Conditional Linear Cryptanalysis of the Advanced Encryption Standard

Takke, Erik C.

Award date:
2022

Department of Mathematics and Computer Science
Coding Theory and Cryptology

Conditional Linear Cryptanalysis of the Advanced Encryption Standard

Master's thesis

E.C. Takke

29 September 2022

Supervision:	*Assessment committee*:
T. Ashur	T. Ashur
B. Škorić	B. Škorić
	A. Ravagnani

Credits: 45

This is a public Master's thesis.

This Master's thesis has been carried out in accordance with the rules of the TU/e Code of Scientific Conduct.

Abstract

We develop two novel techniques to approximate a vectorial Boolean function; the weighted approximation and conditional approximation. When applied to the inversion function on $\mathbb{F}_{2^8}$, both of these approximations yield a greater correlation than a traditional, linear approximation. Since this inversion function is used in abundance in the Advanced Encryption Standard, it is shown how the conditional approximation technique can be used to construct a statistical distinguisher for four-round AES in the known-plaintext model. The existence of this distinguisher was previously ruled out on the basis of a security argument of the Wide Trail Strategy — the framework underlying the AES's design — and thus demonstrates a caveat with this argument. In addition to a distinguishing attack, a novel key-recovery attack is launched, capable of extracting 32 bits of the key using only $2^{125.62}$ data.

Contents

Chapter 1

Introduction

Communication is an essential part to our daily lives that increasingly takes place in the digital space. Although it has many advantages, a major disadvantage of the internet in this regard is the relative ease with which a third party is able to observe any and all data that is sent across the network. To prevent any such party from reading along with what is communicated, the vast majority of internet traffic today is encrypted before sending and decrypted upon receipt. Under the assumption that it is not possible to retrieve any information about the original message from its encryption without knowing the encryption key, this method guarantees secrecy of the message, thus enabling a private conversation.

Cryptanalysis plays a critical part in validating this assumption. This field of research investigates existing encryption methods and attempts to develop techniques to break the secrecy guarantee. The rationale behind this research is that, even though it is preferable that no such techniques are ever discovered, we prefer to find them ourselves and take appropriate action before a malicious third party manages to exploit them for their own gain.

In this thesis we add to the cryptanalysis of the Advanced Encryption Standard (AES) [17]: a prevalent encryption algorithm that finds its use in a variety of protocols and applications, including Transport Layer Security (TLS), Wi-Fi, Bluetooth, and debit and credit card payments [39]. In particular, we construct a statistical distinguisher for AES reduced to four rounds; to our knowledge the first of its kind for this cipher. In addition to demonstrating that this round-reduced cipher is vulnerable to a linear attack, this distinguisher exposes an inherent weakness of the Wide Trail Strategy — the framework underlying the cipher's design. As such, this distinguisher does not only pose a threat against encryption algorithms incorporating 4-round AES as a subroutine, but also against other ciphers constructed using this strategy.

Working towards creating this distinguisher, we start with bringing the reader up to speed with notation used throughout and concepts that lie at the core of this work in Chapter 2. This includes a brief overview of the AES encryption algorithm and as well as an introduction to Linear Cryptanalysis. In Chapter 3, we recall the linear properties of the inversion function on finite fields. This function forms the only source of non-linear behaviour in the AES cipher and has the purpose of thwarting statistical attacks, relying, e.g., on linear cryptanalysis. Following this, the weighted approximation and conditional approximation techniques are presented, which outperform the traditional linear approximation technique when applied to the inversion function. The implications of this result for the AES are presented in Chapter 4. This starts with recalling the Wide Trail Strategy and the security argument it puts forth, and is followed by the construction of the distinguisher and subsequently a distinguishing attack. It is furthermore demonstrated how the distinguisher can be adapted to construct a key-recovery attack capable of retrieving 32 bits of information using $2^{125.61}$ data. This chapter is then closed by the experimental validation of the discovered distinguisher. This thesis concludes with Chapter 5, where the broader implications of this attack are discussed and research directions for future work are proposed.

Chapter 2

Preliminaries

We briefly introduce the notation used throughout this thesis and the concepts that lie at its foundation.

2.1 Notation

In introducing the notation, it is assumed that the reader is acquainted with the concepts of finite fields and vector spaces. An introduction to these concepts can be found in [34]. We refer to [17] for an introduction to their application in the domain of Linear Cryptanalysis.

Let $\mathbb{F}_2$ denote the unique finite field on two elements, which we denote by 0 and 1. We will specifically refer to this field as the *binary field* and call an element in its underlying group a *bit*. We use $\oplus : \mathbb{F}_2 \to \mathbb{F}_2$ to denote the addition operation on this field, which is commonly referred to as the *exclusive-or* or *xor* function; $\wedge : \mathbb{F}_2 \to \mathbb{F}_2$ is used to denote the multiplication operation, which is also known as the *and* operation. We use $\mathbb{F}_{2^m}$ to denote the finite field on 2^m elements, which we represent with bit-strings of length m. To shorten the notation of these bit strings, we make use of the hexadecimal numeral system. This system represents the element $\texttt{10100111}_\texttt{b} \in \mathbb{F}_{2^8}$ as $\texttt{A7}_\texttt{x}$, for example. In the special case that $m = 8$, we refer to the elements in the group underlying $\mathbb{F}_{2^m}$ as *bytes*.

The n-dimensional vector space over $\mathbb{F}_{2^m}$ is denoted by $\mathbb{F}_{2^m}^n$. For a vector $v \in \mathbb{F}_{2^m}^n$, $v_i \in \mathbb{F}_{2^m}$ is used to indicate the ith coefficient of v, with $1 \leq i \leq n$. The operator $\oplus : \mathbb{F}_{2^m}^n \times \mathbb{F}_{2^m}^n \to \mathbb{F}_{2^m}^n$ is used to denote the addition of vectors and is defined as $a \oplus b := (a_1 + b_1, a_2 + b_2, \ldots, a_n + b_n)$, where $a, b \in \mathbb{F}_{2^m}^n$ are two n-dimensional vectors and $+$ denotes the addition operation in the underlying field. Note that $\oplus$ is used to denote both addition on $\mathbb{F}_2$ as well as $\mathbb{F}_{2^m}^n$. Moving forward, it will be clear from context which of the two is applied. Note furthermore that $\mathbb{F}_2^n$ is isomorphic to $\mathbb{F}_{2^n}$, as illustrated by the isomorphism $\mathbb{F}_{2^n} \to \mathbb{F}_2^n : b_n \cdots b_2 b_1 \mapsto (b_n, \ldots, b_2, b_1)$.

A *vectorial Boolean function* $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ with $n, m \in \mathbb{N}_+$ maps an n-dimensional bit-vector to an m-dimensional bit-vector. When $n = m$ and F is invertible, F is called a *permutation*. In the special case that $m = 1$, the function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ is said to be a *Boolean function*. Note that a vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ can be viewed as a vector of Boolean functions $(f_1, \ldots, f_m)$ acting on the same input, with $f_i : \mathbb{F}_2^n \to \mathbb{F}_2$ for $i = 1, \ldots, m$. We commonly represent Boolean functions with lowercase letters, while capital letters are used to represent their vectorial counterpart.

A binary function $F : \mathbb{F}_2^n \times \mathbb{F}_2^m \to \mathbb{F}_2^r$, with $n, m, r \in \mathbb{N}_+$, is a vectorial Boolean function that receives two bit-vectors and returns a third. A relevant example of such a function is the inner product $a^\top x : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2$. For vectors $a, x \in \mathbb{F}_2^n$, this product is defined as $a^\top x := a_1 \wedge x_1 \oplus \ldots \oplus a_n \wedge x_n$. Note that this value expresses the *parity* of the bits x_i for which $a_i = 1$. Observe moreover that $a^\top x$ can be viewed as a Boolean function on x when the value of a is fixed. We refer to this particular family of functions as the *parity functions*. When discussing these parity functions, we commonly refer to a as the *mask* of x.

An important property of the parity functions is their linearity. A Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ is *linear* if $f(x \oplus y) = f(x) \oplus f(y)$ for all $x, y \in \mathbb{F}_2^n$. Analogously, a vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is said to be linear when all Boolean functions $f_1, \ldots, f_m : \mathbb{F}_2^n \to \mathbb{F}_2$ composing it are linear. Since the parity functions are the only Boolean functions that are linear, there thus exist $u_1, \ldots, u_m \in \mathbb{F}_2^n$ such that this function F can be decomposed

into a vector of functions $(u_1^\top x, \ldots, u_m^\top x)$. Observe moreover that for the matrix $M_F \in \mathbb{F}_2^{m\times n}$ where u_i equals the ith row in M, $F(x) = M_F x$ for all $x \in \mathbb{F}_2^n$.

Another important property of the parity functions relates to their *imbalance* [9]. The imbalance of a Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ is defined as

$$Imb(f) := \frac{|\{x \in \mathbb{F}_2^n \mid f(x) = 0\}| - |\{x \in \mathbb{F}_2^n \mid f(x) = 1\}|}{2} = \frac{1}{2}\sum_{x\in\mathbb{F}_2^n} (-1)^{f(x)}, \tag{2.1}$$

where $|\cdot|$ maps a set to its size. When the imbalance of a function is 0, it is said to be *balanced*. In Lemma 2.1 it is demonstrated that the parity functions $a^\top x$ with $a \neq 0$ are balanced. Here, δ is used to denote the *kronecker-delta function*, which equals 1 if the input is zero and 0 otherwise.

Lemma 2.1 (Balanced function)**.** *Let $n \in \mathbb{N}_+$ and $a \in \mathbb{F}_2^n$ be arbitrary. It holds that $Imb(a^\top x) = \delta(a) \cdot \frac{|\mathbb{F}_2^n|}{2}$.*

Proof. The proof of this lemma is deferred to Appendix A.

The *correlation* between two Boolean functions $f, g : \mathbb{F}_2^n \to \mathbb{F}_2$ is defined as $C(f, g) := 2 \cdot \mathbb{P}[f(x) = g(x)] - 1$, where the probability is taken over all $x \in \mathbb{F}_2^n$. When the correlation between two functions is 0, these functions are said to be *uncorrelated*. Given an *input mask* $u \in \mathbb{F}_2^n$ and *output mask* $v \in \mathbb{F}_2^m$, the correlation between the masked input and output of a vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is expressed as

$$C(u^\top x, v^\top F(x)) = 2 \cdot \mathbb{P}\big[u^\top x \oplus v^\top F(x) = 0\big] - 1. \tag{2.2}$$

We use C^F to denote the *correlation matrix* of F [12]. In this matrix, entry $C^F_{v,u}$ is defined as $C(u^\top x, v^\top F(x))$ and thus acts as convenient short-hand notation for this correlation. By applying the *Walsh-Hadamard transform* to F, an equation is found that provides convenient means for computing $C^F_{v,u}$. We present this result in more generalized form in Lemma 2.2.

Lemma 2.2. *Let $f, g : \mathbb{F}_2^n \to \mathbb{F}_2$ be arbitrary Boolean functions. The correlation $C(f, g)$ of these two functions can be computed as*

$$C(f, g) = \frac{1}{|\mathbb{F}_2^n|}\sum_{x\in\mathbb{F}_2^n} (-1)^{f(x)\oplus g(x)}. \tag{2.3}$$

Proof. The proof of this lemma is deferred to Appendix A.

It follows from this lemma that $C^F_{v,w}$ can be computed as

$$C^F_{v,u} = C(u^\top x, v^\top F(x)) = \frac{1}{|\mathbb{F}_2^n|}\sum_{x\in\mathbb{F}_2^n} (-1)^{u^\top x\oplus v^\top F(x)}. \tag{2.4}$$

2.2 Cryptographic primitives

This work operates in the field of *symmetric-key cryptography*. In this field of research a wide variety of cryptographic primitives is used, including but not limited to *cryptographic hash functions*, *stream ciphers* and *block ciphers*. Although the results of this thesis could find use for multiple of these primitives, we restrict ourselves to discussing the implications for block ciphers.

The block cipher is a variation on the *product cipher* introduced by Shannon [55] operating on fixed-length Boolean vectors called *blocks*. More formally, we define a block cipher as a pair of vectorial Boolean functions (E, D) with $E, D : \mathbb{F}_2^n \times \mathbb{F}_2^\kappa \to \mathbb{F}_2^n$. In this tuple, E is called the *encryption function* and D the *decryption function*. When $y = E(x, k)$, we refer to $x \in \mathbb{F}_2^n$ as the *plaintext*, $k \in \mathbb{F}_2^\kappa$ as the *master key* and $y \in \mathbb{F}_2^n$ as the *ciphertext of x under key k*. The encryption and decryption functions form each others functional inverses as $D(E(x, k), k) = x$ holds for all x and k. We introduce $E_k(x) = E(x, k)$ and $D_k(x) = D(x, k)$ as shorthand notation for these functions, which allows this inversion relation to be expressed more clearly as $D_k \equiv E_k^{-1}$ for all k.

A relevant variant of the block cipher is the *iterated block-cipher*. For ciphers of this form, the encryption function E_k can be conveniently described as $E_k = F_{k_r} \circ \cdots \circ F_{k_1}$; the repeated application of a vectorial Boolean function

$F_{k_i}(x) = F(x, k_i) : \mathbb{F}_2^n \times \mathbb{F}_2^{\kappa'} \to \mathbb{F}_2^n$. Here, each invocation of F is referred to as a *round* and hence F is called the *round function* of the cipher; the keys $k_i \in \mathbb{F}_2^{\kappa'}$ for F_{k_i} are referred to as *round keys*. Often, an r-round iterated block cipher is bundled with a *Key Expansion Algorithm (KEA)* used to extract the round keys k_i from the master key k. Note that the size κ' of the round keys may differ from the size κ of the master key. It is moreover commonplace to add a *key whitening* [53] step to the cipher, where the encryption is then computed as $y = k_{r+1} \oplus E_k(x \oplus k_0)$.

We mention two important subcategories of the iterated block cipher. First is the *key-alternating block cipher*. For ciphers in this category, the round function can be decomposed into a function F, the behaviour of which is independent of the key, and a function X_k adding the round key k to the state according to some addition operation. Second is the *Substitution-Permutation Network (SPN)* [20]. In an SPN, the round function F_k can be decomposed into the consecutive application of a non-linear *substitution* function S, and a linear *permutation* function P.

2.3 The Advanced Encryption Standard

We introduce the standardized version of the Rijndael block cipher [17], which is better known as the *Advanced Encryption Standard*, or AES [18]. We commence with a brief summary of its history and follow this up with a high-level overview of the cipher structure. Lastly, the three functions that lie at the heart of the algorithm are expanded upon.

2.3.1 History

Shortly after Matsui broke DES in 1993 [35], the U.S. National Institute of Standards and Technology (NIST) enlisted help from the public in "specifying an encryption algorithm capable of protecting sensitive government information well into the 21st century" [43], which attracted fifteen contestants. Following the end of the submission phase, NIST invited the public to analyse and attack the competitors. Based on these analyses and attacks it was decided to release ten contestants from the competition, allowing cryptanalysts to focus on the five finalists. Following this second round of rigorous testing, NIST ultimately decided in October 2000 to base their Advanced Encryption Standard (AES) [18] on the Rijndael cipher [17]. This block-cipher, designed by Belgian cryptographers Daemen and Rijmen, was chosen for its high level of security, while not compromising on speed and versatility across various platforms. Since its official release in 2001, AES has seen increased use over the years, making it a widely used encryption standard today [39].

2.3.2 Structure

The AES is a block cipher best described as an r-round key-alternating substitution-permutation network. Its encryption function can be expressed as the composition

$$\texttt{AES}_k^r := \texttt{A}_{k_r} \circ \texttt{RF}' \circ \texttt{A}_{k_{r-1}} \circ \texttt{RF} \circ \cdots \circ \texttt{RF} \circ \texttt{A}_{k_1} \circ \texttt{RF} \circ \texttt{A}_{k_0}, \tag{2.5}$$

where $\texttt{RF}, \texttt{RF}' : \mathbb{F}_2^{128} \to \mathbb{F}_2^{128}$ denote the round functions and $\texttt{A}_{k_i} : \mathbb{F}_2^{128} \to \mathbb{F}_2^{128} : x \mapsto x \oplus k_i$ denotes the addition of round key k_i. In the first $r-1$ rounds, the round function `RF` is used, while the last round applies `RF'`. Here, `RF` involves the consecutive application of the three vectorial Boolean functions `SubBytes`, `ShiftRows` and `MixColumns`, while `RF'` is only composed of the functions `SubBytes` and `ShiftRows`. To simplify the discussion, it will be assumed from this point forward that the round function `RF` is used for *all* rounds. The addition of `MixColumns` to this last round will not affect the results presented in this thesis.

The standardized cipher specifies three AES variants — AES-128, AES-192 and AES-256 — differing in two aspects. First, the variants involve a different number of rounds r, namely 10, 12 and 14 respectively. Second, the length of the master key k used are 128, 192 and 256 bits. AES includes a specification for the key expansion algorithm used to generate the individual round keys k_i from the master key. In Chapter 4 we will use the fact that the algorithm reuses the master key as initial round key k_0 for AES-128. Further details of the algorithm will not be investigated in this work. Instead, it is assumed that the round keys generated by this algorithm are statistically independent. This assumption is commonly known as the *stochastic equivalence hypothesis* [32].

In the following, a brief review of the `SubBytes`, `ShiftRows` and `MixColumns` functions is provided and their application is illustrated. In these illustrations, it is most convenient to represent the plaintext, ciphertext and any intermediate encryption state using a 4×4 byte matrix. An example of this representation is provided in Figure 2.1. This example demonstrates that the 128-bit state may additionally be viewed as a vector $(p_0, \ldots, p_{15}) \in \mathbb{F}_{2^8}^{16}$

storing the elements of the matrix in column-major order. Since the inverse functions of `SubBytes`, `ShiftRows` and `MixColumns` follow trivially from their definitions, these will not be discussed.

p_0	p_4	p_8	p_{12}
p_1	p_5	p_9	p_{13}
p_2	p_6	p_{10}	p_{14}
p_3	p_7	p_{11}	p_{15}

Figure 2.1: A convenient representation of the AES state.

2.3.3 SubBytes (SB)

The vectorial Boolean function $\texttt{SubBytes} : \mathbb{F}_2^{128} \to \mathbb{F}_2^{128}$ forms the *substitution part* of this SPN. It transforms the state by applying the same substitution function $S : \mathbb{F}_2^8 \to \mathbb{F}_2^8$ to each byte. We will often refer to this function as the AES *s-box*. A visual representation of the operation is shown in Figure 2.2. The s-box S involves the

$\texttt{SubBytes}:$

p_0	p_4	p_8	p_{12}
p_1	p_5	p_9	p_{13}
p_2	p_6	p_{10}	p_{14}
p_3	p_7	p_{11}	p_{15}

$\mapsto$

$S(p_0)$	$S(p_4)$	$S(p_8)$	$S(p_{12})$
$S(p_1)$	$S(p_5)$	$S(p_9)$	$S(p_{13})$
$S(p_2)$	$S(p_6)$	$S(p_{10})$	$S(p_{14})$
$S(p_3)$	$S(p_7)$	$S(p_{11})$	$S(p_{15})$

Figure 2.2: Illustration of the application of `SubBytes` to the state.

consecutive application of the three vectorial Boolean functions `inv`, `L` and `T`. The function $\texttt{inv} : \mathbb{F}_{2^8} \to \mathbb{F}_{2^8}$ maps each element in $\mathbb{F}_{2^8} \setminus \{0\}$ to its multiplicative inverse, and 0 to itself. The reduction modulus used for this inverse operation is $m := 100011011_{\texttt{b}}$, which was chosen by the designers due to [34, 45]. Following `inv`, the linear transformation `L` is applied to the state, which is best described as a vectorial Boolean function $\mathbb{F}_2^8 \to \mathbb{F}_2^8$. The matrix $M_{\texttt{L}}$ associated with `L` is listed in Equation 2.6. The third component in the composition is the affine translation $\texttt{T} : \mathbb{F}_2^8 \to \mathbb{F}_2^8 : x \mapsto x \oplus 63_{\texttt{x}}$.

$$M_{\texttt{L}} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.6}$$

The `SubBytes` function is the only non-linear component contained in the round function `RF`. As such, one of its purposes is to thwart linear cryptanalysis against the cipher. It is in particular the properties of the non-linear function `inv` that deter linear attacks. Because we will analyse the linear properties of the `inv` function, it may be more insightful to ignore the algebraic structure underlying this function and instead view it as a look up table. The tables for both `inv` and S can be found in Appendix B.

2.3.4 ShiftRows (SR)

The `ShiftRows` operation is a linear function permuting the bytes in the state. Together with the `MixColumns` function, this operation forms the permutation part of the substitution-permutation network. As illustrated in Figure 2.3, this operation cyclically shifts the bytes in the ith row of the state i steps to the left, where indexing starts at the top row with 0. Note here that the top row is shifted 0 bytes to the left and thus remains unchanged, while on the bottom row all bytes are cyclically shifted three places to the left.

ShiftRows:

p_0	p_4	p_8	p_{12}
p_1	p_5	p_9	p_{13}
p_2	p_6	p_{10}	p_{14}
p_3	p_7	p_{11}	p_{15}

$\mapsto$

p_0	p_4	p_8	p_{12}
p_5	p_9	p_{13}	p_1
p_{10}	p_{14}	p_2	p_6
p_{15}	p_3	p_7	p_{11}

Figure 2.3: Illustration of the application of `ShiftRows` to the state.

2.3.5 `MixColumns` (`MC`)

The `MixColumns` function is a linear operation forming the second half of the permutation part of this SPN. In the setting of this thesis, this operation is best described as a linear transformation $\mathtt{H} : \mathbb{F}_2^{32} \to \mathbb{F}_2^{32}$ that is applied to each of the four 32-bit columns in the state. Since `H` is linear, there exists a matrix $M_{\mathtt{H}} \in \mathbb{F}_2^{32\times 32}$ s.t. $\mathtt{H}(x) = M_{\mathtt{H}}x$ for all $x \in \mathbb{F}_2^{32}$. This matrix it is presented in Appendix C and clearly illustrates the 'mixing' behavior of this function. It shows in particular that each bit in the output is the linear combination of at least five and at most seven bits of the input, where from each of the four bytes in the input at least one bit is used. We defer discussing the impact of this function on the linear cryptanalysis of AES to Section 4.1.

2.4 Cryptanalysis

Edgar and Manz [19] define cryptanalysis as "the process of studying cryptographic systems to look for weaknesses or leaks of information." This field of research can be split into the analysis of the implementation of a cryptographic system in software and hardware, and of the mathematics underlying the system. This thesis solely discusses the latter applied to block ciphers.

The process of cryptanalysing the mathematics underlying a cipher can be split into two steps. The first step involves uncovering a mathematical property of the cryptographic system that is unlikely to appear in a permutation uniformly sampled from the set of all permutations [56]. This is then followed up by an attempt to exploit this property to extract information about the cipher that should have been secret. When both steps are successful, an attack against the cipher under investigation has been discovered. Next, three important aspects to an attack are discussed: the model in which an attack resides, its complexity, and its type.

2.4.1 Attack model

The first aspect is the model in which the attack takes place [31, 56]. An attack model specifies the type of data an adversary must have access to in order to successfully perform the attack. Below, four commonly recognized models are presented, along with a brief description of the capabilities of the adversary in these models. Here, the capabilities of the adversary are expanding from one model to the next, resulting in a hierarchy. Typically, an attack is considered more severe when it can be performed in an attack model that is higher up in the list.

Known-ciphertext model. For attacks in the known-ciphertext model, it is assumed that the adversary only has access to a source generating ciphertexts; it can only use these and any mathematical property of the system to make their attack work. Although the model assumes that the adversary does not have access to the plaintexts corresponding with these ciphertexts, it is often still possible to make reasonable assumptions about the plaintexts based on the source from which they originate. For example, the adversary could make assumptions on the alphabet that the plaintexts are written in, based on the geoposition of the source.

