack still work.

Bitwise Multiset Attack: According to Proposition 1, any bit of the states between #3 and #10 has the zero-sum property if the key is correctly guessed. Using the difference of zero-sum properties between correct and wrong key cases, bitwise multiset attacks are applicable to other states in the same manner.

Differential Bias Attack: By properly choosing attack parameters, our differential bias attacks are also made feasible. For instance, if bits of the states after SubBytes are additionally leaked, the parameters of chosen-plaintext differential attacks on AES-128 with the space and time randomization are estimated as $N_{all} = (256 \times 11 + 128 \times 10)^2$, $N_{bias_p}^{(c)} = 2032$ $(= 216 + 216 + 1408 + 96 + 96)$, $N_{bias_p}^{(w)} = 1792$ $(= 96{+}96{+}1408{+}96{+}96)$, $N_{bias_n}^{(c)} = 400$ $(= 168{+}168{+}0{+}32{+}32)$, $N_{bias_n}^{(w)} = 64$ $(= 0 + 0 + 32 + 32)$, and $q = 2^{-16.10}$. The number of required samples is estimated as $N_{sample} = 2^{38.02} (= 2^6 \times 2^{16.01 \cdot 2})$. With $N_s = (N_{all})^{1/2}$, time complexity is $2^{57.02}$ $(= (2^{31} \times 2^{38.02})/(256 \times 11 + 128 \times 10))$ encryptions and the required data is 2^{46} $(= 2^{34} \times (256 \times 11 + 128 \times 10))$ chosen plaintexts.

References

1. D. Boneh, R. A. DeMillo, and R. J. Lipton. On the Importance of Eliminating Errors in Cryptographic Computations. *J. Cryptology*, vol. 14 (2), pages, 101–119, 2001.
2. E. Biham. New Types of Cryptanalytic Attacks Using Related Keys. *J. Cryptology*, vol. 7 (4), pages, 229–246, 1994.

3. A. Biryukov. The Design of a Stream Cipher LEX. In *SAC 2006*, LNCS, vol. 4356, pages 67–75, 2007.
4. A. Biryukov and A. Shamir. Structural Cryptanalysis of SASAS. *J. Cryptology*, vol. 23 (4), pages, 505–518, 2010.
5. A. Bogdanov, I. Kizhvatov and A. Pyshkin. Algebraic Methods in Side-Channel Collision Attacks and Practical Collision Detection. In *INDOCRYPT 2008*, LNCS, vol. 5365, pages 251–265, 2008.
6. J. Borghoff, A. Canteaut, T. Güneysu, E. B. Kavun, M. Knezevic, L. R. Knudsen, G. Leander, V. Nikov, C. Paar, C. Rechberger, P. Rombouts, S. S. Thomsen, and T. Yalçin. PRINCE - A Low-Latency Block Cipher for Pervasive Computing Applications - Extended Abstract. In *ASIACRYPT 2012*, LNCS, vol. 7658, pages 208–225, 2012.
7. S. Chow, P. A. Eisen, H. Johnson, and P. C. van Oorschot. White-Box Cryptography and an AES Implementation. In *SAC 2002*, LNCS, vol. 2595, pages 250–270, 2002.
8. J. Daemen and V. Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002.
9. I. Dinur and A. Shamir. Side Channel Cube Attacks on Block Ciphers. Cryptology ePrint Archive, Report 2009/127, 2009. `http://eprint.iacr.org/`.
10. O. Dunkelman and N. Keller A New Attack on the LEX Stream Cipher. In *ASIACRYPT 2008*, LNCS, vol. 5350, pages 539–556, 2008.
11. FIPS PUB 197, Advanced Encryption Standard (AES), 2001. U.S.Department of Commerce/National Institute of Standards and Technology.
12. T. Kleinjung, A. K. Lenstra and D. Page and N. P. Smart. Using the Cloud to Determine Key Strengths. In *INDOCRYPT 2012*, LNCS, vol. 7668, pages 17–39, 2012.
13. P. C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In *CRYPTO '96*, LNCS, vol. 1109, pages 104–113, 1996.
14. P. C. Kocher, J. Jaffe, and B. Jun. Differential Power Analysis. In *CRYPTO '99*, LNCS, vol. 1666, pages 388–397, 1999.
15. I. Mantin. Predicting and Distinguishing Attacks on RC4 Keystream Generator. In *EUROCRYPT 2005*, LNCS, vol. 3494, pages 491–506, 2005.
16. I. Mantin and A. Shamir. A Practical Attack on Broadcast RC4. In *FSE 2001*, LNCS, vol. 2355, pages 152–164, 2001.
17. S. Micali and L. Reyzin. Physically Observable Cryptography (Extended Abstract). In *TCC 2004*, LNCS, vol. 2951, pages 278–296, 2004.
18. Y. De Mulder. White-Box Cryptography: Analysis of White-Box AES Implementations. PhD thesis, KU Leuven, 2014.
19. Y. Oren, M. Renauld, F.-X. Standaert, and Avishai Wool. Algebraic Side-Channel Attacks Beyond the Hamming Weight Leakage Model. In *CHES 2012*, LNCS, vol. 7428, pages 140–154, 2012.
20. M. Renauld and F.-X. Standaert. Representation-, Leakage- and Cipher-dependencies in Algebraic side-channel attacks. in the proceedings of the ACNS 2010 Industrial Track, 2010.
21. M. Renauld, F.-X. Standaert, and N. Veyrat-Charvillon. Algebraic Side-Channel Attacks on the AES: Why Time also Matters in DPA. In *CHES 2009*, LNCS, vol. 5747, pages 97–111, 2009.
22. K. Schramm, T. J. Wollinger, and C. Paar. A New Class of Collision Attacks and Its Application to DES. In *FSE 2003*, LNCS, vol. 2887, pages 206–222, 2003.
23. P. Sepehrdad, S. Vaudenay, and M. Vuagnoux. Statistical Attack on RC4 - Distinguishing WPA. In *EUROCRYPT 2011*, LNCS, vol. 6632, pages 343–363, 2011.