Known-plaintext model. The known-plaintext model assumes the adversary has means to obtain arbitrary pairs of plaintext and ciphertexts, and subsequently leverage these in the attack. When used in practice, it is often the case that the situation allows part of the plaintext to be guessed with a high degree of certainty. It is, for example, often possible to correctly guess the value of certain header-fields of HTTP-packets [21].

Chosen-plaintext model. In this model it is assumed that the adversary additionally has access to a source that can obtain the ciphertexts of any set of plaintexts they choose. However, it is assumed that the adversary

only has one attempt to perform all the encryptions and must thus select all plaintexts to which the ciphertext is required ahead of observing the ciphertexts.

Adaptively chosen-plaintext model. This model is a subtle variation to the previous in that it assumes that the adversary can perform the encryption step several times, allowing the adversary to adapt its choice of plaintext to encrypt based on previous ciphertexts it received.

2.4.2 Complexity

The second aspect is the cost associated with performing an attack, which is directly linked to its complexity. This complexity is commonly decomposed into the following three factors [25]:

- **Time.** The time it takes to perform the attack, usually expressed in the number of calls to the cipher under attack.
- **Memory.** The amount of memory that is necessary to perform the attack, usually expressed in bytes.
- **Data.** The amount of data that is necessary to perform the attack.

Note that the smaller each of these three factors is, the cheaper it is to perform the attack. It is worth observing that the data-complexity often acts as a lower bound to the time complexity, as it is assumed that all data must be observed for the attack to succeed. Observe furthermore that the time and memory factors together allow us to estimate the practicality of an attack. When an attack can be performed in a reasonable time frame using the current state of technology, it is practical, while attacks whose resource requirements are beyond what is available today only serve a theoretical purpose. One should, however, not forget that improvements to the analysis or to the state of technology may make attacks that are only theoretical today a practical threat in the future. As holds the truism: "attacks always get better, they never get worse."

The complexity of an attack is commonly compared to that of an exhaustive key search. In this generic attack, the adversary attempts to find the secret key by decrypting a small set of ciphertexts using all keys in the key space. Since the adversary is certain to encounter the correct key at some point throughout the search, it is guaranteed that the ciphertexts are eventually decrypted and thus the key found. The complexity of this attack therefore acts as an upper bound for the complexity of attacks extracting any unknown information about the key; when an attack exceeds this bound, it is no longer considered an attack.

2.4.3 Attack type

The third aspect we consider is the attack type [31]. In this thesis we will encounter two different types: a *distinguishing attack* and a *key-recovery attack*. The goal of the former is to distinguish the cipher under investigation from a random permutation. This attack is often modelled as a game, which starts with the challenger generating a random bit, which they keep to themselves. Based on this bit, the challenger selects either the cipher under investigation with an arbitrary master key, or a random permutation acting on the same block size. The adversary is then tasked with determining the bit by analysing data samples it receives from the challenger. When the adversary correctly guesses the bit's value with probability strictly greater than 0.5, the adversary is said to have found a distinguishing property of the cipher.

In a key-recovery attack, a similar game is played. Here, the challenger is restricted to using the cipher under investigation, but may still generate a random master key, which they keep to themselves. The adversary again requests data from the challenger and analyses this, with the intent to recover information about the key. The attack is deemed successful when, using only information extracted from the data, the adversary is able to construct an ordering on the keys in the key space, such that the actual key is located in the first half with probability greater than 0.5.

2.5 Linear Cryptanalysis

In 1992, Matsui and Yamagishi [37] presented a novel cryptanalytic method for block ciphers. This method investigates the existence of linear relations between bits of the plaintext, bits of the ciphertext and bits of the encryption key, that hold with high probability. The presence of such relations in a cipher is undesirable, as the authors demonstrate that these can be used to attack the cipher in the known-plaintext model. As an illustration of its usefulness, the technique was applied to the FEAL cipher family [38], where for each member the attack proves capable of

extracting the encryption key significantly faster than an exhaustive search would. Matsui [35] later generalized the technique to be applicable to all block-ciphers and named it Linear Cryptanalysis. To distinguish this method from those that extended it over the years, we refer to the basic technique as *standard Linear Cryptanalysis*. To further illustrate its power, the technique was used to launch an attack against the Data Encryption Standard (DES) [42] — a cipher widely used at the time. This attack uncovered the full encryption key for 8-round DES using only 2^{21} plaintext-ciphertext pairs, while only 2^{47} pairs were required to extract the key when using full 16-round DES. Both attacks are orders of magnitude faster than an exhaustive key search.

2.5.1 The goal

Linear Cryptanalysis is concerned with using linear methods to approximate part of the non-linear behaviour of an encryption function. Standard Linear Cryptanalysis attempts to discover relations between bits of the plaintext and bits of the ciphertext, which we refer to as *linear approximations*. For a block-cipher (E_k, D_k) operating on n-bit blocks, the relation between a plaintext x and its ciphertext $E_k(x)$ under key k is described by the equation

$$u^\top x = v^\top E_k(x), \tag{2.7}$$

where the linear combination of plaintext bits masked by $u \in \mathbb{F}_2^n$ is compared to the linear combination of bits in the ciphertext masked by $v \in \mathbb{F}_2^n$. For a given key k, this equality holds with some probability

$$p_k = \mathbb{P}[u^\top x = v^\top E_k(x)] = \mathbb{P}[u^\top x \oplus v^\top E_k(x) = 0], \tag{2.8}$$

where it is assumed that x is chosen uniformly at random from the plaintext space. We are interested in the deviation of p_k from $\frac{1}{2}$, which we capture with $C_{v,w}^{E_k} := 2 \cdot p_k - 1$; the correlation between the sum of selected bits in the input and selected bits in the output of E_k. The goal of Linear Cryptanalysis is to discover a linear approximation $\langle u, v \rangle$ for which the absolute correlation, or *magnitude*, with the encryption function E_k is large for almost all keys [27, 35], as approximations of this type can be exploited to attack the encryption function under investigation.

2.5.2 Constructing a linear approximation

When attempting to construct a linear approximations for block ciphers in general, computing the exact value of the correlation associated with the approximation is the most difficult part [27]. However, this step is considerably easier when only iterated block ciphers are concerned. It namely holds that linear approximations for such ciphers are straightforwardly decomposed into a set of *linear trails* [5, 26, 35], which can be constructed iteratively.

Let $E_k := F_{k_r} \circ \cdots \circ F_{k_1}$ denote an r-round iterated block cipher on n-bit blocks. Following [26], a linear trail is defined as a mask tuple $\lambda := (\lambda_1, \ldots, \lambda_{r+1}) \in (\mathbb{F}_2^n)^{r+1}$ comprising the masks of r concatenated 1-round linear approximations $\langle \lambda_i, \lambda_{i+1} \rangle$ for the individual round functions F_{k_i}. Note that the output mask of the approximation for round function F_{k_i} is the input mask for the approximation of $F_{k_{i+1}}$. With $k = (k_1, \ldots, k_r)$ denoting the encryption key, the *correlation contribution* of trail λ is computed as

$$C_\lambda^k = \prod_{i=1}^{r} C(\lambda_i^\top x, \lambda_{i+1}^\top F_{k_i}(x)); \tag{2.9}$$

the product of the correlations associated with the linear approximations of the individual rounds. It then follows that the correlation of $\langle u, v \rangle$ over the block cipher is computed as

$$C_{v,u}^{E_k} = \sum_{\lambda; \lambda_1 = u, \lambda_{k+1} = v} C_\lambda^k; \tag{2.10}$$

the sum of the correlation contributions of the linear trails sharing their input mask and output mask with the linear approximation. This summation contains up to $2^{n(r-1)}$ non-zero correlation contributions, illustrating that further estimations are often necessary. Providing a closed form formula for such estimations is difficult for iterated block ciphers in general, as no assumptions can be made regarding the influence of the round key on the correlation of the 1-round approximation of each round function. When considering the more specific case of a key-alternating block-cipher, we find that the key only influences the sign of a linear trail's correlation contribution.

Let $E_k := X_{k_{r+1}} \circ F \circ X_{k_r} \circ \cdots \circ X_{k_2} \circ F \circ X_{k_1}$ denote an r-round key-alternating block cipher on n-bit blocks, where $X_k : \mathbb{F}_2^n \to \mathbb{F}_2^n : x \mapsto x \oplus k$ denotes the addition of round key k, and $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ denotes the round-function.

Observe that we can construct a linear trail for this function using $(2r+1)$ single-round approximations, and compute its correlation contribution using Equation 2.9. The expression of this equation can now be significantly simplified due to a result in [16]. Here, it is demonstrated that the correlation of a linear approximation $\langle u, v\rangle$ for the key addition function X_k is non-zero if and only if $u = v$. This result is recalled in Lemma 2.3.

Lemma 2.3 (Equation 7.34, [16]). *Let $X_k : \mathbb{F}_2^n \to \mathbb{F}_2^n$ s.t. $x \mapsto x \oplus k$ denote the affine translation for an arbitrary $k \in \mathbb{F}_2^n$. It holds that*

$$\forall u, v \in \mathbb{F}_2^n \setminus \{0\} : C_{v,u}^{X_k} = (-1)^{v^\top k} \cdot \delta(v \oplus u). \tag{2.11}$$

It follows from Lemma 2.3 and Equation 2.9 that the contribution of a linear trail is zero when it contains a 1-round approximation $\langle a, b\rangle$ for a key addition function for which $a \neq b$. It therefore suffices to restrict Equation 2.10 to the trails for which these masks are the same. Note that such linear trails can be represented as $\lambda = (\lambda_1, \ldots, \lambda_{r+1}) \in (\mathbb{F}_2^n)^{r+1}$, where each $\langle \lambda_i, \lambda_{i+1}\rangle$ is an approximation for the ith application of F, and $\langle \lambda_i, \lambda_i\rangle$ is an approximation for X_{k_i}. It then follows that the contribution of this trail can be computed as

$$C_\lambda^k = \left(\prod_{i=1}^{r+1} C_{\lambda_i,\lambda_i}^{X_{k_i}}\right)\left(\prod_{j=1}^{r} C_{\lambda_{j+1},\lambda_j}^{F}\right) = \left(\prod_{i=1}^{r+1}(-1)^{\lambda_i^\top k_i}\right)\left(\prod_{j=1}^{r} C_{\lambda_{j+1},\lambda_j}^{F}\right) = (-1)^{\bigoplus_{i=1}^{r+1}\lambda_i^\top k_i} \cdot \prod_{j=1}^{r} C_{\lambda_{j+1},\lambda_j}^{F}. \tag{2.12}$$

When we use $C_\lambda^F := \prod_{j=1}^r C_{\lambda_{j+1},\lambda_j}^F$ and $\lambda^\top k := \bigoplus_{i=1}^{r+1} \lambda_i^\top k_i$, this is more compactly written as

$$C_\lambda^k = (-1)^{\lambda^\top k} \cdot C_\lambda^F = (-1)^{\lambda^\top k} \cdot \mathtt{sgn}(C_\lambda^F) \cdot |C_\lambda^F|, \tag{2.13}$$

where $\mathtt{sgn} : \mathbb{R} \to \{-1, 1\}$ denotes the signum-function. This equation confirms that only the sign of the correlation is dependent on the key; the magnitude of a correlation contribution $|C_\lambda^F|$ is not treated as a random variable.

Matsui then approximates the correlation of the linear approximation $\langle u, v\rangle$ with the largest correlation contribution magnitude among all trails [35], an estimation that is only reasonable when the summation in Equation 2.10 is dominated by this largest contribution [46, 52]. When the summation contains multiple trails with large absolute contribution, it is dependent on the key whether the signs of these contributions are the same or different, and thus whether these contributions respectively amplify or cancel each other. As a result, the correlation magnitude of a linear approximation fluctuates under changing encryption keys when this is the case. This undesirable effect is best known as the *linear hull effect* [4, 41]. In order to proceed, one must thus assume this term to dominate the summation; an assumption we refer to as the *dominant trail hypothesis*.

2.5.3 Linear attacks

Under the dominant trail hypothesis, Matsui's work provides two techniques to attack a key-alternating block cipher by exploiting a linear approximation that yields a non-zero correlation with it. These key-recovery attacks are both located in the known-plaintext model and are called *Algorithm 1* and *Algorithm 2* [35]. Both methods are briefly discussed.

Algorithm 1. The first algorithm exploits Equation 2.13 to recover information about the key using the sign of the sample correlation. Let (E_k, D_k) be an r-round iterated block cipher on n-bit blocks with round function F for which the adversary constructs a linear trail $\lambda = (u, \lambda_2, \ldots, \lambda_r, v)$ with a large correlation contribution that is dominant for the correlation of the approximation $\langle u, v\rangle$. Given a sufficiently large set T of plaintext-ciphertext pairs encrypted with E_k under the same key $k = (k_1, \ldots, k_{r+1})$, the adversary can compute an approximation of the correlation $C_{v,u}^{E_k}$ as

$$\hat{C} = \frac{1}{|T|} \sum_{(x,y)\in T} (-1)^{u^\top x \oplus v^\top y}; \tag{2.14}$$

the sample correlation. In addition, the adversary can pre-compute the value of C_λ^F, since this value is not influenced by the key. Because C_λ^k is assumed to be the dominant contribution to $C_{v,u}^{E_k}$, it then follows from Equation 2.13 that

$$(-1)^{\lambda^\top k} = C_\lambda^k / C_\lambda^F \approx C_\lambda^{E_k} / C_\lambda^F \approx \hat{C} / C_\lambda^F, \tag{2.15}$$

thus making it possible for the adversary to make an informed guess of the value of $\lambda^\top k =: b$. Given that the function $\lambda^\top x$ is balanced, only half of all keys k will yield $\lambda^\top k = b$, while $\lambda^\top k = b \oplus 1$ will be observed for the other half. As such, this technique enables the adversary to effectively cut the search space in half when the bit is correctly guessed.

Algorithm 2. Whereas Matsui's first attack focuses on observing the sign of the correlation, the second instead leverages its magnitude to extract information about the key, in particular the round key of the last round. To perform this attack on the r-round cipher, the adversary uses an (r-1)-round linear trail $\lambda = (u, \lambda_2, \ldots, \lambda_{r-1}, v)$ with large correlation contribution that is dominant for the approximation $\langle u, v \rangle$. Let T denote the set of plaintext-ciphertext pairs available to the attacker. For each of the last-round keys k under consideration, the adversary constructs the set $T_k := \{(p, D_k^1(c)) \mid (p, c) \in T\}$, where $D_k^1(y)$ denotes the 1-round decryption of y under key k. For each of these sets, the adversary then computes the sample correlations $\hat{C}_k$ of the linear approximation $\langle u, v \rangle$. Since one of the sets T_k corresponds with the actual last-round key k^*, this set now contains plaintexts and their (r-1)-round encrypted ciphertexts. It is therefore expected that the sample correlation magnitude $|\hat{C}_{k*}|$ for T_{k^*} is close to the trail's true correlation magnitude $|C_\lambda^F|$. For the sets corresponding with incorrect key guesses it is meanwhile assumed that the sample correlation is close to zero; an assumption known as the *wrong-key randomization hypothesis* [2, 10, 54]. As such, the adversary can construct an ordering of the round keys based on the magnitude of their set's sample correlation, where those with magnitude close to $|C_\lambda^F|$ are more likely to be the correct key. This ordering can then be used to search for the master key in a manner that is expected to be faster than an exhaustive key search.

2.5.4 Complexity

Not every linear approximation is suitable to successfully perform the Algorithm-1 and Algorithm-2 attacks: only when the magnitude of the correlation is large enough can the attacks succeed. To understand why this is the case, we must discuss the complexity of both attacks.

Before discussing this complexity, let us first recall the definition for the known-plaintext model in Section 2.4.1, which states that "*the known-plaintext model assumes the adversary has means to obtain arbitrary pairs of plaintext and ciphertexts* [...]". In the remainder of this thesis, we assume that 'arbitrary' is to mean that each plaintext-ciphertext pair observed by the adversary is sampled uniformly at random from all pairs. In particular, the pairs are sampled with replacement. Given a large enough sample size, it is thus expected that the adversary will encounter some pairs multiple times in its sample multi-set.

Algorithm 1. The time complexity of this attack is linear in the size of the data set: the adversary utilizes each plaintext-ciphertext pair once in computing the approximation $\hat{C}$ for $C_{v,w}^{E_k}$, after which $(-1)^{\lambda^\top k}$ is determined in constant time using Equation 2.13. Since each plaintext-ciphertext pair needs to be considered only once, this attack can be set up as a *streaming algorithm* [1], utilizing a constant size memory space. The data complexity of the attack is dependent on two factors: the correlation of the linear approximation used and the desired success probability of the attack. In [35] a formula relating the data complexity and these two facets has been presented. We recall this formula in Lemma 2.4.

Lemma 2.4 (Lemma 2, [35])**.** *Let t be the number of given random plaintext-ciphertext pairs and $|C_\lambda^F|$ be the absolute correlation contribution of the trail λ that dominates $\langle u, v \rangle$. Given that $|C_\lambda^F|$ is sufficiently small, the success rate p^* of Algorithm 1 is*

$$p^* = \int_{-2\sqrt{t}\cdot|C_\lambda^F|}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-x^2/2} dx = \Phi\Big(|C_\lambda^F| \cdot \sqrt{t}\Big), \tag{2.16}$$

where Φ denotes the cumulative distribution function of the standard normal distribution.

Proof. The proof of this lemma is deferred to Appendix A.

This result thus illustrates that for any constant success probability p^*, the data complexity is inversely proportional to the square of the correlation. This demonstrates why it is preferable for the adversary to construct a dominant linear trail with a correlation contribution magnitude that is as large as possible: an attack based on the linear approximation containing this trail requires the least amount of data — and thus time — to execute. We present the relation between p^* and t in Table 2.1, where the value of t is expressed in terms of C_λ^F. As an example, the table illustrates that when $t = 4(C_\lambda^F)^{-2}$, the algorithm is expected to succeed with probability 0.977. When we instead express C_λ^F in terms of t, it follows that C_λ^F must be greater than $t^{-1/2}$ for the algorithm to succeed with probability 0.841 or greater. Since the data complexity is naturally upper bounded by the size of the full code book, it follows that C_λ^F must be greater than $2^{-n/2}$ to be used effectively in an Algorithm-1 attack.

Table 2.1: Relation between sample size, correlation and probability of success for Algorithm 1.

t	$(C_\lambda^F)^{-2}$	$2(C_\lambda^F)^{-2}$	$4(C_\lambda^F)^{-2}$	$8(C_\lambda^F)^{-2}$
p^*	0.841	0.921	0.977	0.997

Algorithm 2. We also determine the time, memory and data complexity of an Algorithm-2 attack. The time complexity for this attack is m times the data complexity, with m the number of keys considered, since the sample correlation must be determined for each of these keys. The memory complexity equals $\mathcal{O}(m)$, since all m sample correlations must be stored to construct the ordering. The expression for the data complexity again depends on the success probability and correlation contribution of the trail, as well as two additional factors. The first of these is the *advantage* of an attack. We say that an Algorithm-2 attack obtains an $(\log_2(m) - \log_2(r))$-bit advantage over exhaustive search when the ordering it creates ranks the correct key among the top r in the ordering [54]. We will use $a := \log_2(m) - \log_2(r)$ to denote the advantage of an attack.

The second aspect is the specific wrong-key randomization hypothesis used in the analysis. As previously mentioned, the wrong-key hypothesis assumes that the magnitude of the sample correlation is distributed around zero for all T_k where k is not the correct round key. Selçuk [54] first hypothesized that the correlation for a wrong key would be exactly 0 and thus that the sample correlation would be a distributed according to $\mathcal{N}(0, 1/t)$, with $\mathcal{N}$ denoting the normal distribution. A derivation for this is distribution is found in the proof for Lemma 2.4 located in Appendix A. Under this version of the hypothesis, an expression was derived relating the success probability, correlation, data complexity and advantage to each other. We recall this expression in Lemma 2.5.

Lemma 2.5 (Theorem 2, [54])**.** *Let p^* be the probability that a linear attack on m key candidates with a linear approximation of correlation C and t known plaintext blocks, delivers an a-bit or higher advantage. Assuming that the linear approximation's correlation is independent for each key tried and is equal to 0 for all wrong keys, we have, for sufficiently large t,*

$$p^* = \Phi\Big(\sqrt{t} \cdot |C| - \Phi^{-1}\big(1 - 2^{-a-1}\big)\Big). \tag{2.17}$$

The lemma can furthermore be rewritten to express the data complexity t as

$$t = \left(\frac{\Phi^{-1}(p^*) + \Phi^{-1}(1 - 2^{-a-1})}{|C|}\right)^2. \tag{2.18}$$

Bogdanov and Tischhauser [10] later introduced a variation to the hypothesis, arguing that the correlation associated with a wrong key would be distributed as $\mathcal{N}(0, 2^{-n})$ for an n-bit block cipher. One would therefore expect the sample correlation $\hat{C}_k$ to be distributed as $\mathcal{N}(0, 1/t + 1/2^n)$ for incorrect key guesses k. Along with this hypothesis, the authors derived a different relation for the four factors, which we recall in Lemma 2.6.

Lemma 2.6. *Theorem 2, [10] Consider a linear attack with Matsui's Algorithm 2 on an n-bit block cipher ($n \geq 5$) using a linear approximation with correlation $C \neq 0$ and sufficiently large $t \leq 2^n$ known plaintexts. Denote by p^* the probability that this attack succeeds with an advantage of $a > 0$ bits over exhaustive key search. Then*

$$p^* \approx \Phi\Big(\sqrt{t} \cdot |C| - \sqrt{1 + t/2^n} \cdot \Phi^{-1}\big(1 - 2^{-a-1}\big)\Big), \tag{2.19}$$

Observe that the only difference with Lemma 2.5 is the $\sqrt{1 + t/2^n}$ term. This term has as an effect that the success probability of an attack is significantly reduced when its data complexity is very close to the size of the full code book 2^n. The authors have provided empirical evidence confirming their hypothesis to be a better fit than Selçuk's for the 80-bit ciphers SmallPresent [33] and RC6-5/8/10 [51]. However, for block ciphers with larger block sizes their hypothesis could not be verified due to the time complexity of the verification step becoming too high.

These complexity derivations demonstrate that a linear approximation with large correlation magnitude enables both Algorithm-1 and Algorithm-2 attacks on a cipher, thus illustrating that the existence of such approximations is highly undesirable. It is for this reason that modern ciphers include non-linear functions in their functional description to thwart these attacks. As alluded to in Section 2.3.3, the AES attempts to accomplish this using the non-linear `inv` function contained in the `SubBytes` step.

2.5.5 Extensions

Linear Cryptanalysis has proven a powerful and indispensable tool in the analysis of block ciphers. Following its inception, literature has produced many extensions attempting to improve upon the technique. A significant part of these extensions focus on combining multiple linear approximations, with the goal of constructing an approximation that achieves a correlation that is greater than can be achieved with the linear approximations individually. We briefly discuss the extensions in this category that are most relevant to this work.

The first extension of interest is Multiple Linear Cryptanalysis, a technique first explored by Kaliski and Robshaw [28] in 1994. This work demonstrated that several linear approximations involving the *same* key bits can be combined to extract information on the key while involving significantly fewer data compared to the basic technique. In 2004, Biryukov, De Cannière and Quisquater [8] observed that often times, only few such approximations exist, severely limiting the usability of the attack. Attempting to resolve this issue, it is shown that linear approximations involving *different* key bits can be combined to recover more information on the key, in an attack that has the same or lower complexity than one involving a single linear approximation. Murphy [40] has later criticized an assumption underlying both methods. He showed that linear approximations are not necessarily independent: a statement assumed true by the previous authors.

Hermelin, Cho and Nyberg [26, 27] introduced Multidimensional Linear Cryptanalysis in an attempt to resolve this issue. This work provides a novel technique expanding on its predecessor by adding the requirement that the linear approximations that are combined need to form a linear subspace. In [47] it is recognized by Nyberg that this requirement, although resolving the independence issue, often leads to the inclusion of many weak linear trails, thus weakening the approximation. The Affine Linear Cryptanalysis [3, 47] extension aims to resolve this by allowing all linear approximations in a half space to be discarded if these are considered invaluable to the attack.

Biham and Perle [6] have approached the use of multiple linear approximations from a different angle, introducing the concept of Conditional Linear Cryptanalysis. Instead of combining multiple approximations into one, their work demonstrates how the correlation of a linear approximation of the Feistel-cipher DES [42] can be improved by conditioning the data set. In this conditioning, the data set is partitioned based on a set of (other) linear approximations. By only considering the data in some of the parts, it is illustrated that the correlation of the linear approximation is significantly improved.

Chapter 3

Analysis of the inversion function

This chapter analyses the vectorial Boolean function `inv`, which was introduced in Section 2.3 as an integral part of the AES s-box. First, we recall in Section 3.1 the seminal work by Nyberg [45] which provides an upper bound of $2^{1-n/2}$ on the correlation magnitude of any linear approximation for the inverse X^{-1} in $\mathbb{F}_{2^n}$. We then recall the work of Keliher et al. [29, 30] which discusses the complete *Linear Approximation Table* (LAT) for the special case that $n = 8$, and conclude that Nyberg's upper bound is tight in this situation.

In symmetric-key folklore, these two results are understood to mean that any attempt to approximate this multiplicative inverse in $\mathbb{F}_{2^8}$ must incur a cost of 2^{-3} at the minimum. Conversely, this thesis presents in Section 3.2 two approximation techniques exhibiting a higher correlation when applied to `inv`. Finally, in Section 3.3 it is demonstrated how the approximations can be adapted when the function under investigation is used in composition with affine transformations. The discussion in this chapter is made from a mathematical point of view, and is purposefully kept abstract. The implications for symmetric-key cryptanalysis and in particular the AES, are deferred to Chapter 4.

3.1 Linear Approximations of the inversion function

Let $X^{-1} : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ denote the function that maps a finite field element to its multiplicative inverse. Nyberg proved in [45] that this function possesses properties that are highly desirable in cryptography. In particular, it is shown that the correlation of any linear approximation for this function is upper bounded by $2^{1-n/2}$. This property, among others, led Daemen and Rijmen to use this function with $n = 8$ as part of the Rijndael s-box [17]. Recall that this particular instance of the inversion function was previously introduced in Section 2.3 as $\texttt{inv} : \mathbb{F}_2^8 \to \mathbb{F}_2^8$.

We briefly discuss the linear properties of the `inv` function. To this end, we construct its correlation matrix $C^{\texttt{inv}}$. This matrix is isomorphic to the Linear Approximation Table of `inv`, which was previously discussed in [29, 30], and reveals a clear structure underlying the `inv` operator. In particular, it holds that for any output mask $v \in \mathbb{F}_2^8 \setminus \{0\}$, the distribution of values in column $C^{\texttt{inv}}_{v,\cdot}$ is fixed [29, Lemma 2]. We recall this distribution in Table 3.1. For an arbitrary $v \in \mathbb{F}_2^8 \setminus \{0\}$, this table lists each value of $C^{\texttt{inv}}_{v,u}$ along with the number of input masks ϕ for which this correlation is attained. Since `inv` is an involution, we additionally conclude that the same distribution arises when u is fixed and v varies instead.

Table 3.1: Frequency distribution of the correlation between linear combinations of bits in the `inv` input and output.

$C^{\texttt{inv}}_{v,u}$	$-\frac{7}{64}$	$-\frac{6}{64}$	$-\frac{5}{64}$	$-\frac{4}{64}$	$-\frac{3}{64}$	$-\frac{2}{64}$	$-\frac{1}{64}$	0	$\frac{1}{64}$	$\frac{2}{64}$	$\frac{3}{64}$	$\frac{4}{64}$	$\frac{5}{64}$	$\frac{6}{64}$	$\frac{7}{64}$	$\frac{8}{64}$
ϕ	8	16	8	18	24	16	32	17	16	20	16	16	16	20	8	5

Table 3.1 reveals that, regardless the choice of output mask v, there always exist exactly five input masks $u \in \mathbb{F}_2^8 \setminus \{0\}$ s.t. $C^{\texttt{inv}}_{v,u} = \frac{8}{64} = 2^{-3}$, where it should be noted that the exact values of these masks depend on the choice of v. Let us use $\Omega_v = \{\omega_1, \omega_2, \omega_3, \omega_4, \omega_5\}$ to denote the set containing these five masks moving forward. In Appendix D, the set Ω_v is listed for each $v \in \mathbb{F}_2^8 \setminus \{0\}$ in Table D.1.

Scanning the sets in this table we find that, irrespective of the chosen output mask v, Ω_v always contains two masks that can be expressed as a linear combination of the other three. To be more precise: for any $v \in \mathbb{F}_2^8 \setminus \{0\}$, there exists an ordering on the elements in Ω_v such that $\omega_1 \oplus \omega_2 = \omega_4$ and $\omega_1 \oplus \omega_3 = \omega_5$. Note that the sets in Table D.1 are listed such that each set adheres to this ordering. Since $a \oplus b = c \iff a \oplus c = b$, it follows that there are in fact eight such orderings. In the remaining analysis we will assume any one of these eight to be applied to Ω_v; all results hold irrespective of the chosen ordering.

3.2 Novel approximation methods

The combined results of Nyberg and Keliher et al. show that the absolute correlation of a linear approximation with `inv` is upper bounded by 2^{-3} and that this bound is tight. Table 3.1 additionally indicates that this upper bound is not just attained by a single linear approximation, but that five exist for every output mask $v \in \mathbb{F}_2^8 \setminus \{0\}$. In an attempt to exploit this property, this section introduces two approximation techniques that combine multiple linear approximations with a shared output mask. It is furthermore demonstrated that both methods allow for the construction of approximations of the `inv` function that achieve a correlation magnitude greater than 2^{-3}.

3.2.1 Weighted approximation

It is a well-known result in linear cryptanalysis literature that a Boolean function can be expressed as a weighted combination of linear Boolean functions [12, 17]; an expression that is closely related to Parseval's identity. We recall this result in Lemma 3.1 and provide a novel proof.

Lemma 3.1 (Equation 12, [12])**.** *Let $n, m \in \mathbb{N}_+$ and $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ any arbitrary vectorial Boolean function. For any $v \in \mathbb{F}_2^m$ and $a \in \mathbb{F}_2^n$ it holds that*

$$(-1)^{v^\top F(a)} = \sum_{u \in \mathbb{F}_2^n} C_{v,u}^F \cdot (-1)^{u^\top a}. \tag{3.1}$$

Proof. Let $v \in \mathbb{F}_2^m$ and $a \in \mathbb{F}_2^n$ be arbitrary. It holds that

$$\begin{aligned}
\sum_{u \in \mathbb{F}_2^n} C_{v,u}^F \cdot (-1)^{u^\top a} &\overset{(1)}{=} \sum_{u \in \mathbb{F}_2^n} \left(\frac{1}{|\mathbb{F}_2^n|} \sum_{x \in \mathbb{F}_2^n} (-1)^{u^\top x \oplus v^\top F(x)} \right) \cdot (-1)^{u^\top a} \\
&= \frac{1}{|\mathbb{F}_2^n|} \sum_{x \in \mathbb{F}_2^n} (-1)^{v^\top F(x)} \sum_{u \in \mathbb{F}_2^n} (-1)^{u^\top (x \oplus a)} \\
&\overset{(2)}{=} \frac{1}{|\mathbb{F}_2^n|} \sum_{x \in \mathbb{F}_2^n} (-1)^{v^\top F(x)} \cdot |\mathbb{F}_2^n| \cdot \delta(x \oplus a) \\
&= (-1)^{v^\top F(a)},
\end{aligned}$$

where (1) follows from Lemma 2.2 and (2) from Lemma 2.1. □

This lemma shows that when $v^\top F(x)$ is transformed to $(-1)^{v^\top F(x)}$, it can be decomposed into a weighted combination of the functions $(-1)^{u^\top x}$, where the weight of each is determined as the correlation between $u^\top x$ and $v^\top F(x)$. While it takes all parity functions with non-zero correlation to compute $(-1)^{v^\top F(x)}$ precisely, this equation suggests that a subset can be used to construct an approximation for $(-1)^{v^\top F(x)}$, which can be transformed back to an approximation for $v^\top F(x)$. The weighted approximation method attempts to achieve exactly this. Let us introduce this technique in an abstract form.

A *weighted approximation* $\langle U, W, v \rangle$ is a tuple consisting of a selection of input masks $U = (u_1, \ldots, u_r) \in (\mathbb{F}_2^n)^r$, their non-zero weights $W = (w_1, \ldots, w_r) \in \mathbb{R}^r$, and a mask $v : \mathbb{F}_2^m$, which can be used to approximate $v^\top F(x)$ with $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ a vectorial Boolean function. Let us use $f : \mathbb{F}_2^n \to \mathbb{F}_2$ to denote $v^\top F(x)$. We first construct an approximation for the function $(-1)^{f(x)}$ and then transform this into an approximation for $f(x)$. First, the functions $(-1)^{u_i^\top x}$ with weights w_i are combined in similar fashion to Equation 3.1, forming

$$Q_{U,W}(x) = \sum_{i=1}^{r} w_i \cdot (-1)^{u_i^\top x}. \tag{3.2}$$

Observe that the sign of this expression can act as an approximation for $(-1)^{f(x)}$. To transform this into an approximation for $f(x)$, we apply an indicator function to $Q_{U,W}$, forming

$$M_{U,W}(x) := \begin{cases} 0 & \text{if } Q_{U,W}(x) > 0 \\ 1 & \text{if } Q_{U,W}(x) < 0 \end{cases}. \tag{3.3}$$

Moving forward, we make use of the shorthand notation $C^F_{v,U,W} := C(M_{U,W}(x), v^\top F(x))$ to denote the correlation between the weighted approximation $M_{U,W}$ and Boolean function $v^\top F$.

When using this technique to construct an approximation for $f(x)$, one must thus decide which input masks to include in U and how to weigh them. We treat these topics in reverse order.

Choice of weights. In Equation 3.1 the weights w_i are chosen as the correlation between the parity functions $u_i^\top x$ and the function $v^\top F(x)$, for all $1 \leq i \leq r$. We provide a brief analysis underpinning the same choice of weight for the weighted approximation.

Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be an arbitrary vectorial Boolean function and $u \in \mathbb{F}_2^n$ an arbitrary input mask. Let us now view $(-1)^{u^\top x}$ as a 'predictor' for $(-1)^{f(x)}$. We first of all observe that the correlation between these two functions equals

$$2 \cdot \mathbb{P}\Big[(-1)^{u_i^\top x} = (-1)^{f(x)}\Big] - 1 = 2 \cdot \mathbb{P}\big[u_i^\top x = f(x)\big] - 1 = C\big(u^\top x, f(x)\big) \tag{3.4}$$

by construction. Let us define $C^f_u := C(u^\top x, f(x))$ and assume that f and u have been chosen such that this correlation is non-zero. Observe that the *sign* of this correlation indicates whether the predictor is either correct or incorrect for the majority of the input space. Multiplying $(-1)^{u^\top x}$ with $\mathtt{sgn}(C^f_u)$ therefore yields a predictor for $(-1)^{f(x)}$ that is predominantly correct. It thus holds that the correlation of each rectified parity function with $(-1)^{f(x)}$ is

$$C\Big(\mathtt{sgn}(C^f_u) \cdot (-1)^{u^\top x}, (-1)^{f(x)}\Big) = |C^f_u| > 0. \tag{3.5}$$

Let us call $\mathtt{sgn}(C^f_u) \cdot (-1)^{u^\top x}$ a *rectified* predictor. When constructing a predictor for $(-1)^{f(x)}$ using a linear combination of rectified predictors, it remains to establish the magnitude of their weights. Since the predictors are linearly combined, it is natural to assign a weight to each predictor that is proportional to its success rate. Since each rectified predictor is correct on at least half of the input space, we can choose the weight of $(-1)^{u^\top x}$ as $\frac{1}{2}|C^f_u| = |\mathbb{P}\big[u^\top x = f(x)\big] - \frac{1}{2}|$. Because the function $M_{U,W}$ is invariant under the multiplication of the weights W with any positive scalar, we may scale the weights with a factor of 2 without impacting $C^F_{v,U,W}$. As a result, a weight of $\mathtt{sgn}(C^f_u) \cdot |C^f_u| = C^f_u$ for each transformed parity function $(-1)^{u^\top x}$ is expected to yield an approximation for f that outperforms the parity functions individually.

Choice of masks. Having established that the correlation acts as a proper weight for each parity function, we now discuss choosing masks to include in this approximation. In this choice, a balance must be struck between the number of masks in U and the correlation of the approximation with the target function: a weighted approximation $\langle U, W, v\rangle$ with $U = \{u\}$ is equivalent to the linear approximation $\langle u, v\rangle$, while an approximation with $U = \mathbb{F}_2^n$ can compute f exactly, as illustrated by Equation 3.1. It is up to the designer to determine this balancing point based on the setting in which the approximation is deployed. We discuss constructing a small U while maximizing the correlation of the approximation.

As illustrated, the weighted approximation benefits from masks u with large correlation magnitude $|C^f_u|$, which will therefore be considered as members of U. It is encouraged to avoid constructing U for which $Q_{U,W}$ is close to zero. The reason for this is illustrated by the fact that $M_{U,W}$ is not defined for x with $Q_{U,W}(x) = 0$. The situation where $Q_{U,W}(x)$ is close to 0 can be viewed as the predictors $(-1)^{u^\top x}$ 'not reaching a consensus' about the value of $(-1)^{f(x)}$. Especially when $Q_{U,W}$ is close to zero for a large part of the input space is it expected that $M_{U,W}$ will often be incorrect, thus forming a poorly performing approximation for $f(x)$. Observe that this situation can be avoided by constructing U from an odd number of masks that have identical absolute weight. This choice of U thwarts an additional problem: when the difference in weight between masks in U is large, it is possible that some masks do not even influence the sign of $Q_{U,W}$, leading to no change of the behaviour of $M_{U,W}$ and thus no improvement in the approximation's correlation. This effect is illustrated by the extreme case that the weight of

one mask $u' \in U$ is greater than the sum of the weights of the others. In this situation, it follows that

$$\mathtt{sgn}(Q_{U,W}(x)) = \mathtt{sgn}\left(\sum_{u \in U} C_u^f \cdot (-1)^{u^\top x}\right) = \mathtt{sgn}\left(C_{u'}^f \cdot (-1)^{u'^\top x}\right) = \mathtt{sgn}\left(C_{u'}^f\right) \cdot (-1)^{u'^\top x} \tag{3.6}$$

for all x, which implies that either $M_{U,W}(x) \equiv u'^\top x$ or $M_{U,W}(x) \equiv u'^\top x \oplus 1$. In either case this has as a result that $|C^F_{v,U,W}| = |C^F_{v,u'}|$, implying that the weighted approximation method yields no improvement compared to the linear approximation $\langle u', v\rangle$.

We lastly discuss the inclusion of linearly dependent masks in U. Observe that when $a, b \in U$ with non-zero weights, the values of $(-1)^{a^\top x}$ and $(-1)^{b^\top x}$ must be gathered to compute $Q_{U,W}$. By multiplying these values as $(-1)^{a^\top x} \cdot (-1)^{b^\top x} = (-1)^{(a \oplus b)^\top x}$, it follows that $a \oplus b$ can be computed; the value of $(a \oplus b)^\top x$ does not have to be gathered. If the magnitude of $C^f_{a \oplus b}$ does not significantly deviate from that of C^f_a and C^f_b, it is thus encouraged to consider including $a \oplus b$ in U.

Application to `inv`. To illustrate the utility of the weighted approximation method, we use it to construct three approximations for `inv`. Each approximation is created for the same arbitrary output mask $v \in \mathbb{F}_2^8 \setminus \{0\}$, yet uses a slightly different set of input masks. We base the three input mask sets on Ω_v, since the masks ω_i in this set achieve the same, high correlation of $C^{\mathtt{inv}}_{v,\omega_i} = 2^{-3}$ with the output mask v for each $1 \leq i \leq 5$. Let us moreover recall that the set of input masks is best constructed with an odd number of members that is greater than 1: using a single mask yields an approximation that is equivalent to a linear approximation. As such, we will consider sets containing three, five and seven masks.

Recall that the masks $\omega_4, \omega_5 \in \Omega_v$ can be expressed as linear combinations of ω_1, ω_2 and ω_3. To observe the value in considering these linearly dependent masks, we use both $U_1 := \{\omega_1, \omega_2, \omega_3\}$ and $U_2 = \Omega_v$ to construct a weighted approximation. To further investigate the influence of the inclusion of linearly dependent masks in U, we additionally consider the set $U_3 := \mathtt{span}(\Omega_v) \setminus \{0\}$, where `span` denotes the linear span of a set of vectors. This set includes the remaining two non-zero linear combinations of the masks in U_1. Let us name these two masks $\omega_6 := \omega_2 \oplus \omega_3$ and $\omega_7 := \omega_1 \oplus \omega_2 \oplus \omega_3$. The correlations of these two masks with $v^\top \mathtt{inv}$ are $C^{\mathtt{inv}}_{v,\omega_6} = C^{\mathtt{inv}}_{v,\omega_7} = -2^{-4}$.

We thus construct the weighted approximations $\langle U_1, W_1, v\rangle, \langle U_2, W_2, v\rangle$, and $\langle U_3, W_3, v\rangle$, where the weight $w_i \in W$ associated with each mask $u_i \in U$ is chosen as $C^{\mathtt{inv}}_{v,u_i}$. The correlations of M_{U_1,W_1}, M_{U_2,W_2} and M_{U_3,W_3} with $v^\top \mathtt{inv}$ can then be computed using Lemma 2.2. It holds that $C^{\mathtt{inv}}_{v,U_1,W_1} = \frac{28}{128} \approx 2^{-2.2}$, $C^{\mathtt{inv}}_{v,U_2,W_2} = 2^{-2}$ and $C^{\mathtt{inv}}_{v,U_3,W_3} = 2^{-2}$. We firstly remark that the correlation of $\langle U_1, W_1, v\rangle$ with `inv` improves over the 2^{-3} upper bound shown by Nyberg for linear approximations. The results moreover show that including ω_4 and ω_5 leads to a further improvement of the correlation, demonstrating the benefit of introducing linearly dependent masks to the input mask set. It is therefore surprising that the correlation magnitude of $\langle U_3, W_3, v\rangle$ does not improve compared to $\langle U_2, W_2, v\rangle$. Further investigation here reveals that $\mathtt{sgn}(Q_{U_2,W_2}) \equiv \mathtt{sgn}(Q_{U_3,W_3})$, i.e. the weights of ω_6 and ω_7 are too small to impact the sign.

With this example we have thus demonstrated that the weighted approximation technique can be utilized to construct an approximation for `inv` that achieves a correlation of 2^{-2} — outperforming all individual linear approximations — by combining five linear approximations, of which three are linearly independent.

3.2.2 Conditional approximation

We continue with a second approximation technique that adapts Conditional Linear Cryptanalysis — a technique introduced in [6] for the Feistel cipher DES [42] — to SPNs. Whereas the weighted approximation aggregates multiple linear approximations to create an approximation, the conditional approximation technique instead partitions the plaintext space using a set of linear approximations in an attempt to increase the correlation of another linear approximation on some of the parts. Let us introduce the approximation technique.

A *conditional approximation* $\langle u, v\rangle|_U$ for a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ comprises a linear approximation $\langle u, v\rangle$ with $u \in \mathbb{F}_2^n$ and $v \in \mathbb{F}_2^m$, and a set of conditioning input masks $U = \{u_1, \ldots, u_r\} \subset \mathbb{F}_2^n$. In this approximation, the set U is leveraged to create a partition $\mathcal{R}_U$ on the plaintext space $\mathbb{F}_2^n$. This partition consists of *plaintext classes*, where each class $\mathcal{R}_U^b$ with $b \in \mathbb{F}_2^r$ contains those plaintexts $x \in \mathbb{F}_2^n$ for which $(u_1^\top x, \ldots, u_r^\top x) = b$. For each plaintext class we are interested in

$$p_b = \mathbb{P}\big[u^\top x = v^\top F(x) \;\big|\; x \in \mathcal{R}_U^b\big]; \tag{3.7}$$

the probability that the input bit masked by u and output bit masked by v are the same for a uniformly chosen plaintext in the class $\mathcal{R}_U^b$. Note that the definition of correlation naturally extends to the definition of *conditional correlation* as

$$C_{v,u}^F\Big|_{x\in\mathcal{R}_U^b} := 2\cdot\mathbb{P}\big[u^\top x = v^\top F(x) \mid x\in\mathcal{R}_U^b\big] - 1; \tag{3.8}$$

the correlation of the linear approximation $\langle u, v\rangle$ with F conditioned on $x\in\mathcal{R}_U^b$. Observe that Lemma 2.2 can be adapted to conditional approximations as well, from which it then follows that

$$C_{v,u}^F\Big|_{x\in\mathcal{R}_U^b} = \frac{1}{|\mathcal{R}_U^b|}\sum_{x\in\mathcal{R}_U^b}(-1)^{u^\top x\oplus v^\top F(x)}. \tag{3.9}$$

Depending on the choice of masks in U, it may be possible to achieve a conditional correlation in one or more classes that exceeds the correlation of the unconditioned linear approximation, i.e. for a non-uniform choice of inputs.

Relation to linear approximations. To provide an understanding for how a conditional approximation fits in with standard linear cryptanalysis, we illustrate a relation between both approximation techniques. To this end, we first introduce Lemma 3.2.

Lemma 3.2. *Let $F:\mathbb{F}_2^n\to\mathbb{F}_2^n$ be an arbitrary Boolean permutation and $v\in\mathbb{F}_2^n\setminus\{0\}$ arbitrary output mask. For any input mask $u\in\mathbb{F}_2^n\setminus\{0\}$ it holds that*

$$C_{v,0}^F\Big|_{x\in\mathcal{R}_{\{u\}}^0} = -C_{v,0}^F\Big|_{x\in\mathcal{R}_{\{u\}}^1} = C_{v,u}^F, \tag{3.10}$$

where $\mathcal{R}_{\{u\}}^b := \{x\in\mathbb{F}_2^n \mid u^\top x = b\}$ for all $b\in\mathbb{F}_2$.

Proof. Note that $|\mathcal{R}_{\{u\}}^0| = |\mathcal{R}_{\{u\}}^1| = |\mathbb{F}_2^n|/2$ since $u^\top x$ is a balanced function. It can thus be shown that

$$\begin{aligned}
C_{v,0}^F\Big|_{x\in\mathcal{R}_{\{u\}}^0} + C_{v,0}^F\Big|_{x\in\mathcal{R}_{\{u\}}^1} &= \left(\frac{1}{|\mathcal{R}_{\{u\}}^0|}\sum_{x\in\mathcal{R}_{\{u\}}^0}(-1)^{0^\top x\oplus v^\top F(x)}\right) + \left(\frac{1}{|\mathcal{R}_{\{u\}}^1|}\sum_{x\in\mathcal{R}_{\{u\}}^1}(-1)^{0^\top x\oplus v^\top F(x)}\right)\\
&= \frac{1}{|\mathcal{R}_{\{u\}}^0|}\sum_{x\in\mathbb{F}_2^n}(-1)^{v^\top F(x)}\\
&\overset{(1)}{=} \frac{1}{|\mathcal{R}_{\{u\}}^0|}\sum_{x\in\mathbb{F}_2^n}(-1)^{v^\top x}\\
&\overset{(2)}{=} 0,
\end{aligned}$$

where step (1) holds because F is a permutation, while step (2) follows from Lemma 2.1. Subsequently,

$$\begin{aligned}
2\cdot C_{v,0}^F\Big|_{x\in\mathcal{R}_{\{u\}}^0} &= C_{v,0}^F\Big|_{x\in\mathcal{R}_{\{u\}}^0} - C_{v,0}^F\Big|_{x\in\mathcal{R}_{\{u\}}^1}\\
&= \left(\frac{1}{|\mathcal{R}_{\{u\}}^0|}\sum_{x\in\mathcal{R}_{\{u\}}^0}(-1)^{0^\top x\oplus v^\top F(x)}\right) - \left(\frac{1}{|\mathcal{R}_{\{u\}}^1|}\sum_{x\in\mathcal{R}_{\{u\}}^1}(-1)^{0^\top x\oplus v^\top F(x)}\right)\\
&\overset{(3)}{=} \left(\frac{1}{|\mathcal{R}_{\{u\}}^0|}\sum_{x\in\mathcal{R}_{\{u\}}^0}(-1)^{u^\top x\oplus v^\top F(x)}\right) - \left(-\frac{1}{|\mathcal{R}_{\{u\}}^1|}\sum_{x\in\mathcal{R}_{\{u\}}^1}(-1)^{u^\top x\oplus v^\top F(x)}\right)\\
&= \frac{2}{|\mathbb{F}_2^n|}\sum_{x\in\mathbb{F}_2^n}(-1)^{u^\top x\oplus v^\top F(x)}\\
&= 2\cdot C_{v,u}^F,
\end{aligned}$$

where step (3) multiplies the summations with $(-1)^{u^\top x}$, which equals 1 for all $x\in\mathcal{R}_{\{u\}}^0$ and -1 for $x\in\mathcal{R}_{\{u\}}^1$. The lemma follows from these two equations. □

It follows from the lemma that a conditional approximation $\langle 0, v\rangle|_{\{u\}}$ restricted to the plaintext partition $\mathcal{R}^0_{\{u\}}$ with $u, v \in \mathbb{F}_2^n \setminus \{0\}$ is equivalent to the linear approximation $\langle u, v\rangle$. Note moreover that the correlation magnitude of this approximation restricted to $\mathcal{R}^1_{\{u\}}$ is identical to $|C^F_{v,u}|$ as well.

We can illustrate further relations between a conditional approximation and linear approximation by observing a relation between conditional approximations. Equation 3.9 can be used to show that the conditional correlation magnitude remains constant under the addition of elements in U to input mask u. We present this in Lemma 3.3.

Lemma 3.3. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be an arbitrary vectorial Boolean function, $u \in \mathbb{F}_2^n \setminus \{0\}$ and $v \in \mathbb{F}_2^m \setminus \{0\}$ arbitrary masks, and $U \subset \mathbb{F}_2^n$ with $|U| = r$ an arbitrary set of vectors. It then holds that*

$$(-1)^{b_i} \cdot C^F_{v,u\oplus u_i}\Big|_{x\in\mathcal{R}^b_U} = C^F_{v,u}\Big|_{x\in\mathcal{R}^b_U}$$

for any conditional mask $u_i \in U$ and any class $b = (b_1, \ldots, b_r) \in \mathbb{F}_2^r$.

Proof. Let $1 \le i \le r$ arbitrary and recall that for all $x \in \mathcal{R}^b_U$, $u_i^\top x = b_i$ for all $u_i \in U$. We can use this to show that

$$\begin{aligned}
C^F_{v,u}\Big|_{x\in\mathcal{R}^b_U} &= \frac{1}{|\mathcal{R}^b_U|}\sum_{x\in\mathcal{R}^b_U}(-1)^{u^\top x\oplus v^\top F(x)} \\
&= \frac{1}{|\mathcal{R}^b_U|}\sum_{x\in\mathcal{R}^b_U}(-1)^{u^\top x\oplus v^\top F(x)}\cdot(-1)^{u_i^\top x}\cdot(-1)^{u_i^\top x} \\
&\overset{(1)}{=} (-1)^{b_i}\cdot\frac{1}{|\mathcal{R}^b_U|}\sum_{x\in\mathcal{R}^b_U}(-1)^{(u\oplus u_i)^\top x\oplus v^\top F(x)} \\
&= (-1)^{b_i}\cdot C^F_{v,u\oplus u_i}\Big|_{x\in\mathcal{R}^b_U}.
\end{aligned}$$

Here, step (1) follows from the observation that $u_i^\top x = b_i$ for all $x \in \mathcal{R}^b_U$ and that $a^\top x \oplus b^\top x = (a \oplus b)^\top x$. □

Applied to our previous example, this yields that the conditional approximations $\langle 0, v\rangle|_{\{u\}}$ and $\langle u, v\rangle|_{\{u\}}$ achieve the same correlation magnitude on each plaintext class, and thus $\langle u, v\rangle|_{\{u\}}$ restricted to $\mathcal{R}^0_{\{u\}}$ is equivalent to $\langle u, v\rangle$ as well. Note that Lemma 3.3 can alternatively be viewed as a technique to invert the sign of the conditional correlation associated with half of the plaintext classes. This will prove useful in constructing a conditional approximation for `inv`.

Constructing U. In [6], several techniques are presented to for creating conditional approximations with great performance, including heuristic approaches involving known (non-conditional) linear approximations and an extension to a search algorithm presented by Matsui [36]. We add to this that one should only consider conditioning masks for U that are linearly independent. When U contains an element that can be expressed as a linear combination of the others, the partition $\mathcal{R}_U$ will contain empty plaintext classes. This can be illustrated with the example $U := \{a, b, a \oplus b\}$. Observe that $(a^\top x, b^\top x, (a \oplus b)^\top x)$ cannot equal $(0, 0, 1)$, $(0, 1, 0)$, $(1, 0, 0)$ or $(1, 1, 1)$ for any x due to the linear dependence of the elements in U. This has as a result that the plaintext classes associated with these vectors will be empty. When U only contains linearly independent vectors, the number of plaintext classes is $2^{|U|}$. It moreover follows that all classes are of equal size $2^{n-|U|}$, since the functions $u_i^\top x$ are balanced for all $u_i \in U$.

Application to `inv`. We construct a conditional approximation for the inversion function `inv`. Intuitively, the highest conditional correlation is achieved when a linear approximation $\langle u, v\rangle$ is conditioned on input masks u_i that yield a high correlation with its output mask v. We have moreover seen that the conditioning set U should be constructed using linearly independent input masks to avoid empty plaintext classes. To this end, we construct the conditional approximation $\langle 0, v\rangle|_U$, with $U = \{\omega_1, \omega_2, \omega_3\} \subset \Omega_v$. Since $|U| = 3$, this gives rise to a plaintext space partition $\mathcal{R}_U$ comprising $2^{|U|} = 8$ plaintext classes, each containing $2^{8-|U|} = 2^5 = 32$ plaintexts. With this partition in place, we are interested in the correlations $C^{\texttt{inv}}_{v,0}|_{x\in\mathcal{R}^b_U}$ for each $b \in \mathbb{F}_2^3$, which can be computed using Equation 3.9. The values are displayed in Table 3.2.

From this table we can gather that the conditional approximation $\langle 0, v\rangle|_U$ achieves a conditional correlation of $\pm 2^{-1}$ for some plaintext classes. This is a substantial increase when compared to the 2^{-3} correlation upper bound

Table 3.2: $C^{\texttt{inv}}_{v,0}\Big|_{x\in\mathcal{R}^b_U}$ for each $b = (\omega_1^\top x, \omega_2^\top x, \omega_3^\top x) \in \mathbb{F}_2^3$.

$\omega_1^\top x$	0	0	0	0	1	1	1	1
$\omega_2^\top x$	0	0	1	1	0	0	1	1
$\omega_3^\top x$	0	1	0	1	0	1	0	1
$C^{\texttt{inv}}_{v,0}\Big\|_{x\in\mathcal{R}^b_U}$	2^{-1}	2^{-2}	2^{-2}	-2^{-1}	-2^{-3}	-2^{-3}	-2^{-3}	-2^{-3}

for linear approximations applied to `inv`, illustrating the utility of the conditional approximation. Upon closer inspection, we observe that there are exactly two plaintext classes with correlation magnitude 2^{-1}: one with a positive sign and one with a negative. It is therefore concluded that the partition induced by U is finer than necessary; we may construct a coarser partition by combining parts in the existing partition such that there is only one class with conditional correlation $\pm 2^{-1}$. We illustrate how this can be achieved.

When merging two plaintext classes, it follows that the conditional correlation of the merged class is the average of the original classes. When attempting to maintain a high correlation in the merged class, it is thus essential that the correlation signs of the classes being merged are the same. Since this is not the case in our situation, it is necessary to augment the correlation sign of one of the two classes yielding a correlation magnitude of 2^{-1}. Observe that this can be accomplished by multiplying the correlation of all classes for which $\omega_3^\top x = 1$ with -1. According to Lemma 3.3, this can be achieved by considering the conditional approximation $\langle \omega_3, v\rangle|_U$ instead of $\langle 0, v\rangle|_U$. An overview of the correlations associated with $\langle \omega_3, v\rangle|_U$ when applied to `inv` is shown in Table 3.3.

Table 3.3: $C^{\texttt{inv}}_{v,\omega_3}\Big|_{x\in\mathcal{R}^b_U}$ for each $b = (\omega_1^\top x, \omega_2^\top x, \omega_3^\top x) \in \mathbb{F}_2^3$.

$\omega_1^\top x$	0	0	0	0	1	1	1	1
$\omega_2^\top x$	0	0	1	1	0	0	1	1
$\omega_3^\top x$	0	1	0	1	0	1	0	1
$C^{\texttt{inv}}_{v,\omega_3}\Big\|_{x\in\mathcal{R}^b_U}$	2^{-1}	-2^{-2}	2^{-2}	2^{-1}	-2^{-3}	2^{-3}	-2^{-3}	2^{-3}

With the signs of both classes now the same, it is possible to modify the set of conditioning masks such that these two classes are merged and the conditional correlation is maintained. To merge the classes, one needs to discover the linear properties that the classes have in common and modify the set U accordingly. In our case, observe that the value of $\omega_1^\top x$ is the same for the two classes of interest and that the value of $\omega_2^\top x \oplus \omega_3^\top x = (\omega_2 \oplus \omega_3)^\top x = \omega_6^\top x$ is moreover constant. This means that these two classes are merged when we instead partition on $U' := \{\omega_1, \omega_6\} \subset \texttt{span}(\Omega_v)$. The conditional correlations associated with the approximation $\langle \omega_3, v\rangle|_{U'}$ are presented in Table 3.4.

Table 3.4: $C^{\texttt{inv}}_{v,\omega_3}\Big|_{x\in\mathcal{R}^b_{U'}}$ for each $b = (\omega_1^\top x, \omega_6^\top x) \in \mathbb{F}_2^2$.

$\omega_1^\top x$	0	0	1	1
$\omega_6^\top x$	0	1	0	1
$C^{\texttt{inv}}_{v,\omega_3}\Big\|_{x\in\mathcal{R}^b_{U'}}$	2^{-1}	0	0	0

Here, the linear approximation $\langle \omega_3, v\rangle$ achieves a correlation of 2^{-1} in exactly one plaintext class. Leveraging the equivalence relation listed in Lemma 3.3, it furthermore follows that the conditional approximations $\langle \omega_2, v\rangle|_{U'}$, $\langle \omega_4, v\rangle|_{U'}$, and $\langle \omega_5, v\rangle|_{U'}$ also yield conditional correlations of magnitude 2^{-1} for this plaintext class.

We have thus shown that for `inv` the conditional approximation technique can be used to partition the plaintext space in four parts such that for one part we can observe a conditional correlation of 2^{-1} for the linear approximation $\langle \omega_3, v\rangle$, which is significantly higher than its unconditioned correlation.

3.3 Composition with affine transformations

We have thus demonstrated that weighted and conditional approximations for `inv` can be constructed that achieve a higher correlation than any linear approximation. In cryptography, `inv` is rarely used in isolation, however. Recall for example that the AES s-box does not just comprise the `inv` function, but is additionally composed of the linear transformation `L` and the affine translation `T`. Moreover, each application of `SubBytes` is preceded by the addition of a round key — an affine translation as well. To determine the utility of the new approximation techniques in linear cryptanalysis, it should be investigated how both types of approximations for `inv` can be adapted to form approximations for `inv` composed with an affine transformation such that the correlation magnitude is maintained.

To this end, we present a general discussion on extending an approximation for an arbitrary vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ to one for F composed with an affine transformation. For each of the four possible compositions, we derive this extension in two steps: we first express the correlation of an approximation for the composition in terms of the correlation of an approximation for F itself, after which this expression is used to derive the rules for extending an approximation for F to one for the composition. Throughout, we use L to denote an arbitrary linear vectorial Boolean function, while X_k denotes the affine translation by some arbitrary constant k.

3.3.1 Weighted Approximation

We commence with constructing a weighted approximation for the compositions $X_k \circ F$, $F \circ X_k$, $L \circ F$ and $F \circ L$. Let us start with the compositions $X_k \circ F$. To this end, we introduce Lemma 3.4.

Lemma 3.4. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be an arbitrary vectorial Boolean function and $X_k : \mathbb{F}_2^m \to \mathbb{F}_2^m$ an affine translation by $k \in \mathbb{F}_2^m$. Moreover, let $U \subseteq \mathbb{F}_2^n$ be arbitrary, with $r := |U|$. For any $v \in \mathbb{F}_2^m$ and $W \in \mathbb{R}^r$ it then holds that*

$$(-1)^{v^\top k} \cdot C^{X_k \circ F}_{v,U,W} = C^F_{v,U,W}. \tag{3.11}$$

Proof. Observe that

$$\begin{aligned}
C^{X_k \circ F}_{v,U,W} &= \frac{1}{|\mathbb{F}_2^n|} \sum_{x \in \mathbb{F}_2^n} (-1)^{M_{U,W}(x) \oplus v^\top (X_k \circ F)(x)} \\
&= \frac{1}{|\mathbb{F}_2^n|} \sum_{x \in \mathbb{F}_2^n} (-1)^{M_{U,W}(x) \oplus v^\top (k \oplus F(x))} \\
&= \frac{1}{|\mathbb{F}_2^n|} \sum_{x \in \mathbb{F}_2^n} (-1)^{M_{U,W}(x) \oplus v^\top F(x)} \cdot (-1)^{v^\top k} \\
&= (-1)^{v^\top k} \cdot \frac{1}{|\mathbb{F}_2^n|} \sum_{x \in \mathbb{F}_2^n} (-1)^{M_{U,W}(x) \oplus v^\top F(x)} \\
&= (-1)^{v^\top k} \cdot C^F_{v,U,W}.
\end{aligned}$$

It then follows that $(-1)^{v^\top k} \cdot C^{X_k \circ F}_{v,U,W} = C^F_{v,U,W}$. □

Lemma 3.4 illustrates that all weighted approximations $\langle U, W, v \rangle$ yield a correlation of the same magnitude when used to approximate both F and $X_k \circ F$; only the sign could be inverted depending on the value of k. This illustrates that it suffices to construct a well-performing weighted approximation for F as this can also be used to approximate $X_k \circ F$. Note that this argument remains valid when the value of k is unknown, as could be the case when it is part of a round key. We transition to discussing the composition $L \circ F$. To this end, let us introduce Lemma 3.5.

Lemma 3.5. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be an arbitrary vectorial Boolean functions and $L : \mathbb{F}_2^m \to \mathbb{F}_2^m$ an arbitrary linear permutation. Moreover, let $U \subseteq \mathbb{F}_2^n$ be arbitrary, with $r := |U|$. For any $v \in \mathbb{F}_2^m$ and $W \in \mathbb{R}^r$ it then holds that*

$$C^{L \circ F}_{v',U,W} = C^F_{v,U,W}, \tag{3.12}$$

where $v' := M^\top_{L^{-1}} v$.

Proof. Let us first observe that the matrix $M_{L^{-1}}$ exists since L is a linear permutation. Observe furthermore that $M_{L^{-1}} = M_L^{-1}$ by construction. It now holds that

$$\begin{aligned}(M_{L^{-1}}^\top v)^\top (L \circ F)(x) &= (M_{L^{-1}}^\top v)^\top (M_L F(x)) \\ &= (M_L^\top M_{L^{-1}}^\top v)^\top F(x) \\ &= v^\top F(x)\end{aligned}$$

for all $x \in \mathbb{F}_2^n$, which implies that $(M_{L^{-1}}^\top v)^\top (L \circ F) \equiv v^\top F$. It is then concluded that

$$C_{v',U,W}^{L\circ F} = C\big(M_{U,W}, v'^\top (L \circ F)\big) = C\big(M_{U,W}, v^\top F\big) = C_{v,U,W}^F$$

□

Lemma 3.5 shows that extending an approximations for F to one for $L \circ F$ involves modifying the output mask for which the correlation is achieved. In particular, an approximation $\langle U, W, v\rangle$ for F is easily modified to an approximation for $L \circ F$ by replacing the output mask v with $M_{L^{-1}}^\top v$.

Readers well-known with the field of Linear Cryptanalysis will observe that the two previous extension rules are equivalent to those for linear approximations. We will now discuss *prepending* F with both X_k and L and discover that the rules that arise here will deviate from this norm. We first treat the composition $F \circ X_k$ in Lemma 3.6.

Lemma 3.6. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be an arbitrary vectorial Boolean function and let $X_k : \mathbb{F}_2^n \to \mathbb{F}_2^n : x \mapsto x \oplus k$ denote the addition of an arbitrary $k \in \mathbb{F}_2^n$. Moreover, let $U \subseteq \mathbb{F}_2^n$ be arbitrary, with $r := |U|$. For any $v \in \mathbb{F}_2^m$ and $W \in \mathbb{R}^r$ it holds that*

$$C_{v,U,W'}^{F\circ X_k} = C_{v,U,W}^F \tag{3.13}$$

where $W' := \Big(w_1 \cdot (-1)^{u_1^\top k}, \ldots, w_r \cdot (-1)^{u_r^\top k}\Big)$, with $W = (w_1, \ldots, w_r)$.

Proof. Let $w_i' := w_i \cdot (-1)^{u_i^\top k}$ for all $1 \le i \le r$. We first observe that

$$\begin{aligned}(Q_{U,W'} \circ X_k)(x) &= Q_{U,W'}(x \oplus k) \\ &= \sum_{i=1}^r w_i' \cdot (-1)^{u_i^\top (x\oplus k)} \\ &= \sum_{i=1}^r w_i' \cdot (-1)^{u_i^\top k} \cdot (-1)^{u^\top x} \\ &= \sum_{i=1}^r w_i \cdot (-1)^{u_i^\top k} \cdot (-1)^{u_i^\top k} \cdot (-1)^{u^\top x} \\ &= \sum_{i=1}^r w_i \cdot (-1)^{u^\top x} \\ &= Q_{U,W}(x)\end{aligned}$$

for any $x, k \in \mathbb{F}_2^n$. It follows from this that $Q_{U,W'} \circ X_k \equiv Q_{U,W}$ and thus $M_{U,W'} \circ X_k \equiv M_{U,W}$. As a result, it holds that

$$\begin{aligned}C_{v,U,W'}^{F\circ X_k} &= C(M_{U,W'}, v^\top (F \circ X_k)) \\ &\overset{(1)}{=} C(M_{U,W'} \circ X_k, v^\top (F \circ X_k \circ X_k)) \\ &\overset{(2)}{=} C(M_{U,W}, v^\top F) \\ &= C_{v,U,W}^F.\end{aligned}$$

Here, step (1) follows from the fact that X_k is a permutation; prepending this to both functions leads to no change in correlation; step (2) uses the fact that $(X_k \circ X_k)(x) = x \oplus k \oplus k = x$. □

This result illustrates that a weighted approximation $\langle U, W, v\rangle$ for F can be adapted for $F \circ X_k$ by updating the weights of the predictors. Note that this adjustment can only be made when k is known. When this value is unknown instead — for example because it is part of an unknown round key — it is unclear how the weights for the input masks should be adapted for the approximation to maintain its correlation. This is left for future research.

We finally discuss adapting a weighted approximation for F to one for $F \circ L$. In Lemma 3.7 it is shown that an approximation $\langle U, W, v\rangle$ for F can be adapted for $F \circ L$ by appropriately modifying the set of input masks. Note here that the weights of the original masks should still be used.

Lemma 3.7. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be an arbitrary vectorial Boolean function and let $L : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an arbitrary linear permutation. Moreover, let $U \subseteq \mathbb{F}_2^n$ be arbitrary, with $r := |U|$. For any $v \in \mathbb{F}_2^m$, and $W \in \mathbb{R}^r$ it holds that*

$$C^{F\circ L}_{v,U',W} = C^F_{v,U,W}, \tag{3.14}$$

where $U' := \left\{M_L^\top u_i \;\middle|\; u_i \in U\right\}$.

Proof. The proof to this lemma is similar to that of Lemma 3.4, but hinges instead on the observation that $M_{U',W} \circ L^{-1} \equiv M_{U,W}$. The full proof can be found in Appendix A.

We can thus conclude that the weighted approximation can be expanded to include affine transformations that are either prepended or appended to F, while maintaining the same correlation magnitude. Only when an affine translation with an unknown translation vector is prepended to F, do we not yet know how the weights of the approximation should be adapted to be certain that the correlation is maintained.

3.3.2 Conditional Approximation

We next investigate the possibility of extending a conditional approximation for F to one for F composed with an affine transformation. We first discuss the general composition $G \circ F$ with $G : \mathbb{F}_2^m \to \mathbb{F}_2^r$ an arbitrary vectorial Boolean function. Lemma 3.8 expresses the correlation of a conditional approximation on $G \circ F$ in terms of the correlations of approximations for G and F.

Lemma 3.8. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ and $G : \mathbb{F}_2^m \to \mathbb{F}_2^r$ be arbitrary vectorial Boolean functions. For any $u \in \mathbb{F}_2^n$, $v \in \mathbb{F}_2^r$ and $U \subseteq \mathbb{F}_2^n$ it holds that*

$$C^{G\circ F}_{v,u}\Big|_{x\in\mathcal{R}_U^b} = \sum_{z\in\mathbb{F}_2^m} C^G_{v,z} \cdot C^F_{z,u}\Big|_{x\in\mathcal{R}_U^b} \tag{3.15}$$

Proof. Let us recall Equation 3.9, which states that

$$C^F_{v,u}\Big|_{x\in\mathcal{R}_U^b} = \frac{1}{|\mathcal{R}_U^b|} \sum_{x\in\mathcal{R}_U^b} (-1)^{u^\top x \oplus v^\top F(x)}.$$

It then holds that

$$\begin{aligned}
\sum_{z\in\mathbb{F}_2^m} C^G_{v,z} \cdot C^F_{z,u}\Big|_{x\in\mathcal{R}_U^b} &= \sum_{z\in\mathbb{F}_2^m} \left(\frac{1}{|\mathbb{F}_2^m|} \sum_{x\in\mathbb{F}_2^m} (-1)^{z^\top x \oplus v^\top G(x)}\right) \cdot \left(\frac{1}{|\mathcal{R}_U^b|} \sum_{y\in\mathcal{R}_U^b} (-1)^{u^\top y \oplus z^\top F(y)}\right) \\
&= \frac{1}{|\mathbb{F}_2^m|}\frac{1}{|\mathcal{R}_U^b|} \sum_{x\in\mathbb{F}_2^m} \sum_{y\in\mathcal{R}_U^b} (-1)^{u^\top y \oplus v^\top G(x)} \sum_{z\in\mathbb{F}_2^m} (-1)^{z^\top (x \oplus F(y))} \\
&\overset{(1)}{=} \frac{1}{|\mathbb{F}_2^m|}\frac{1}{|\mathcal{R}_U^b|} \sum_{x\in\mathbb{F}_2^m} \sum_{y\in\mathcal{R}_U^b} (-1)^{u^\top y \oplus v^\top G(x)} \cdot |\mathbb{F}_2^m| \cdot \delta(x \oplus F(y)) \\
&= \frac{1}{|\mathcal{R}_U^b|} \sum_{y\in\mathcal{R}_U^b} (-1)^{u^\top y \oplus v^\top G(F(y))} \\
&= C^{G\circ F}_{v,u}\Big|_{x\in\mathcal{R}_U^b},
\end{aligned}$$

where step (1) follows from Lemma 2.1. □

Before applying this lemma to X_k and L, let us first recall two lemmas. We have previously seen in Lemma 2.3 that $C^{X_k}_{v,u} = \delta(v \oplus u)$, which is to say that the correlation of linear approximation $\langle u, v\rangle$ with the X_k function is non-zero if and only if $u = v$. In [16] a similar equality is provided for linear vectorial Boolean functions. We recall this property in Lemma 3.9.

Lemma 3.9 (Equation 7.36, [16])**.** *Let $L : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be a linear vectorial Boolean function. For any input mask $u \in \mathbb{F}_2^n$ and output mask $v \in \mathbb{F}_2^m$ it holds that*

$$C^L_{v,u} = \delta(M_L^\top v \oplus u).$$

It follows from this lemma that only the linear approximations $\langle M_L^\top v, v\rangle$ yield a non-zero correlation with the linear transformation L. Observe that these approximations can alternatively be expressed as $\langle u, M_{L^{-1}}^\top u\rangle$ when $n = m$.

These lemmas can be combined with Lemma 3.8 to express the correlation of conditional approximations for respectively $X_k \circ F$ and $L \circ F$ in terms of their correlation with F. For the composition $X_k \circ F$ this yields

$$(-1)^{v^\top k} \cdot C^{X_k \circ F}_{v,u}\Big|_{x \in \mathcal{R}^b_U} = C^F_{v,u}\Big|_{x \in \mathcal{R}^b_U}, \tag{3.16}$$

for any $k \in \mathbb{F}_2^n$. This shows that a conditional approximation for F can also be used to approximate $X_k \circ F$; only the sign of the correlation could change. Here, the argument again holds regardless of whether k is known or not. When Lemma 3.8 is combined with Lemma 3.9, we find that

$$C^{L\circ F}_{M_{L^{-1}}^\top v,u}\Big|_{x \in \mathcal{R}^b_U} = C^F_{v,u}\Big|_{x \in \mathcal{R}^b_U}. \tag{3.17}$$

This result demonstrates that a conditional approximation $\langle u, v\rangle|_U$ for F can be adapted for $L \circ F$ by replacing the output mask v with $M_{L^{-1}}^\top v$.

We next consider the compositions $F \circ X_k$ and $F \circ L$. Since the obtained relations are significantly different in either situation, these cases are treated separately. Lemma 3.10 expresses the correlation of a conditional approximation for $F \circ X_k$ in terms of the conditional correlation of F.

Lemma 3.10. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be an arbitrary vectorial Boolean function and let $X_k : \mathbb{F}_2^n \to \mathbb{F}_2^n : x \mapsto x \oplus k$ denote the addition of an arbitrary $k \in \mathbb{F}_2^n$. For arbitrary masks $u \in \mathbb{F}_2^n$ and $v \in \mathbb{F}_2^m$, and set of linearly independent conditioning masks $U \subseteq \mathbb{F}_2^n$ it holds that*

$$(-1)^{u^\top k} \cdot C^{F\circ X_k}_{v,u}\Big|_{x \in \mathcal{R}^{b'}_U} = C^F_{v,u}\Big|_{x \in \mathcal{R}^b_U}, \tag{3.18}$$

where $b' := (u_1^\top k, \ldots, u_r^\top k) \oplus b$, with $r := |U|$.

Proof. Observe that $|\mathcal{R}^b_U| = |\mathcal{R}^{b'}_U|$ since the masks in U are assumed linearly independent. It holds that

$$\begin{aligned}
(-1)^{u^\top k} \cdot C^{F\circ X_k}_{v,u}|_{x \in \mathcal{R}^{b'}_U} &= (-1)^{u^\top k} \cdot \frac{1}{|\mathcal{R}^{b'}_U|} \sum_{x \in \mathcal{R}^{b'}_U} (-1)^{u^\top x \oplus v^\top (F\circ X_k)(x))} \\
&= (-1)^{u^\top k} \cdot \frac{1}{|\mathcal{R}^{b'}_U|} \sum_{x;(u_1^\top x, \ldots, u_r^\top x) = b'} (-1)^{u^\top x \oplus v^\top F(x \oplus k)} \\
&= (-1)^{u^\top k} \cdot \frac{1}{|\mathcal{R}^{b'}_U|} \sum_{y \oplus k;(u_1^\top (y\oplus k), \ldots, u_r^\top (y\oplus k)) = b'} (-1)^{u^\top (y\oplus k) \oplus v^\top F(y)} \\
&= (-1)^{u^\top k} \cdot \frac{1}{|\mathcal{R}^b_U|} \sum_{y \oplus k;(u_1^\top y, \ldots, u_r^\top y) = b} (-1)^{u^\top y \oplus v^\top F(y)} \cdot (-1)^{u^\top k} \\
&= \frac{1}{|\mathcal{R}^b_U|} \sum_{y \oplus k;(u_1^\top y, \ldots, u_r^\top y) = b} (-1)^{u^\top y \oplus v^\top F(y)} \\
&\overset{(1)}{=} \frac{1}{|\mathcal{R}^b_U|} \sum_{y;(u_1^\top y, \ldots, u_r^\top y) = b} (-1)^{u^\top y \oplus v^\top F(y)} \\
&= C^F_{v,u}|_{x \in \mathcal{R}^b_U}
\end{aligned}$$

Observe that step (1) holds because X_k is an automorphism; it only permutes the order in which the elements are considered in the summation. □

We can deduce from Lemma 3.10 that prepending F with an affine translation has two effects on the conditional correlation of the approximation $\langle u, v\rangle|_U$. First, the sign of the conditional correlations may be inverted, depending on the value of $u^\top k$. Since the correlation magnitude remains constant, this does not have to be accounted for in extending $\langle u, v\rangle|_U$ to $F \circ X_k$. The second effect is that the conditional correlations are permuted among the classes, depending on the value of $(u_1^\top k, \ldots, u_r^\top k)$. It is not possible to modify the approximation to account for this and thus means that it is no longer clear which class corresponds with which conditional correlation when the value of k is unknown.

In Table 3.5 we illustrate both effects with the conditional approximation $\langle \omega_3, v\rangle|_{\{\omega_1,\omega_6\}}$ applied to the function $\texttt{inv} \circ X_k$. The table illustrates that the value of $(\omega_1^\top k, \omega_6^\top k)$ determines which class $b = (\omega_1^\top x, \omega_6^\top x)$ yields the conditional correlation with magnitude 2^{-1}. Moreover, the sign of the correlation is determined by $(-1)^{\omega_3^\top k}$.

Table 3.5: $C^{\texttt{inv}\circ X_k}_{v,\omega_3}\Big|_{x\in\mathcal{R}^b_U}$ for $k \in \mathbb{F}_2^8$, $b \in \mathbb{F}_2^2$ and $U = \{\omega_1, \omega_6\} \subset \texttt{span}(\Omega_v)$.

			$\omega_1^\top x$	0	0	1	1
			$\omega_6^\top x$	0	1	0	1
$\omega_3^\top k$	$\omega_1^\top k$	$\omega_6^\top k$					
0	0	0		2^{-1}	0	0	0
	0	1		0	2^{-1}	0	0
	1	0		0	0	2^{-1}	0
	1	1		0	0	0	2^{-1}
1	0	0		-2^{-1}	0	0	0
	0	1		0	-2^{-1}	0	0
	1	0		0	0	-2^{-1}	0
	1	1		0	0	0	-2^{-1}

With Lemma 3.11 we lastly discuss the composition $F \circ L$. The lemma demonstrates that a conditional approximation $\langle u, v\rangle|_U$ for F can be adapted to approximate $F \circ L$ by replacing u with $M_L^\top u$ and the masks in U with $\{M_L^\top u_i \mid u_i \in U\}$.

Lemma 3.11. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be an arbitrary vectorial Boolean function and let $L : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an arbitrary linear permutation. For arbitrary $u \in \mathbb{F}_2^n$, $v \in \mathbb{F}_2^m$ and $U \subseteq \mathbb{F}_2^n$ it holds that*

$$C^{F\circ L}_{v,u'}|_{x\in\mathcal{R}^b_{U'}} = C^F_{v,u}|_{x\in\mathcal{R}^b_U}, \tag{3.19}$$

where $u' := M_L^\top u$, $U' := \{M_L^\top u_i \mid u_i \in U\}$.

Proof. The proof to this lemma is analogous to that of Lemma 3.10 and can be found in Appendix A.

We have thus illustrated that a conditional approximation for F can be extended to one for F composed with an affine transformation without impacting the conditional correlation magnitudes. The only exception to this is the composition $F \circ X_k$. We observed for this composition that an approximation for F can also be used to approximate $F \circ X_k$, but that the conditional correlations are permuted among the plaintext classes, depending on the value of k. When k is unknown, this yields that it is unknown which plaintext class yields the desired conditional correlation.

This finalizes the mathematical analysis of the weighted and conditional approximation and their application to the `inv` function. In the following chapter we investigate the implications of these result for the AES. In particular, we demonstrate how the conditional approximation can be used to launch a distinguishing attack against four-round AES.

Chapter 4

Application to 4-round AES

The `inv` function is used bountifully in the AES encryption algorithm, given its presence in the `SubBytes` function. As suggested in Section 2.3.3, one of the primary goals of this function in AES is to thwart linear attacks against the cipher. It has in fact been argued by the designers that the correlation contribution of a linear trail for four rounds of AES is upper bounded by 2^{-75} [17, Section 9.5.2], which in symmetric-key folklore is understood to mean that this round-reduced cipher can withstand distinguishing attacks that rely on linear cryptanalysis. However, this upper bound depends on the assumption that the correlation of the best approximation for the `inv` function is upper bounded by 2^{-3}. The fact that we have presented two novel approximation techniques that both exceed this bound thus suggests that these four rounds might actually be vulnerable to such an attack. In this chapter we confirm this suspicion by developing a statistical distinguishing attack against this round-reduced version of the cipher.

Working towards this goal, we first review the Wide Trail Strategy in Section 4.1. This design strategy lies at the core of the AES design and forms the basis for the aforementioned security argument. This is followed by the construction of the four-round distinguisher in Section 4.2. We then use this distinguisher in Section 4.3 to construct both a distinguishing attack, as well as a key-recovery attack against the cipher. The chapter closes with an experimental validation of the constructed conditional approximation used in the distinguisher in Section 4.4.

4.1 The Wide Trail Strategy

The *Wide Trail Strategy* (WTS) was first introduced as a block cipher design philosophy by Daemen in 1995 [11]. Throughout the design of the SHARK [50], SQUARE [13], and BKSQ [14] ciphers, the concept was further developed into a broadly-applicable block cipher design framework [15], ultimately forming the basis of the Rijndael cipher. Following Rijndael's standardization as AES in 2000, the WTS has seen more widespread use, for example in the block ciphers Fides [7] and LED [23], and the hash-function Photon [22].

The strategy attempts to avert linear approximations with large correlations from forming in a cipher by forcing the correlation contribution of all linear trails through the cipher to be small. To achieve this, the strategy advocates the use of key-alternating SPNs with a specific format for the substitution function $\gamma : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and permutation function $\theta : \mathbb{F}_2^n \to \mathbb{F}_2^n$. Here, *bricklayer function* γ introduces non-linear behaviour into the cipher by treating the n-bit block as the concatenation of m-bit bundles and applying a non-linear permutation to each bundle — these permutations are referred to as the s-boxes of γ. Meanwhile, the permutation function θ is a linear transformation aimed at spreading information between the bundles. This concept of information spreading is closely linked to that of *diffusion*, which was first introduced by Shannon [55] to denote the quantitative spreading of information.

Observe that the encryption function of AES exhibits this exact structure. Operating on 128-bit blocks, the encryption function makes use of bundles with size $m = 8$. The role of γ is fulfilled by the `SubBytes` function, which applies the non-linear, invertible s-box S to each byte in the state. The function θ is composed of the functions `ShiftRows` and `MixColumns`. In this composition, `ShiftRows` first moves bundles that are close to each other in the context of `MixColumns` to positions that are distant. Following this, `MixColumns` is applied to groups of four bundles, 'mixing' their contents.

Computing the correlation contribution of a proper linear trail for such a cipher is rather straightforward. Following Equation 2.9, only the linear approximations for the round-functions influence the correlation magnitude, while Lemma 3.9 demonstrates that a proper linear approximation for θ yields a correlation of amplitude 1. As such, only the approximations over the function γ need to be considered. According to [17, Equation 7.37], the correlation of an approximation for a bricklayer function is computed as

$$C^{\gamma}_{v,u} = \prod_{i=0}^{n/m-1} C^{S_i}_{[v]_i,[u]_i}, \tag{4.1}$$

where S_i denotes the ith s-box in γ, and $[v]_i$ denotes the mask for the ith m-bit bundle. As such, the correlation of a trail is computed as the product of the approximation correlations for each of the s-boxes.

When all s-boxes are the same — as is the case for AES — this expression allows for the creation of a simple upper bound on the correlation of a trail. For any chosen permutation function θ it is possible to compute a lower bound n_r on the number of *active* s-boxes any r-round trail must contain. We consider an s-box active in a trail when it is assigned a non-zero input mask, output mask, or both. When we furthermore observe that $C^{S}_{v,u} = 1$ if and only if $u = v = 0$, we can see that the absolute correlation contribution of an r-round linear trail is upper bounded by c^{n_r}, where c is an upper bound on the magnitude of the correlation of an approximation for the s-box. It was proven by the designers of Rijndael that $n_4 = 25$ for the permutation function $\theta = \texttt{MC} \circ \texttt{SR}$. We have moreover seen in Chapter 3 that the maximum correlation magnitude attainable for a linear approximation of the s-box is $c = 2^{-3}$. As such, it is concluded that the correlation contribution of any linear trail for four-round AES is upper bounded by $(2^{-3})^{25} = 2^{-75}$.

This bound is understood to imply that no linear attack against four-round AES can succeed. There is, however, a caveat in this argumentation. In particular, it is implicitly assumed that a single linear approximation yields the strongest approximation technique for approximating an s-box and in particular the `inv` function; an assumption we have previously shown to be incorrect. It is this incorrect assumption that we will exploit to attack four-round AES.

4.2 Constructing the distinguisher

Before we can launch a distinguishing attack against four-round AES, we must design the distinguisher that sets four-round AES apart from a random permutation. To this end, we will be constructing a conditional approximation that has a large absolute correlation with the four-round AES function, yet is expected to yield a small correlation for an arbitrary permutation. In particular, we will construct a four-round trail that we assume to be the dominant contributor to the correlation of this approximation.

4.2.1 Strategy

We have seen in Section 4.1 that the correlation of a trail for 4-round AES is upper bounded by the product of the correlations achieved when trailing each s-box. Given that any 4-round AES trail must encounter at least $n_4 = 25$ s-boxes, we construct an approximation that encounters exactly 25 and achieves a high correlation for as many of as possible.

Let us first observe that we can construct a conditional approximation for the s-box S that achieves a 2^{-1} correlation on one of four plaintext classes. To see this, recall that we have previously shown such a conditional approximation to exist for `inv`. When we furthermore recall Lemma 3.8 and the fact that the s-box $S = \texttt{T} \circ \texttt{L} \circ \texttt{inv}$ with `L` and `T` affine transformations, it follows that this approximation can be modified to approximate S with a 2^{-1} conditional correlation on one plaintext class. Note here that this approximation can still be created for any output mask $v \in \mathbb{F}_2^8 \setminus \{0\}$.

We can, however, not use this approximation technique to approximate the s-boxes in all four rounds. To see why, let us recall the AES encryption function restricted to four rounds:

$$\texttt{AES}^4_k = \texttt{A}_{k_5} \circ \texttt{RF} \circ \texttt{A}_{k_4} \circ \texttt{RF} \circ \texttt{A}_{k_3} \circ \texttt{RF} \circ \texttt{A}_{k_2} \circ \texttt{RF} \circ \texttt{A}_{k_1} \circ \texttt{RF} \circ \texttt{A}_{k_0}, \tag{4.2}$$

where $\texttt{RF} = \texttt{MC} \circ \texttt{SR} \circ \texttt{SB}$. Observe here that the `SubBytes` function in the first round is only preceded by the affine translation $\texttt{A}_{k_0}$ and that the `SubBytes` function in the last round is only followed by affine transformation

$\mathtt{A}_{k_5} \circ \mathtt{MC} \circ \mathtt{SR}$. Meanwhile, the SubBytes functions in the middle rounds are both preceded and followed by other, non-linear SubBytes functions. Since we have thus far only discovered means to extend the conditional approximation over affine transformations, we are restricted to using this technique to approximate the s-boxes in the first and last round. It was decided to further constrain ourselves to only approximate s-boxes in the first round using the novel technique; we will approximate the s-boxes in the second, third and fourth round using a standard linear trail.

As we will observe during construction, it is possible to build these approximations such that the first round encounters sixteen s-boxes, while nine are trailed in subsequent rounds. We commence with constructing three one-round linear approximations for the last three rounds, where special attention is paid to the approximation for the approximation of the third round. Hereafter, the conditional approximation for the first round function is formed. Ultimately, the four approximations are combined and the four-round distinguisher is created.

4.2.2 Linear approximations for the round function

Constructing a linear approximation $\langle u, v \rangle$ for the round function RF can itself be decomposed in the construction of a linear trail $\lambda = (u, \lambda_1, \lambda_2, v)$, where the approximations $\langle u, \lambda_1 \rangle$, $\langle \lambda_1, \lambda_2 \rangle$ and $\langle \lambda_2, v \rangle$ approximate the functions SubBytes, ShiftRows and MixColumns, respectively. There is exactly only one trail with a non-zero correlation contribution for this approximation. Since ShiftRows and MixColumns are both linear, it follows from Lemma 3.9 that the correlation of their approximation is zero unless $\lambda_1 = M_{\mathtt{SR}}^{\top}\lambda_2$ and $\lambda_2 = M_{\mathtt{MC}}^{\top}v$, respectively. As such, the linear trail $\lambda' := (u, M_{\mathtt{SR}}^{\top}M_{\mathtt{MC}}^{\top}v, M_{\mathtt{MC}}^{\top}v, v)$ is the only — and thus dominant — contributor of the linear approximation $\langle u, v \rangle$. This approximation therefore yields a correlation of

$$C^{\mathtt{RF}}_{v,u} = C^{\mathtt{RF}}_{\lambda'} = C^{\mathtt{SB}}_{M_{\mathtt{SR}}^{\top}M_{\mathtt{MC}}^{\top}v,u}. \tag{4.3}$$

Note that this implies that an approximation for the SubBytes function induces an approximation for the full round function, allowing us to focus on constructing the former.

With this in mind, we will now construct a linear trail $\lambda = (\lambda_1, \lambda_2, \lambda_3, \lambda_4)$ for the second, third and fourth round. Since the conditional approximation of the first round will encounter sixteen s-boxes, this trail should encounter the remaining nine. Adhering to this, we will construct our three-round trail such that there are four active s-boxes in the first and last round, and a single s-box in the middle. Since it should only contain one active s-box, we start with constructing the linear approximation $\langle \lambda_2, \lambda_3 \rangle$ for the middle round. In particular, we focus on the construction of the approximation $\langle \lambda_2, \lambda_3'' \rangle$ for the SubBytes function, as the approximations $\langle \lambda_3'', \lambda_3' \rangle$ for ShiftRows and $\langle \lambda_3', \lambda_3 \rangle$ for MixColumns will follow automatically. Because this round should only contain one active s-box, the input mask λ_2 should be chosen such that it contains only one non-zero byte. Let us choose $\lambda_2 = (\mathtt{01_x}, \mathtt{00_x}, \ldots, \mathtt{00_x}) \in \mathbb{F}_{2^8}^{16}$. We must now choose the output mask λ_3'' such that the correlation magnitude $|C^{\mathtt{SB}}_{\lambda_3'',\lambda_2}|$ of this approximation is maximized. Since SubBytes is a bricklayer function, this correlation can be computed as

$$C^{\mathtt{SB}}_{\lambda_3'',\lambda_2} = \prod_{i=0}^{15} C^{S}_{[\lambda_3'']_i,[\lambda_2]_i} = C^{S}_{[\lambda_3'']_0,\mathtt{01_x}} \cdot \prod_{i=1}^{15} C^{S}_{[\lambda_3'']_i,\mathtt{00_x}}. \tag{4.4}$$

Observe that the magnitude of this correlation is maximized when the absolute correlation of the approximations over each s-box is maximized. Since $C^{S}_{v,0} = \delta(v)$, it follows that we should choose $[\lambda_3'']_i = 0$ for $1 \leq i \leq 15$. Leveraging lemmas 2.3 and 3.9, we moreover find that

$$C^{S}_{v,u} = C^{\mathtt{T}\circ\mathtt{L}\circ\mathtt{inv}}_{v,u} = (-1)^{v^{\top}\mathtt{63_x}} \cdot C^{\mathtt{L}\circ\mathtt{inv}}_{v,u} = (-1)^{v^{\top}\mathtt{63_x}} \cdot C^{\mathtt{inv}}_{M_{\mathtt{L}}^{\top}v,u}. \tag{4.5}$$

As such, $|C^{S}_{v,\mathtt{01_x}}|$ is maximized at 2^{-3} when $\mathtt{01_x} \in \Omega_{M_{\mathtt{L}}^{\top}v}$, or equivalently $M_{\mathtt{L}}^{\top}v \in \Omega_{\mathtt{01_x}}$ due to the symmetry of inv. As remarked in Section 3.1, there are exactly five masks $v \in \mathbb{F}_2^8 \setminus \{0\}$ for which this holds, namely $\mathtt{48_x}$, $\mathtt{50_x}$, $\mathtt{88_x}$, $\mathtt{90_x}$ and $\mathtt{C0_x}$. We opt to use $\mathtt{88_x}$, thus making $\lambda_3'' = (\mathtt{88_x}, \mathtt{00_x}, \ldots, \mathtt{00_x})$. With this mask in place, the values of the masks λ_3' and λ_3 are fixed, because the correlation of the approximations $\langle \lambda_3'', \lambda_3' \rangle$ and $\langle \lambda_3', \lambda_3 \rangle$ is only non-zero when $\lambda_3' = (M_{\mathtt{SR}}^{-1})^{\top}\lambda_3''$ and $\lambda_3 = (M_{\mathtt{MC}}^{-1})^{\top}\lambda_3'$. We provide a schematic representation of the resulting linear trail in Figure 4.1. To improve readability, the zero-bytes of the four masks have been left blank in the figure.

With the masks λ_2 and λ_3 fixed, now the masks λ_1 and λ_4 have to be chosen. Given that the approximations $\langle \lambda_1, \lambda_2 \rangle$ and $\langle \lambda_3, \lambda_4 \rangle$ both encounter four s-boxes, these masks should be chosen such that the correlation for either approximation is $(2^{-3})^4 = 2^{-12}$. We present our chosen trails in figures 4.2 and 4.3. Concatenating the three one-round approximations yields the three-round linear trail presented in Figure 4.4. Note that the four masks presented in this table also act as the masks for each of the four round keys k_1, k_2, k_3 and k_4 used in $\mathtt{AES}_k^4$.

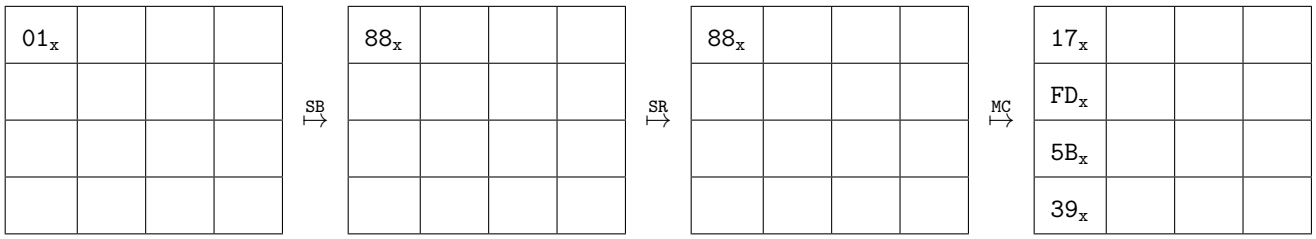

Figure 4.1: One-round trail $(\lambda_2, \lambda_3'', \lambda_3', \lambda_3)$ for the third round.

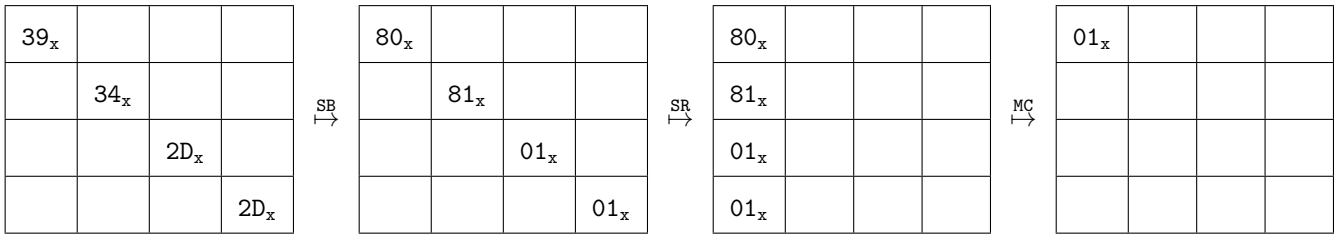

Figure 4.2: One-round trail $(\lambda_1, \lambda_2'', \lambda_2', \lambda_2)$ for the second round.

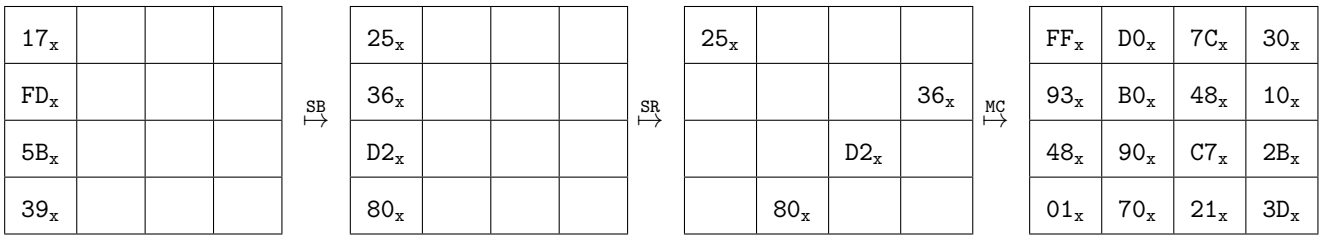

Figure 4.3: One-round trail $(\lambda_3, \lambda_4'', \lambda_4', \lambda_4)$ for the fourth round.

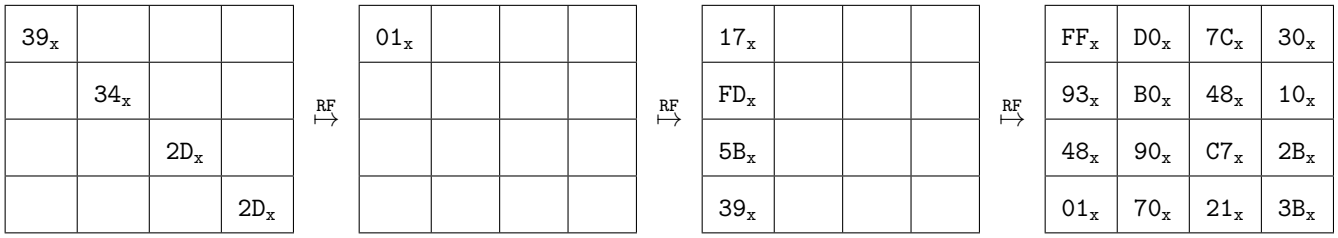

Figure 4.4: Constructed three-round trail $(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$.

4.2.3 A 1-round conditional linear trail

With the three-round linear trail in place, we now construct the 1-round conditional trail $\langle \lambda_0, \lambda_1 \rangle|_U$ for the first round. To derive the construction rules for a conditional approximation of an *bricklayer function* such as `SubBytes`, we introduce Lemma 4.1. In this lemma $u|v : \mathbb{F}_2^n \times \mathbb{F}_2^m \to \mathbb{F}_2^{n+m}$ is used to denote the vector concatenation operator, whilst $\mathbf{0}_r$ denotes the zero-vector of length r.

Lemma 4.1. *Let $F_1 : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and $F_2 : \mathbb{F}_2^m \to \mathbb{F}_2^m$ be arbitrary vectorial Boolean functions. For arbitrary masks $u_1, v_1 \in \mathbb{F}_2^n$ and $u_2, v_2 \in \mathbb{F}_2^m$, sets $U_1 \subset \mathbb{F}_2^n$ and $U_2 \subset \mathbb{F}_2^m$ with linearly independent vectors, and class indicators $b_1 \in \mathbb{F}_2^{|U_1|}$ and $b_2 \in \mathbb{F}_2^{|U_2|}$ it holds that*

$$C^{F_1}_{v_1,w_1}\Big|_{x \in \mathcal{R}^{b_1}_{U_1}} \cdot C^{F_2}_{v_2,w_2}\Big|_{x \in \mathcal{R}^{b_2}_{U_2}} = C^{F}_{v,w}\Big|_{x \in \mathcal{R}^{b}_{U}} \tag{4.6}$$

where $F : \mathbb{F}_2^{n+m} \to \mathbb{F}_2^{n+m} : x|y \mapsto F_1(x)|F_2(y)$ is the bricklayer-style application of F_1 and F_2, $v := v_1|v_2$,

$w := w_1|w_2$, $b := b_1|b_2$ *and* $U := \{u_i|\mathbf{0}_m \mid u_i \in U_1\} \cup \{\mathbf{0}_n|u_j \mid u_j \in U_2\}$.

Proof. Let $r_1 := |U_1|$, $r_2 := |U_2|$ and $r := |U| = r_1 + r_2$. Observe in the first place that

$$|\mathcal{R}_{U_1}^{b_1}| \cdot |\mathcal{R}_{U_2}^{b_2}| = 2^{n-r_1} \cdot 2^{m-r_2} = 2^{n+m-r_1-r_2} = 2^{n+m-r} = |\mathcal{R}_U^b|.$$

It moreover holds that for an arbitrary $y \in \mathbb{F}_2^n$ and $z \in \mathbb{F}_2^m$, the following expressions are equivalent:

$$\begin{aligned} y|z \in \mathcal{R}_U^b &\equiv ((u_1|\mathbf{0}_m)^\top(y|z), \ldots, (\mathbf{0}_n|u_r)^\top(y|z)) = (b_1|b_2) \\ &\equiv (u_1^\top y, \ldots, u_{r_1}^\top y, u_{r_1+1}^\top z, \ldots, u_r^\top z) = (b_1, \ldots, b_r) \\ &\equiv y \in \mathcal{R}_{U_1}^{b_1} \text{ and } z \in \mathcal{R}_{U_2}^{b_2}. \end{aligned}$$

As a result, it holds that

$$\begin{aligned} C_{v,w}^F|_{x \in \mathcal{R}_U^b} &= \frac{1}{|\mathcal{R}_U^b|} \sum_{x \in \mathcal{R}_U^b} (-1)^{w^\top x \oplus v^\top F(x)} \\ &= \frac{1}{|\mathcal{R}_{U_1}^{b_1}|} \cdot \frac{1}{|\mathcal{R}_{U_2}^{b_2}|} \sum_{y|z \in \mathcal{R}_U^b} (-1)^{(w_1|w_2)^\top(y|z) \oplus (v_1|v_2)^\top(F_1(y)|F_2(z))} \\ &= \frac{1}{|\mathcal{R}_{U_1}^{b_1}|} \cdot \frac{1}{|\mathcal{R}_{U_2}^{b_2}|} \sum_{y \in \mathcal{R}_{U_1}^{b_1}} \sum_{z \in \mathcal{R}_{U_2}^{b_2}} (-1)^{w_1^\top y \oplus w_2^\top z \oplus v_1^\top F_1(y) \oplus v_2^\top F_2(z)} \\ &= \frac{1}{|\mathcal{R}_{U_1}^{b_1}|} \sum_{y \in \mathcal{R}_{U_1}^{b_1}} (-1)^{w_1^\top y \oplus v_1^\top F_1(y)} \cdot \frac{1}{|\mathcal{R}_{U_2}^{b_2}|} \sum_{z \in \mathcal{R}_{U_2}^{b_2}} (-1)^{w_2^\top z \oplus v_2^\top F_2(z)} \\ &= C_{y \in \mathcal{R}_{U_1}^{b_1}}^{F_1} \cdot C_{y \in \mathcal{R}_{U_2}^{b_2}}^{F_2} \end{aligned}$$

where $y \in \mathbb{F}_2^n$, $z \in \mathbb{F}_2^m$. □

This lemma thus states that the conditional correlation of an approximation for a bricklayer function is the product of the conditional correlations with the individual s-boxes, restricted to a specific plaintext class. When we apply this to the `SubBytes` function, we find that it suffices to construct sixteen conditional approximations $\langle u_i, v_i \rangle_{U_i}$ for S, and that the correlation of the their combination is computed as

$$C_{v,w}^{\mathtt{SB}}\Big|_{x \in \mathcal{R}_U^b} = \prod_{i=0}^{15} C_{v_i,u_i}^S\Big|_{x \in \mathcal{R}_{U_i}^{b_i}}, \tag{4.7}$$

where $v = v_0|\ldots|v_{15}$, $u = u_0|\ldots|u_{15}$, $b = b_0|\ldots|b_{15}$ and $U = \bigcup_{i=0}^{15}\{\mathbf{0}_{8i}|u|\mathbf{0}_{120-8i} \mid u \in U_i\}$.

It thus follows that the construction of the linear approximation $\langle \lambda_0, \lambda_1 \rangle$ used in this conditional approximation is almost identical to that of the previous three. We do, however, have to pay closer attention to the construction of the input masks to the s-boxes. In addition to adhering to the relation shown in Equation 4.5, we have to make sure that the input mask u_i of each s-box is chosen such that it does not equal $\omega_1 \in \Omega_{M_{\mathtt{L}}^\top v_i}$, with v_i the output mask of the s-box. Recall here we have seen at the end of Section 3.2.2 that for any $v \in \mathbb{F}_2^8 \setminus \{0\}$, the conditional approximation $\langle \omega, v \rangle|_U$ only yields a 2^{-1} correlation with `inv` on $\mathcal{R}_U^{(0,0)}$ with $U = \{\omega_1, \omega_6\} \subset \mathtt{span}(\Omega_v)$ when $\omega \in \Omega_v \setminus \{\omega_1\}$. In Table 4.5 the chosen trail for the approximation $\langle \lambda_0, \lambda_1 \rangle$ is presented.

It remains to construct the conditioning set U. The conditioning set U_i for the approximation $\langle [\lambda_0]_i, [\lambda_1'']_i \rangle|_{U_i}$ of the ith s-box in `SubBytes` is computed as $\{\omega_1, \omega_6\} \subset \mathtt{span}(\Omega_v)$, with $v = M_{\mathtt{L}}^\top [\lambda_1'']_i$. We have moreover seen in Equation 4.7 that these s-box conditioning sets can be combined as $\bigcup_{i=0}^{15}\{\mathbf{0}_{8i}|u|\mathbf{0}_{120-8i} \mid u \in U_i\}$ to form the conditioning set U for the `SubBytes` function. The resulting set is presented in Equation 4.8. Here, the ω_1-masks form the left column, while the ω_6-masks are presented on the right. To enhance the readability of this set, all zero-bytes are represented using $\cdot\cdot$.

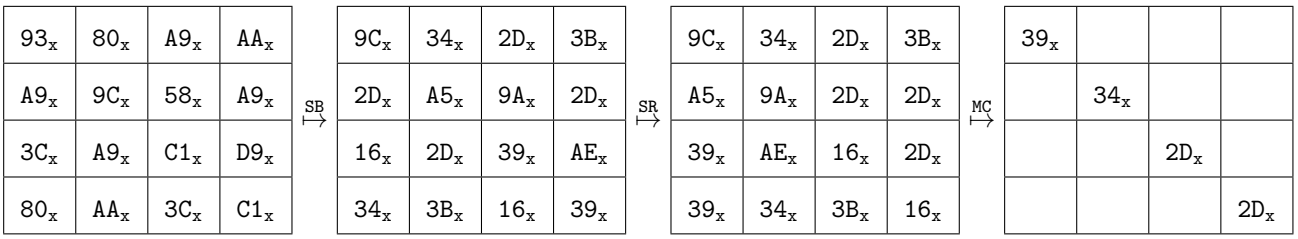

Figure 4.5: One-round trail $(\lambda_0, \lambda_1'', \lambda_1', \lambda_1)$ for first round.

$$
\begin{aligned}
U = \{&\texttt{E5} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x, \quad \texttt{3A} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x \\
&\cdot\cdot \texttt{93} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x, \quad \cdot\cdot \texttt{20} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x \\
&\cdot\cdot \cdot\cdot \texttt{62} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x, \quad \cdot\cdot \cdot\cdot \texttt{19} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x \\
&\cdot\cdot \cdot\cdot \cdot\cdot \texttt{7D} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x, \quad \cdot\cdot \cdot\cdot \cdot\cdot \texttt{EA} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x \\
&\cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{7D} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x, \quad \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{EA} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x \\
&\cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{C0} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x, \quad \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{6B} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x \\
&\cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{93} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x, \quad \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{20} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x \\
&\cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{79} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x, \quad \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{CE} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x \\
&\cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{93} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x, \quad \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{20} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x \\
&\cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{94} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x, \quad \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{49} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x \\
&\cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{7A} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x, \quad \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{F9} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x \\
&\cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{62} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x, \quad \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{19} \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot_x \\
&\cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{79} \cdot\cdot \cdot\cdot \cdot\cdot_x, \quad \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{CE} \cdot\cdot \cdot\cdot \cdot\cdot_x \\
&\cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{93} \cdot\cdot \cdot\cdot_x, \quad \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{20} \cdot\cdot \cdot\cdot_x \\
&\cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{9D} \cdot\cdot_x, \quad \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{F2} \cdot\cdot_x \\
&\cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{7A}_x, \quad \cdot\cdot \cdot\cdot \cdot\cdot \cdot\cdot \texttt{F9}_x\}
\end{aligned}
\tag{4.8}
$$

When the four one-round approximations are concatenated, the trail $(\lambda_0, \lambda_1, \lambda_2, \lambda_3, \lambda_4)$ is formed. By construction, this trail has a large correlation contribution of $(2^{-3})^{25} = 2^{-75}$ to the approximation $\langle \lambda_0, \lambda_4 \rangle$, which we present in Figure 4.6. Combined with the conditioning set U, the conditional approximation $\langle \lambda_0, \lambda_4 \rangle|_U$ is formed, which is expected to yield a correlation magnitude of $(2^{-1})^{16} \cdot (2^{-3})^9 = 2^{-43}$ on exactly one of the 2^{32} plaintext classes induced by U.

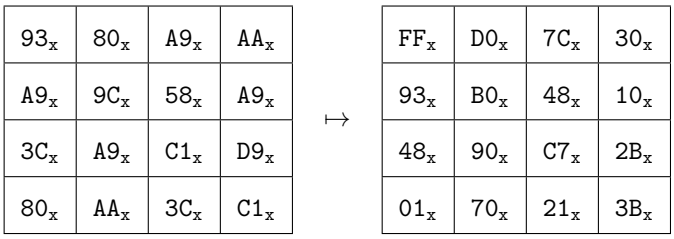

Figure 4.6: Four-round approximation $\langle \lambda_0, \lambda_4 \rangle$.

4.3 Attacking four-round AES

With the four-round approximation in place, we can now exploit it to perform both a distinguishing as well as a key-recovery attack on the four-round AES cipher.

4.3.1 Distinguishing attack

For the distinguishing attack, the adversary retrieves all possible plaintext-ciphertext pairs from the challenger. The adversary partitions these pairs based on the plaintext using the conditioning set U, forming 2^{32} classes containing 2^{96} pairs each. The adversary now computes the correlation of each class with the approximation $\langle \lambda_0, \lambda_4 \rangle$ and concludes that the challenger encrypted the pairs using four-round AES if and only if exactly one class yields a correlation with a magnitude of 2^{-43}.

To see why this is a distinguishing property, we illustrate that one is very unlikely to observe a 2^{-43} correlation when the challenger uses an arbitrary permutation. Let us view the permutation E used by the challenger as a set of 2^{32} 'sub-functions' E^b, where we define each sub-function as $E^b : \mathcal{R}_U^b \mapsto I_U^b$, with $I_U^b := \{E_k^b(x) \mid x \in \mathcal{R}_U^b\}$. The assumption that E is chosen uniformly random is now equivalent to assuming that all of these E^b are chosen uniformly random. Observe that we can view these sub-functions E^b as equivalent to 96-bit permutations, since $|\mathcal{R}_U^b| = |I_U^b| = 2^{96}$.

Let us now recall the work by Bogdanov and Tischhauser [10], where it is argued that the correlation of a linear approximation with an arbitrary n-bit permutation can be modelled using the normal distribution $\mathcal{N}(0, 2^{-n})$. Applied to our specific case, we would thus expect the correlation of $\langle \lambda_0, \lambda_1 \rangle$ with each sub-permutation to be distributed according to $N := \mathcal{N}(0, 2^{-96})$. We can compute the probability that an arbitrary 96-bit permutation yields a correlation magnitude of 2^{-43} with the constructed trail as

$$\mathbb{P}\big[|X| > 2^{-43}\big] = 2 \cdot \mathbb{P}\big[X > 2^{-43}\big] = 2 \cdot \mathbb{P}\big[2^{-48} \cdot X > 2^5\big] = 2 \cdot \Phi(-2^5) < 2^{-743.987}, \tag{4.9}$$

where $X \sim N$. Given that an arbitrary permutation is composed of 2^{32} arbitrary sub-permutations, the probability that at least one of the sub-permutations achieves the desired correlation can be approximated as $2^{-743} \cdot 2^{32} = 2^{-711}$. This probability is so incredibly small that we conclude it to be a distinguishing property for 4-round AES.

4.3.2 Key-recovery attack

In addition to the distinguishing attack, this conditional approximation also provides us means to launch a key-recovery attack against the cipher. In particular, by determining the plaintext class with the non-zero correlation, we can recover 32 bits of information about the key. Recall that we have seen in Table 3.5 that the absolute correlation of $\langle \omega_2, v \rangle|_{\{\omega_1, \omega_6\}}$ with $\texttt{inv} \circ X_k$ is 2^{-1} on $\mathcal{R}_U^b$ when $b = (\omega_1^\top k, \omega_6^\top k)$. Thus, by uncovering the plaintext class $\mathcal{R}_U^b$ for which our approximation $\langle \lambda_0, \lambda_4 \rangle|_U$ yields a correlation magnitude of 2^{-43}, we can recover the value of $u^\top k$ for all $u \in U$. Note that this is equivalent to recovering 32-bits of information about the key since $|U| = 32$.

Complexity. Let us discuss the number of plaintext-ciphertext samples necessary to perform this attack. Because in this setting it is known that the challenger uses the 4-round AES to encrypt the pairs, it suffices to determine *which* class yields the 2^{-43} correlation, rather than determining *whether* such a class is present. This allows for a reduction in the data complexity compared to the distinguishing attack.

In this attack, we request data from the challenger and use this data to determine the sample correlation for each plaintext class. Given a sufficiently large number of samples per class, it is expected that the unique class with the non-zero correlation yields the sample correlation with the greatest magnitude. We can thus use these sample magnitudes to order the plaintext classes, and expect the correct class to be near the head of the ordering.

Despite not being an Algorithm-2 attack exactly, we can still use Lemma 2.5 to compute the data complexity of this attack: we are still attempting to distinguish one non-zero correlation class from a large number of zero-correlation classes. The only difference is that we are now working with plaintext classes instead of key classes. We do, however, have to account for the fact that the lemma will only yield a data complexity t' for a single class, instead of the combined data complexity of all classes. In Table 4.1 we present the data complexity per class for fixed success probability 0.95 and varying advantage a. Based on this table, we can conclude that with 2^{92} plaintext-ciphertext pairs per class, we should be able to discern the non-zero correlation plaintext class in at least 95% of experiments.

Table 4.1: Data complexity per plaintext class in terms of advantage.

a	1	8	16	32
$\log_2(t')$	88.43	90.36	91.16	91.99

The second step is then to determine the total data complexity t. Here, one has to account for the fact that t' samples must be encountered for each class, while we assume that in the known-plaintext model samples are gathered uniformly at random from the full plaintext-ciphertext space. As such, after working through $2^{32} \cdot t'$ arbitrary plaintext-ciphertext pairs, it is very unlikely that exactly t' were seen for each class: it is expected that some classes will still be short some samples. The problem of determining the necessary sample size t is equivalent to a problem known in literature as the *dixie-cup problem*. Here, one attempts to obtain m copies of n unique objects by uniformly sampling these objects with replacement. It was shown by Newman [44] that the expected number of objects $E_m(n)$ one must sample before obtaining m instances of each of the n types equals

$$E_m(n) = n \ln n + (m-1) \cdot n \ln \ln n + \mathcal{O}(1) \tag{4.10}$$

as n approaches infinity. In our situation, we attempt to encounter $m = t'$ samples for $n = 2^{32}$ plaintext classes and are thus interested in total number of expected samples $t = E_m(n)$. We present the total data complexity t in terms of the advantage a in Table 4.7.

Figure 4.7: Expected data complexity in terms of advantage.

a	1	8	16	32
$\log_2(t)$	122.06	123.99	124.79	125.62

We conclude from this table that $2^{125.62}$ data samples will allow us to extract the non-zero correlation class with 95% certainty. Since each sample is only used in computing the sample correlation of one class, the time complexity of this attack is equal to the data complexity. The memory complexity of this attack is $\mathcal{O}(2^{32})$, since each of the 2^{32} sample correlations must be stored simultaneously. Note that this attack is of interest: the data complexity is smaller than the size of the full code book, while the time necessary to determine the non-zero class — and thus 32 bits of the key — is shorter than an exhaustive key search.

4.4 Experimental validation

To close, we experimentally validate the conditional approximation we created. Given that the data complexity of both the distinguishing attack as well as the key recovery attack are beyond what we can compute in reasonable time, we only validate a part of the distinguisher. Because linear approximations have been extensively researched, it was decided to focus the experimental validation on the conditional linear approximation $\langle \lambda_0, \lambda_1 \rangle|_U$ created for the first round. For this approximation, we expect that exactly one of the 2^{32} plaintext classes will yield a 2^{-16} correlation with $\texttt{AES}_k^1$, while the others all achieve a correlation of 0. To validate this hypothesis, we have devised the following experiment:

Experiment. To start, a master key k is randomly generated. Based on this key, it is determined which plaintext class is expected to yield the non-zero correlation with the $\texttt{AES}_k^1$ cipher. For this and fifteen arbitrary other classes, a sample correlation with the approximation $\langle \lambda_0, \lambda_1 \rangle$ is computed using $t' = 2^{35.61}$ uniformly random sampled plaintext-ciphertext pairs per class. The same samples are furthermore used to determine the sample correlation of the approximation $\langle 0, \lambda_1 \rangle$ for each class.

Hypotheses. Under the assumption that exactly one plaintext class yields a 2^{-16} correlation with $\langle \lambda_0, \lambda_1 \rangle$, while the other classes all yield a correlation of 0 with this approximation, we expect to observe three things:

First, the plaintext class with the 2^{-16} correlation yields the sample correlation with the largest magnitude. Observe here that it follows from Lemma 2.5 that with $t' = 2^{35.61}$ randomly sampled plaintext-ciphertext pairs per class, a 4-bit advantage is achieved with 0.95 probability when the correlation over the first round is in fact 2^{-16} for the correct class. Given that we consider $16 = 2^4$ classes, a 4-bit advantage is only achieved when the class with the non-zero correlation yields the largest magnitude.

Second, the sample correlation for the other fifteen classes is distributed according to $\mathcal{N}(0, 2^{-35.61})$. Recall here that the sample correlation $\hat{C}$ of an approximation with correlation C based on t uniformly random samples is distributed as $\mathcal{N}(C, \frac{1-C^2}{t})$.

Third, the sample correlation of all classes with the approximation $\langle 0, \lambda_1 \rangle$ is also distributed with $\mathcal{N}(0, 2^{-35.61})$. Since the correlation of this approximation is certainly zero regardless of the class, a result that deviates from this expectation would indicate a bias in the generated samples.

4.4.1 Results

The experiment was performed four times, each with a different master key. In Table 4.2, the randomly generated key, as well as the plaintext class $U^\top k = (u_1^\top k, \ldots, u_{32}^\top k)$ expected to yield a 2^{-16} correlation magnitude are presented for each experiment.

Table 4.2: Experiment settings

experiment	k	$U^\top k$
1	$\texttt{C5E267521462BA1186D4BF25CADE8FFB}_\texttt{x}$	$\texttt{2C3C3B11}_\texttt{x}$
2	$\texttt{CB819B4BED6948C2EA1785CBC6BA052B}_\texttt{x}$	$\texttt{4F53A073}_\texttt{x}$
3	$\texttt{2694CADD22CBDCAD40ED5E0FBBA46C74}_\texttt{x}$	$\texttt{0B4329B3}_\texttt{x}$
4	$\texttt{00691AE8AA5C63239D980B645012A2DE}_\texttt{x}$	$\texttt{351A608C}_\texttt{x}$

The sample correlations for the sixteen classes are presented in Table 4.3, where the classes are ordered based on the magnitude of the sample correlation. The code used to compute these sample correlations can be found in Appendix E. Cross-referencing with Table 4.2, we conclude that the greatest correlation magnitude is always obtained by the class which we expect to yield the 2^{-16} correlation, confirming the first hypothesis. When aggregating the four experiments, these suggest that the normal distribution $\mathcal{N}(0, 2^{-35.61})$ is not a perfect fit for the sampled correlations of the sixty zero-correlation classes. In particular, we find that 76.7% of sample correlations lie within one standard deviation of zero, 96.7% within two, and two extreme samples lie 3.06 and 3.14 standard deviations away from the mean. Although the sample size is small, these results suggest that the actual distribution has a smaller variation than hypothesized. We do currently have no hypothesis explaining why this distribution could be narrower than expected. Lastly, we note that $\mathcal{N}(0, 2^{-35.61})$ is a good fit for the sample correlations of all classes with $\langle 0, \lambda_1 \rangle$. Here, 70.3% of samples lie within one standard deviation from the mean, 95.3% within two, 98.4% within three, and there is one outlier at 3.0001 standard deviations away from 0. This result underpins the claim that the samples used in this experiment have been generated uniformly at random.

Table 4.3: Correlations per plaintext class

(a) Correlations experiment #1

b	$C^{AES_k^4}_{\lambda_1,\lambda_0}$	$C^{AES_k^4}_{\lambda_1,0}$
$\texttt{2C3C3B11}_\texttt{x}$	$2^{-15.44}$	$-2^{-17.84}$
$\texttt{9A567E78}_\texttt{x}$	$2^{-17.20}$	$2^{-17.67}$
$\texttt{39CB0CDC}_\texttt{x}$	$2^{-17.44}$	$-2^{-18.56}$
$\texttt{5EA0E538}_\texttt{x}$	$2^{-17.60}$	$-2^{-19.52}$
$\texttt{E5F7DE38}_\texttt{x}$	$2^{-17.75}$	$-2^{-17.12}$
$\texttt{20321547}_\texttt{x}$	$-2^{-18.06}$	$2^{-18.81}$
$\texttt{9A01EEAB}_\texttt{x}$	$2^{-18.37}$	$2^{-19.08}$
$\texttt{1D649732}_\texttt{x}$	$-2^{-18.49}$	$2^{-17.04}$
$\texttt{9E5600E7}_\texttt{x}$	$-2^{-18.98}$	$2^{-18.35}$
$\texttt{FBAEF9E9}_\texttt{x}$	$-2^{-19.32}$	$2^{-18.60}$
$\texttt{C0C102B4}_\texttt{x}$	$2^{-20.12}$	$2^{-17.53}$
$\texttt{D5A78172}_\texttt{x}$	$2^{-20.47}$	$2^{-20.12}$
$\texttt{BFC6AB67}_\texttt{x}$	$2^{-20.56}$	$2^{-18.05}$
$\texttt{89FB23BF}_\texttt{x}$	$2^{-20.78}$	$-2^{-17.90}$
$\texttt{F0488353}_\texttt{x}$	$-2^{-20.90}$	$-2^{-16.57}$
$\texttt{82138799}_\texttt{x}$	$2^{-24.39}$	$2^{-17.33}$

(b) Correlations experiment #2

b	$C^{AES_k^4}_{\lambda_1,\lambda_0}$	$C^{AES_k^4}_{\lambda_1,0}$
$\texttt{4F53A073}_\texttt{x}$	$2^{-16.15}$	$2^{-18.92}$
$\texttt{87C1ABB1}_\texttt{x}$	$2^{-16.19}$	$-2^{-18.33}$
$\texttt{B6D4C9B5}_\texttt{x}$	$-2^{-17.13}$	$2^{-18.38}$
$\texttt{223FC669}_\texttt{x}$	$-2^{-17.21}$	$-2^{-23.53}$
$\texttt{944CD767}_\texttt{x}$	$2^{-17.26}$	$2^{-18.19}$
$\texttt{96E591F0}_\texttt{x}$	$2^{-17.43}$	$2^{-18.42}$
$\texttt{5189FEDD}_\texttt{x}$	$2^{-18.29}$	$2^{-16.40}$
$\texttt{3216543A}_\texttt{x}$	$-2^{-19.03}$	$2^{-17.95}$
$\texttt{1B3AF07B}_\texttt{x}$	$2^{-19.38}$	$2^{-17.54}$
$\texttt{166FF615}_\texttt{x}$	$-2^{-19.75}$	$-2^{-19.49}$
$\texttt{BCF41D3A}_\texttt{x}$	$-2^{-19.74}$	$-2^{-17.41}$
$\texttt{3FBD312A}_\texttt{x}$	$-2^{-19.74}$	$-2^{-20.13}$
$\texttt{0441364F}_\texttt{x}$	$2^{-20.29}$	$-2^{-17.40}$
$\texttt{4A6B30C1}_\texttt{x}$	$-2^{-20.21}$	$2^{-20.99}$
$\texttt{E7E50D8F}_\texttt{x}$	$-2^{-20.32}$	$2^{-21.43}$
$\texttt{9ABF491B}_\texttt{x}$	$-2^{-20.04}$	$2^{-17.51}$

(c) Correlations experiment #3

b	$C^{AES_k^4}_{\lambda_1,\lambda_0}$	$C^{AES_k^4}_{\lambda_1,0}$
$\texttt{0B4329B3}_\texttt{x}$	$-2^{-16.10}$	$2^{-21.49}$
$\texttt{3E2F272A}_\texttt{x}$	$2^{-17.84}$	$-2^{-19.97}$
$\texttt{1A2D9E5C}_\texttt{x}$	$-2^{-17.84}$	$2^{-21.61}$
$\texttt{7D0578B9}_\texttt{x}$	$-2^{-17.93}$	$2^{-21.31}$
$\texttt{A0A3B4EF}_\texttt{x}$	$2^{-18.20}$	$2^{-19.70}$
$\texttt{D05E9882}_\texttt{x}$	$2^{-18.39}$	$-2^{-16.87}$
$\texttt{EEF183BE}_\texttt{x}$	$2^{-18.44}$	$2^{-18.03}$
$\texttt{985254A6}_\texttt{x}$	$-2^{-18.46}$	$2^{-18.34}$
$\texttt{EB685829}_\texttt{x}$	$2^{-18.69}$	$2^{-17.68}$
$\texttt{4F33AC99}_\texttt{x}$	$-2^{-18.76}$	$-2^{-19.29}$
$\texttt{4B400627}_\texttt{x}$	$2^{-19.39}$	$-2^{-17.89}$
$\texttt{3C115FEA}_\texttt{x}$	$2^{-20.54}$	$-2^{-17.47}$
$\texttt{513A6C22}_\texttt{x}$	$-2^{-20.77}$	$-2^{-20.42}$
$\texttt{BB9DB8E6}_\texttt{x}$	$-2^{-21.12}$	$-2^{-18.92}$
$\texttt{0CFB97BD}_\texttt{x}$	$2^{-21.36}$	$-2^{-21.92}$
$\texttt{FB3199D4}_\texttt{x}$	$-2^{-23.45}$	$-2^{-18.69}$

(d) Correlations experiment #4

b	$C^{AES_k^4}_{\lambda_1,\lambda_0}$	$C^{AES_k^4}_{\lambda_1,0}$
$\texttt{351A608C}_\texttt{x}$	$-2^{-15.63}$	$-2^{-19.66}$
$\texttt{A12E7C96}_\texttt{x}$	$-2^{-16.15}$	$-2^{-19.11}$
$\texttt{470609B6}_\texttt{x}$	$-2^{-17.26}$	$-2^{-17.20}$
$\texttt{AED80607}_\texttt{x}$	$-2^{-17.33}$	$2^{-17.84}$
$\texttt{DEFA6672}_\texttt{x}$	$2^{-17.35}$	$2^{-18.28}$
$\texttt{BD8E791F}_\texttt{x}$	$2^{-17.53}$	$-2^{-16.22}$
$\texttt{B7F12015}_\texttt{x}$	$2^{-17.85}$	$2^{-20.78}$
$\texttt{1122B235}_\texttt{x}$	$2^{-17.86}$	$2^{-17.29}$
$\texttt{F74985F1}_\texttt{x}$	$-2^{-18.09}$	$2^{-19.43}$
$\texttt{250C1472}_\texttt{x}$	$-2^{-18.16}$	$-2^{-18.10}$
$\texttt{36153352}_\texttt{x}$	$2^{-18.34}$	$2^{-18.33}$
$\texttt{53DA8166}_\texttt{x}$	$2^{-18.84}$	$2^{-19.47}$
$\texttt{EB9B1701}_\texttt{x}$	$-2^{-19.33}$	$-2^{-17.00}$
$\texttt{F87F0D4C}_\texttt{x}$	$-2^{-19.44}$	$2^{-17.19}$
$\texttt{F6F3652F}_\texttt{x}$	$2^{-19.75}$	$2^{-18.21}$
$\texttt{7838A23F}_\texttt{x}$	$-2^{-22.08}$	$2^{-18.22}$

Chapter 5

Conclusion

In this thesis we have observed that `inv`, the inversion function on 2^8 elements, has five input masks ω for every output mask $v \in \mathbb{F}_2^8 \setminus \{0\}$ s.t. $C_{v,\omega}^{\texttt{inv}} = 2^{-3}$. Given this fact, we designed the weighted and conditional approximation techniques which yield a correlation of 2^{-2} and 2^{-1}, respectively, when applied to `inv`. As such, both techniques outperform any linear approximation for the `inv` function. Of the two, the conditional approximation was used to construct a statistical distinguisher for four-round AES in the known-plaintext model. Previously, the existence of such a distinguisher for this cipher was argued to be impossible. In addition to using this distinguisher to construct a distinguishing attack for $\texttt{AES}_k^4$, we have moreover demonstrated an key-recovery attack capable of extracting 32 bits of information on the key using only $2^{125.62}$ data. Lastly, a small-scale experiment was performed, which validated that the conditional approximation can be used to achieve a 4-bit advantage when approximating one-round AES with sixteen active s-boxes and sampling $2^{35.61}$ data for all sixteen plaintext classes.

With the distinguisher established and validated, this thesis demonstrates that four-round AES is distinguishable from a random permutation in the known-plaintext model. In addition to illustrating that this round-reduced version is unsafe, this result furthermore presents a potential weakness in any cipher using it as a subroutine. Even more impacting is the fact that the vulnerability in four-round AES stems from a caveat in the security argument of the Wide Trail Strategy. This security argument implicitly assumed that the correlation magnitude of an approximation for the s-box is upper bounded by 2^{-3}. As such, the conditional approximation forms a threat to the security of any cipher constructed according to this design framework.

5.1 Future Work

Research is a never-ending adventure, as is the case here. In addition to some issues that arose, a large number of ideas was pruned during the formation of this thesis. We briefly present seven topics we consider most worth further investigation.

WTS-based ciphers. Since this work has revealed a caveat for a security argument of the Wide Trail Strategy, any other cryptographic objects designed according to this framework might be vulnerable to a conditional attack. As such, the design of these ciphers must be revisited to verify this and patch the design where necessary.

Revisiting the WTS. Additionally, the Wide Trail Strategy itself should be revisited and updated to ensure future ciphers built according to this framework are not vulnerable to conditional attacks.

The zero-correlation classes. In the results, it was observed that the sample correlations of the "zero-correlation" classes do not perfectly fit the expected distribution. Further investigation to verify whether this anomaly is persistent under large data sets is necessary. When the anomaly persists, it is worth investigating where this anomaly originates from.

First and last round. In Section 4.2 the decision was made to restrict ourselves to only use the conditional approximation for the s-boxes in the first round. The reason underlying this decision relates to Partitioning Cryptanalysis [24]. By conditioning the plaintext-ciphertext pairs on both the plaintext as well as the ciphertext, it

is expected that the classes no longer have equal sizes. It is unclear at this point whether and how this were to influence the presented attacks. Given that it could improve the correlation of the linear trail, we are very curious whether the conditional approximation could also be used for the s-boxes in the last round. Perhaps conditioning and partitioning attacks could be united here.

Inner rounds. In Section 3.3 we only considered extending the weighted and conditional approximations when the function of interest was combined with an affine operation. Further research in extending these approximations for combinations with other non-linear approximations should be performed, as this could potentially allow s-boxes in inner rounds to be approximated using this technique as well.

Additional conditioning masks. During the construction of the conditional approximation for `inv` in Section 3.2.2, two conditioning masks were ultimately used. Preliminary experimental results suggest that for any output mask $v \in \mathbb{F}_2^8 \setminus \{0\}$, there exists a third mask which, when added to the set of conditioning masks, leads to the formation of eight plaintext classes, where for on one a $2^{-1/2}$ correlation is achieved. Since some of the other classes in this partition also yield a non-zero correlation, Lemma 2.5 can no longer be used to compute the data complexity for attacks leveraging this approximation. Still, we reckon that a more careful analysis of the sampled conditional correlations of the plaintext classes could allow for a key-attack that fits within the full code book and recovers 48 bits of the key.

The weighted approximation. Recall the discussion on extending the weighted approximation when the function under investigation is prepended with an affine translation. It was mentioned here that no method was found to adapt the weights when the translation vector is unknown. Given that its 2^{-2} correlation is very promising, further research into this technique should be performed.

Bibliography

[1] Alon, N., Matias, Y., and Szegedy, M. The Space Complexity of Approximating the Frequency Moments. *Journal of Computer and System Sciences 58*, 1 (1999), 137–147.

[2] Ashur, T., Beyne, T., and Rijmen, V. Revisiting the Wrong-Key-Randomization Hypothesis. *Journal of Cryptology 33*, 2 (Apr 2020), 567–594.

[3] Ashur, T., Khan, M., and Nyberg, K. Structural and Statistical Analysis of Multidimensional Linear Approximations of Random Functions and Permutations. *IEEE Transactions on Information Theory 68* (2022), 1296–1315.

[4] Ashur, T., and Posteuca, R. On linear hulls in one round of DES. *IACR Cryptol. ePrint Arch.* (2018), 635.

[5] Biham, E. On Matsui's linear cryptanalysis. In *Advances in Cryptology — EUROCRYPT'94* (Berlin, Heidelberg, 1995), A. De Santis, Ed., Springer Berlin Heidelberg, pp. 341–355.

[6] Biham, E., and Perle, S. Conditional Linear Cryptanalysis – Cryptanalysis of DES with Less Than 2^{42} Complexity. *IACR Transactions on Symmetric Cryptology 2018*, 3 (Sep. 2018), 215–264.

[7] Bilgin, B., Bogdanov, A., Knežević, M., Mendel, F., and Wang, Q. Fides: Lightweight authenticated cipher with side-channel resistance for constrained hardware. In *Cryptographic Hardware and Embedded Systems - CHES 2013* (Berlin, Heidelberg, 2013), G. Bertoni and J.-S. Coron, Eds., Springer Berlin Heidelberg, pp. 142–158.

[8] Biryukov, A., De Cannière, C., and Quisquater, M. On Multiple Linear Approximations. In *Advances in Cryptology – CRYPTO 2004* (Berlin, Heidelberg, 2004), M. Franklin, Ed., Springer Berlin Heidelberg, pp. 1–22.

[9] Bogdanov, A., Leander, G., Nyberg, K., and Wang, M. Integral and multidimensional linear distinguishers with correlation zero. In *Advances in Cryptology – ASIACRYPT 2012* (Berlin, Heidelberg, 2012), X. Wang and K. Sako, Eds., Springer Berlin Heidelberg, pp. 244–261.

[10] Bogdanov, A., and Tischhauser, E. On the Wrong Key Randomisation and Key Equivalence Hypotheses in Matsui's Algorithm 2. In *Fast Software Encryption* (Berlin, Heidelberg, 2014), S. Moriai, Ed., Springer Berlin Heidelberg, pp. 19–38.

[11] Daemen, J. *Cipher and hash function design, strategies based on linear and differential cryptanalysis, PhD Thesis.* K.U.Leuven, 1995. `http://jda.noekeon.org/`.

[12] Daemen, J., Govaerts, R., and Vandewalle, J. Correlation matrices. In *Fast Software Encryption* (Berlin, Heidelberg, 1995), B. Preneel, Ed., Springer Berlin Heidelberg, pp. 275–285.

[13] Daemen, J., Knudsen, L., and Rijmen, V. The block cipher Square. In *Fast Software Encryption* (Berlin, Heidelberg, 1997), E. Biham, Ed., Springer Berlin Heidelberg, pp. 149–165.

[14] Daemen, J., and Rijmen, V. The Block Cipher BKSQ. In *Smart Card Research and Applications* (Berlin, Heidelberg, 2000), J.-J. Quisquater and B. Schneier, Eds., Springer Berlin Heidelberg, pp. 236–245.

[15] Daemen, J., and Rijmen, V. The Wide Trail Design Strategy. In *Cryptography and Coding* (Berlin, Heidelberg, 2001), B. Honary, Ed., Springer Berlin Heidelberg, pp. 222–238.

[16] Daemen, J., and Rijmen, V. *Correlation Matrices.* Springer Berlin Heidelberg, Berlin, Heidelberg, 2020, pp. 91–113.

[17] Daemen, J., and Rijmen, V. *The Design of Rijndael*, 2 ed. Springer, 2020.

[18] Dworkin, M., Barker, E., Nechvatal, J., Foti, J., Bassham, L., Roback, E., and Dray, J. *Advanced Encryption Standard (AES)*, November 2001.

[19] Edgar, T. W., and Manz, D. O. Chapter 2 - Science and Cyber Security. In *Research Methods for Cyber Security*, T. W. Edgar and D. O. Manz, Eds. Syngress, 2017, pp. 33–62.

[20] Feistel, H. Cryptography and Computer Privacy. *Scientific American 228*, 5 (1973), 15–23.

[21] Fielding, R., Nottingham, M., and Reschke, J. "HTTP Semantics", STD 97, RFC 9110. `https://www.rfc-editor.org/info/rfc9110`, June 2022.

[22] Guo, J., Peyrin, T., and Poschmann, A. The photon family of lightweight hash functions. In *Advances in Cryptology – CRYPTO 2011* (Berlin, Heidelberg, 2011), P. Rogaway, Ed., Springer Berlin Heidelberg, pp. 222–239.

[23] Guo, J., Peyrin, T., Poschmann, A., and Robshaw, M. The LED Block Cipher. In *Cryptographic Hardware and Embedded Systems – CHES 2011* (Berlin, Heidelberg, 2011), B. Preneel and T. Takagi, Eds., Springer Berlin Heidelberg, pp. 326–341.

[24] Harpes, C., and Massey, J. L. Partitioning cryptanalysis. In *Fast Software Encryption* (Berlin, Heidelberg, 1997), E. Biham, Ed., Springer Berlin Heidelberg, pp. 13–27.

[25] Hellman, M. A cryptanalytic time-memory trade-off. *IEEE Transactions on Information Theory 26*, 4 (1980), 401–406.

[26] Hermelin, M., Cho, J. Y., and Nyberg, K. Multidimensional Linear Cryptanalysis. *Journal of Cryptology 32*, 1 (Jan 2019), 1–34.

[27] Hermelin, M., and Nyberg, K. Linear Cryptanalysis Using Multiple Linear Approximations. *IACR Cryptol. ePrint Arch.* (2011), 93.

[28] Kaliski, B. S., and Robshaw, M. J. B. Linear Cryptanalysis Using Multiple Approximations. In *Advances in Cryptology — CRYPTO '94* (Berlin, Heidelberg, 1994), Y. G. Desmedt, Ed., Springer Berlin Heidelberg, pp. 26–39.

[29] Keliher, L., Meijer, H., and Tavares, S. Improving the Upper Bound on the Maximum Average Linear Hull Probability for Rijndael. In *Selected Areas in Cryptography* (Berlin, Heidelberg, 2001), S. Vaudenay and A. M. Youssef, Eds., Springer Berlin Heidelberg, pp. 112–128.

[30] Keliher, L., Meijer, H., and Tavares, S. E. New Method for Upper Bounding the Maximum Average Linear Hull Probability for SPNs. In *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding* (2001), B. Pfitzmann, Ed., vol. 2045 of *Lecture Notes in Computer Science*, Springer, pp. 420–436.

[31] Knudsen, L. R. Contemporary block ciphers. In *School organized by the European Educational Forum* (1998), Springer, pp. 105–126.

[32] Lai, X., Massey, J. L., and Murphy, S. Markov ciphers and differential cryptanalysis. In *Advances in Cryptology — EUROCRYPT '91* (Berlin, Heidelberg, 1991), D. W. Davies, Ed., Springer Berlin Heidelberg, pp. 17–38.

[33] Leander, G. Small Scale Variants Of The Block Cipher PRESENT. Cryptology ePrint Archive, Paper 2010/143, 2010. `https://eprint.iacr.org/2010/143`.

[34] Lidl, R., and Niederreiter, H. *Introduction to Finite Fields and their Applications*, 2 ed. Cambridge University Press, 1994.

[35] Matsui, M. Linear Cryptanalysis Method for DES Cipher. In *Advances in Cryptology — EUROCRYPT '93* (Berlin, Heidelberg, 1993), T. Helleseth, Ed., Springer Berlin Heidelberg, pp. 386–397.

[36] Matsui, M. On correlation between the order of S-boxes and the strength of DES. In *Advances in Cryptology — EUROCRYPT'94* (Berlin, Heidelberg, 1995), A. De Santis, Ed., Springer Berlin Heidelberg, pp. 366–375.

[37] Matsui, M., and Yamagishi, A. A New Method for Known Plaintext Attack of FEAL Cipher. In *Advances in Cryptology - EUROCRYPT '92, Workshop on the Theory and Application of of Cryptographic Techniques, Balatonfüred, Hungary, May 24-28, 1992, Proceedings* (1992), vol. 658 of *Lecture Notes in Computer Science*, Springer, pp. 81–91.

[38] Miyaguchi, S. The FEAL Cipher Family. In *Advances in Cryptology-CRYPTO' 90* (Berlin, Heidelberg, 1991), A. J. Menezes and S. A. Vanstone, Eds., Springer Berlin Heidelberg, pp. 628–638.

[39] Mouha, N., and Dworkin, M. Review of the Advanced Encryption Standard. `https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=932413`, July 2021.

[40] Murphy, S. The Independence of Linear Approximations in Symmetric Cryptanalysis. *IEEE Transactions on Information Theory 52* (2006), 5510–5518.

[41] Murphy, S. The effectiveness of the linear hull effect. *Journal of Mathematical Cryptology 6*, 2 (2012), 137–147.

[42] National Institute of Standards and Technology (NIST). Data Encryption Standard (DES). `https://csrc.nist.gov/csrc/media/publications/fips/46/3/archive/1999-10-25/documents/fips46-3.pdf`, October 1999.

[43] National Institute of Standards and Technology (NIST). AES Development. `https://csrc.nist.gov/projects/cryptographic-standards-and-guidelines/archived-crypto-projects/aes-development`, 2021.

[44] Newman, D. J. The Double Dixie Cup Problem. *The American Mathematical Monthly 67*, 1 (1960), 58–61.

[45] Nyberg, K. Differentially uniform mappings for Cryptography. In *Advances in Cryptology — EUROCRYPT '93* (Berlin, Heidelberg, 1994), T. Helleseth, Ed., Springer Berlin Heidelberg, pp. 55–64.

[46] Nyberg, K. Linear approximation of block ciphers. In *Advances in Cryptology — EUROCRYPT'94* (Berlin, Heidelberg, 1995), A. De Santis, Ed., Springer Berlin Heidelberg, pp. 439–444.

[47] Nyberg, K. Affine Linear Cryptanalysis. *Cryptography and Communications 11*, 3 (May 2019), 367–377.

[48] O'Neill, M. E. PCG: A family of simple fast space-efficient statistically good algorithms for random number generation. *ACM Transactions on Mathematical Software* (2014).

[49] O'Neill, M. E. PCG, A Family of Better Random Number Generators. `https://www.pcg-random.org/`, 2018.

[50] Rijmen, V., Daemen, J., Preneel, B., Bosselaers, A., and De Win, E. The cipher SHARK. In *Fast Software Encryption* (Berlin, Heidelberg, 1996), D. Gollmann, Ed., Springer Berlin Heidelberg, pp. 99–111.

[51] Rivest, R. L., Robshaw, M. J., Sidney, R., and Yin, Y. L. The RC6 block cipher. In *First advanced encryption standard (AES) conference* (1998), p. 16.

[52] Röck, A., and Nyberg, K. Generalization of Matsui's Algorithm 1 to linear hull for key-alternating block ciphers. *Designs, codes and cryptography 66*, 1 (2013), 175–193.

[53] Rogaway, P. The security of DESX. *RSA Laboratories Cryptobytes 2*, 2 (1996).

[54] Selçuk, A. A. On Probability of Success in Linear and Differential Cryptanalysis. *Journal of Cryptology 21*, 1 (Jan 2008), 131–147.

[55] Shannon, C. E. Communication theory of secrecy systems. *The Bell System Technical Journal 28*, 4 (1949), 656–715.

[56] Siil, K. A. An Introduction to Cryptanalysis. *AT&T Technical Journal 73*, 5 (1994), 24–29.

Appendix A

Proofs

This appendix contains the proofs of some of the lemmas presented in the thesis. The proofs are presented in the same order as the lemmas are introduced.

Lemma 2.1 (Balanced function)**.** *Let $n \in \mathbb{N}_+$ and $a \in \mathbb{F}_2^n$ be arbitrary. It holds that $Imb(a^\top x) = \delta(a) \cdot \frac{|\mathbb{F}_2^n|}{2}$.*

Proof. We will prove the statement by induction on n. Let $n = 1$ and $a \in \mathbb{F}_2^n$ arbitrary. Then

$$Imb(a^\top x) = \frac{1}{2}\sum_{x\in\mathbb{F}_2}(-1)^{a^\top x} = \frac{1}{2}\big((-1)^{a\wedge 0} + (-1)^{a\wedge 1}\big) = \frac{1}{2}(1 + (-1)^a) = \frac{1}{2}\cdot 2\delta(a) = \delta(a)\cdot\frac{|\mathbb{F}_2|}{2}.$$

Next, we demonstrate that if the lemma holds for some $n = k \geq 1$, it still holds for $n = k + 1$. Let $a \in \mathbb{F}_2^{k+1}$ arbitrary and define $\overline{a} := (a_1, \ldots, a_k) \in \mathbb{F}_2^k$ and $a' := a_{k+1} \in \mathbb{F}_2$. Then

$$\begin{aligned}
Imb(a^\top x) &= \frac{1}{2}\sum_{x\in\mathbb{F}_2^{k+1}}(-1)^{a^\top x} \\
&= \frac{1}{2}\sum_{x\in\mathbb{F}_2^k}\sum_{y\in\mathbb{F}_2}(-1)^{\overline{a}^\top x\oplus a'\wedge y} \\
&= \frac{1}{2}\left(\sum_{x\in\mathbb{F}_2^k}(-1)^{\overline{a}^\top x}\right)\left(\sum_{y\in\mathbb{F}_2}(-1)^{a'\wedge y}\right) \\
&= \frac{1}{2}\cdot\delta(\overline{a})\cdot|\mathbb{F}_2^n|\cdot\delta(a')\cdot|\mathbb{F}_2| \\
&= \delta(a)\cdot\frac{|\mathbb{F}_2^{k+1}|}{2}.
\end{aligned}$$

The lemma follows by induction on n. □

Lemma 2.2. *Let $f, g : \mathbb{F}_2^n \to \mathbb{F}_2$ be arbitrary Boolean functions. The correlation $C(f, g)$ of these two functions can be computed as*

$$C(f,g) = \frac{1}{|\mathbb{F}_2^n|}\sum_{x\in\mathbb{F}_2^n}(-1)^{f(x)\oplus g(x)}. \tag{2.3}$$

Proof. Observe that $|\mathbb{F}_2^n| = 2^n$. Then

$$\begin{aligned}
C(f,g) &= 2 \cdot \mathbb{P}[f(x) = g(x)] - 1 \\
&= 2 \cdot \mathbb{P}[f(x) \oplus g(x) = 0] - 1 \\
&= 2 \cdot 2^{-n} \cdot |\{x \in \mathbb{F}_2^n \mid f(x) \oplus g(x) = 0\}| - 1 \\
&= 2 \cdot 2^{-n} \cdot \left|\left\{x \in \mathbb{F}_2^n \;\middle|\; (-1)^{f(x)\oplus g(x)} = 1\right\}\right| - 1 \\
&= 2 \cdot 2^{-n} \cdot \left|\left\{x \in \mathbb{F}_2^n \;\middle|\; \frac{1}{2} \cdot \left((-1)^{f(x)\oplus g(x)} + 1\right) = 1\right\}\right| - 1 \\
&= 2 \cdot 2^{-n} \cdot \left(\sum_{x \in \mathbb{F}_2^n} \frac{1}{2} \cdot \left((-1)^{f(x)\oplus g(x)} + 1\right)\right) - 1 \\
&= 2^{-n} \cdot \left(\sum_{x \in \mathbb{F}_2^n} \left((-1)^{f(x)\oplus g(x)} + 1\right)\right) - 1 \\
&= 2^{-n} \cdot \left(2^n + \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x)\oplus g(x)}\right) - 1 \\
&= \frac{1}{|\mathbb{F}_2^n|} \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x)\oplus g(x)}.
\end{aligned}$$

□

Lemma 2.4 (Lemma 2, [35])**.** *Let t be the number of given random plaintext-ciphertext pairs and $|C_\lambda^F|$ be the absolute correlation contribution of the trail λ that dominates $\langle u, v\rangle$. Given that $|C_\lambda^F|$ is sufficiently small, the success rate p^* of Algorithm 1 is*

$$p^* = \int_{-2\sqrt{t}\cdot|C_\lambda^F|}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-x^2/2} dx = \Phi\Big(|C_\lambda^F| \cdot \sqrt{t}\Big), \tag{2.16}$$

where Φ denotes the cumulative distribution function of the standard normal distribution.

Proof. Observe that Algorithm 1 is a decision problem at its core: it is up to the adversary to establish whether $C_{v,u}^{E_k} = C_\lambda^F$ or $C_{v,u}^{E_k} = -C_\lambda^F$ and use the sample correlation $\hat{C} \approx C_{v,u}^{E_k}$ to decide this. Given that $C_{v,u}^{E_k} \approx C_\lambda^k = (-1)^{\lambda^\top k} \cdot \mathtt{sgn}(C_\lambda^F) \cdot |C_\lambda^F|$, the value of $b := \lambda^\top k$ can be derived when this decision is made. Let us start with deriving the statistical distribution of the sample correlation $\hat{C}$. We present this derivation for a general situation and then apply it to this specific case.

Distribution of sample correlation. Let T denote a random data multi-set used to determine the sample correlation $\hat{C}$ for some linear characteristic $\langle u, v\rangle$ applied to an n-bit block cipher. Assume the set T contains t plaintext-ciphertext samples $(x, E_k(x))$ with all encryptions performed under the same, unknown key k. We use the random variable $\hat{q}$ to express the number of elements in T for which the characteristic is correct, i.e. $\hat{q} = |\{(x, E_k(x)) \in T \mid u^\top x \oplus v^\top E_k(x) = 0\}|$. Since the samples in T are all drawn uniformly at random from the full plaintext-ciphertext-pair space, $\hat{q}$ is distributed according to $\mathcal{B}(t, p_k)$, where $\mathcal{B}$ denotes the binomial distribution and $p_k = \mathbb{P}\big[u^\top x \oplus v^\top E_k(x) = 0\big]$. Observe that for large enough t this distribution is closely approached by the normal distribution $\mathcal{N}(tp_k, tp_k(1-p_k))$. When we compute the sample probability $\hat{p}$ as $\hat{q}/t$, it follows that this random variable is distributed as $\mathcal{N}(p_k, p_k(1-p_k)/t)$. It then also holds that the sample correlation $\hat{c} = 2\hat{p} - 1$ is distributed as $\hat{c} \sim \mathcal{N}(2p_k - 1, 4p_k(1-p_k)/t) \equiv \mathcal{N}\big(c, (1-c^2)/t\big)$, with $c = 2p_k - 1$.

Applied to this situation, it thus holds that $\hat{C} \sim \mathcal{N}(C_{v,u}^{E_k}, (1 - (C_{v,u}^{E_k})^2)/t)$. Since $|C_{v,u}^{E_k}|$ is assumed static under k, it is best to guess that $b = 0$ when $\mathtt{sgn}(\hat{C}) = \mathtt{sgn}(C_\lambda^F)$ and 1 otherwise. The probability that this guess is correct depends on the overlap between the distributions N_0 and N_1, with $N_i := \mathcal{N}((-1)^i \cdot |C_\lambda^F|, (1 - |C_\lambda^F|^2)/t)$; the smaller the overlap, the greater the chance of guessing correct. Given that C_λ^F is fixed for the trail, the overlap between

these probabilities can only be reduced by increasing the sample size t. Let us now derive the relation between t, the correlation magnitude $|C_\lambda^F|$, and the probability of guessing correct p^*. Observe that we may express p^* as

$$p^* = \mathbb{P}\Big[\mathtt{sgn}(\hat{C}) = \mathtt{sgn}(C_\lambda^F) \;\Big|\; b = 0\Big] \cdot \mathbb{P}[b = 0] + \mathbb{P}\Big[\mathtt{sgn}(\hat{C}) = -\mathtt{sgn}(C_\lambda^F) \;\Big|\; b = 1\Big] \cdot \mathbb{P}[b = 1].$$

Under the assumption that $\mathbb{P}[b = 0] = \mathbb{P}[b = 1] = \frac{1}{2}$, we can use random variables $X \sim N_0$ and $Y \sim N_1$ to express this success probability as

$$p^* = \mathbb{P}[X > 0] \cdot \frac{1}{2} + \mathbb{P}[Y < 0] \cdot \frac{1}{2} = \frac{1}{2}(\mathbb{P}[X > 0] + \mathbb{P}[Y < 0]) = \frac{1}{2}(\mathbb{P}[X > 0] + \mathbb{P}[X > 0]) = \mathbb{P}[X > 0],$$

since X and $-Y$ are identically distributed. When rewriting X to a standard normal random variable, we find that

$$p^* = \mathbb{P}[X > 0] = \mathbb{P}\left[\frac{X - |C_\lambda^F|}{\sqrt{(1 - |C_\lambda^F|^2)/t}} > \frac{-|C_\lambda^F|}{\sqrt{(1 - |C_\lambda^F|^2)/t}}\right] = \Phi\left(\frac{|C_\lambda^F|}{\sqrt{(1 - |C_\lambda^F|^2)/t}}\right) \approx \Phi\Big(|C_\lambda^F| \cdot \sqrt{t}\Big),$$

for sufficiently small $|C_\lambda^F|$. □

Lemma 3.7. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be an arbitrary vectorial Boolean function and let $L : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an arbitrary linear permutation. Moreover, let $U \subseteq \mathbb{F}_2^n$ be arbitrary, with $r := |U|$. For any $v \in \mathbb{F}_2^m$, and $W \in \mathbb{R}^r$ it holds that*

$$C_{v,U',W}^{F \circ L} = C_{v,U,W}^F, \tag{3.14}$$

where $U' := \big\{M_L^\top u_i \;\big|\; u_i \in U\big\}$.

Proof. We first observe that L^{-1} exists since L is a linear permutation. It thus holds that

$$(M_L^\top u)^\top L^{-1}(x) = (M_L^\top u)^\top M_{L^{-1}} x = (M_{L^{-1}}^\top M_L^\top u)^\top x = u^\top x$$

for any $u, x \in \mathbb{F}_2^n$. It can be used to show that

$$\begin{aligned}
(Q_{U',W} \circ L^{-1})(x) &= Q_{U',W}(L^{-1}(x)) \\
&= \sum_{i=1}^{r} w_i \cdot (-1)^{u_i'^\top L^{-1}(x)} \\
&= \sum_{i=1}^{r} w_i \cdot (-1)^{(M_{L^{-1}}^\top u_i)^\top L^{-1}(x)} \\
&= \sum_{i=1}^{r} w_i \cdot (-1)^{u_i^\top x} \\
&= Q_{U,W}(x),
\end{aligned}$$

where $r := |U|$. Note that this implies that $Q_{U',W} \circ L^{-1} \equiv Q_{U,W}$ and thus $M_{U',W} \circ L^{-1} \equiv M_{U,W}$. It then follows that

$$\begin{aligned}
C_{v,U',W}^{F \circ L} &= C(M_{U',W}, v^\top (F \circ L)) \\
&= C(M_{U',W} \circ L^{-1}, v^\top (F \circ L \circ L^{-1})) \\
&= C(M_{U,W}, v^\top F) \\
&= C_{v,U,W}^F.
\end{aligned}$$

□

Lemma 3.11. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be an arbitrary vectorial Boolean function and let $L : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an arbitrary linear permutation. For arbitrary $u \in \mathbb{F}_2^n$, $v \in \mathbb{F}_2^m$ and $U \subseteq \mathbb{F}_2^n$ it holds that*

$$C_{v,u'}^{F \circ L}|_{x \in \mathcal{R}_{U'}^b} = C_{v,u}^F|_{x \in \mathcal{R}_U^b}, \tag{3.19}$$

where $u' := M_L^\top u$, $U' := \{M_L^\top u_i \mid u_i \in U\}$.

Proof. We observe that $u^\top Mx = (M^\top u)^\top x$ for any M. This allows us to show that

$$\begin{aligned}
C^{F\circ L}_{v,u'}\Big|_{x\in\mathcal{R}^b_{U'}} &= \frac{1}{|\mathcal{R}^b_{U'}|}\sum_{x\in\mathcal{R}^b_{U'}}(-1)^{u'^\top x\oplus v^\top F(L(x))}\\
&= \frac{1}{|\mathcal{R}^b_{U'}|}\sum_{x;((M_L^\top u_1)^\top x,\ldots,(M_L^\top u_p)^\top x)=b}(-1)^{(M_L^\top u)^\top x\oplus v^\top F(M_L x)}\\
&= \frac{1}{|\mathcal{R}^b_{U'}|}\sum_{x;(u_1^\top(M_L x),\ldots,u_p^\top(M_L x))=b}(-1)^{u^\top(M_L x)\oplus v^\top F(M_L x)}\\
&= \frac{1}{|\mathcal{R}^b_{U'}|}\sum_{M_{L^{-1}}y;(u_1^\top y,\ldots,u_p^\top y)=b}(-1)^{u^\top y\oplus v^\top F(y)}\\
&\overset{(1)}{=} \frac{1}{|\mathcal{R}^b_{U}|}\sum_{y;(u_1^\top y,\ldots,u_p^\top y)=b}(-1)^{u^\top y\oplus v^\top F(y)}\\
&= \frac{1}{|\mathcal{R}^b_{U}|}\sum_{y\in\mathcal{R}^b_{U}}(-1)^{u^\top y\oplus v^\top F(y)}\\
&= C^F_{v,u}\Big|_{x\in\mathcal{R}^b_U}.
\end{aligned}$$

Observe that step (1) holds because $M_{L^{-1}}$ is a permutation; we only reorder the elements in the summation in this step. □

Appendix B

Look up table AES s-box

The look up table for the `inv` function and s-box S are provided in Table B.1 and B.2, respectively. In these tables, the value for $X^{-1}(\mathtt{xy}_\mathtt{x})$ respectively $S(\mathtt{xy}_\mathtt{x})$ is stated at the intersection of row `x` and column `y`.

Table B.1: Look-up table for `inv`; the inversion function on $\mathbb{F}_{2^8}$.

		y														
		· 0	· 1	· 2	· 3	· 4	· 5	· 6	· 7	· 8	· 9	· A	· B	· C	· D	· E	· F
x	0 ·	00_x	01_x	$8D_x$	$F6_x$	CB_x	52_x	$7B_x$	$D1_x$	$E8_x$	$4F_x$	29_x	$C0_x$	$B0_x$	$E1_x$	$E5_x$	$C7_x$
	1 ·	74_x	$B4_x$	AA_x	$4B_x$	99_x	$2B_x$	60_x	$5F_x$	58_x	$3F_x$	FD_x	CC_x	FF_x	40_x	EE_x	$B2_x$
	2 ·	$3A_x$	$6E_x$	$5A_x$	$F1_x$	55_x	$4D_x$	$A8_x$	$C9_x$	$C1_x$	$0A_x$	98_x	15_x	30_x	44_x	$A2_x$	$C2_x$
	3 ·	$2C_x$	45_x	92_x	$6C_x$	$F3_x$	39_x	66_x	42_x	$F2_x$	35_x	20_x	$6F_x$	77_x	BB_x	59_x	19_x
	4 ·	$1D_x$	FE_x	37_x	67_x	$2D_x$	31_x	$F5_x$	69_x	$A7_x$	64_x	AB_x	13_x	54_x	25_x	$E9_x$	09_x
	5 ·	ED_x	$5C_x$	05_x	CA_x	$4C_x$	24_x	87_x	BF_x	18_x	$3E_x$	22_x	$F0_x$	51_x	EC_x	61_x	17_x
	6 ·	16_x	$5E_x$	AF_x	$D3_x$	49_x	$A6_x$	36_x	43_x	$F4_x$	47_x	91_x	DF_x	33_x	93_x	21_x	$3B_x$
	7 ·	79_x	$B7_x$	97_x	85_x	10_x	$B5_x$	BA_x	$3C_x$	$B6_x$	70_x	$D0_x$	06_x	$A1_x$	FA_x	81_x	82_x
	8 ·	83_x	$7E_x$	$7F_x$	80_x	96_x	73_x	BE_x	56_x	$9B_x$	$9E_x$	95_x	$D9_x$	$F7_x$	02_x	$B9_x$	$A4_x$
	9 ·	DE_x	$6A_x$	32_x	$6D_x$	$D8_x$	$8A_x$	84_x	72_x	$2A_x$	14_x	$9F_x$	88_x	$F9_x$	DC_x	89_x	$9A_x$
	A ·	FB_x	$7C_x$	$2E_x$	$C3_x$	$8F_x$	$B8_x$	65_x	48_x	26_x	$C8_x$	12_x	$4A_x$	CE_x	$E7_x$	$D2_x$	62_x
	B ·	$0C_x$	$E0_x$	$1F_x$	EF_x	11_x	75_x	78_x	71_x	$A5_x$	$8E_x$	76_x	$3D_x$	BD_x	BC_x	86_x	57_x
	C ·	$0B_x$	28_x	$2F_x$	$A3_x$	DA_x	$D4_x$	$E4_x$	$0F_x$	$A9_x$	27_x	53_x	04_x	$1B_x$	FC_x	AC_x	$E6_x$
	D ·	$7A_x$	07_x	AE_x	63_x	$C5_x$	DB_x	$E2_x$	EA_x	94_x	$8B_x$	$C4_x$	$D5_x$	$9D_x$	$F8_x$	90_x	$6B_x$
	E ·	$B1_x$	$0D_x$	$D6_x$	EB_x	$C6_x$	$0E_x$	CF_x	AD_x	08_x	$4E_x$	$D7_x$	$E3_x$	$5D_x$	50_x	$1E_x$	$B3_x$
	F ·	$5B_x$	23_x	38_x	34_x	68_x	46_x	03_x	$8C_x$	DD_x	$9C_x$	$7D_x$	$A0_x$	CD_x	$1A_x$	41_x	$1C_x$

Table B.2: Look-up table for the AES s-box function S

x \ y	· 0	· 1	· 2	· 3	· 4	· 5	· 6	· 7	· 8	· 9	· A	· B	· C	· D	· E	· F
0 ·	63_x	$7C_x$	77_x	$7B_x$	$F2_x$	$6B_x$	$6F_x$	$C5_x$	30_x	01_x	67_x	$2B_x$	FE_x	$D7_x$	AB_x	76_x
1 ·	CA_x	82_x	$C9_x$	$7D_x$	FA_x	59_x	47_x	$F0_x$	AD_x	$D4_x$	$A2_x$	AF_x	$9C_x$	$A4_x$	72_x	$C0_x$
2 ·	$B7_x$	FD_x	93_x	26_x	36_x	$3F_x$	$F7_x$	CC_x	34_x	$A5_x$	$E5_x$	$F1_x$	71_x	$D8_x$	31_x	15_x
3 ·	04_x	$C7_x$	23_x	$C3_x$	18_x	96_x	05_x	$9A_x$	07_x	12_x	80_x	$E2_x$	EB_x	27_x	$B2_x$	75_x
4 ·	09_x	83_x	$2C_x$	$1A_x$	$1B_x$	$6E_x$	$5A_x$	$A0_x$	52_x	$3B_x$	$D6_x$	$B3_x$	29_x	$E3_x$	$2F_x$	84_x
5 ·	53_x	$D1_x$	00_x	ED_x	20_x	FC_x	$B1_x$	$5B_x$	$6A_x$	CB_x	BE_x	39_x	$4A_x$	$4C_x$	58_x	CF_x
6 ·	$D0_x$	EF_x	AA_x	FB_x	43_x	$4D_x$	33_x	85_x	45_x	$F9_x$	02_x	$7F_x$	50_x	$3C_x$	$9F_x$	$A8_x$
7 ·	51_x	$A3_x$	40_x	$8F_x$	92_x	$9D_x$	38_x	$F5_x$	BC_x	$B6_x$	DA_x	21_x	10_x	FF_x	$F3_x$	$D2_x$
8 ·	CD_x	$0C_x$	13_x	EC_x	$5F_x$	97_x	44_x	17_x	$C4_x$	$A7_x$	$7E_x$	$3D_x$	64_x	$5D_x$	19_x	73_x
9 ·	60_x	81_x	$4F_x$	DC_x	22_x	$2A_x$	90_x	88_x	46_x	EE_x	$B8_x$	14_x	DE_x	$5E_x$	$0B_x$	DB_x
A ·	$E0_x$	32_x	$3A_x$	$0A_x$	49_x	06_x	24_x	$5C_x$	$C2_x$	$D3_x$	AC_x	62_x	91_x	95_x	$E4_x$	79_x
B ·	$E7_x$	$C8_x$	37_x	$6D_x$	$8D_x$	$D5_x$	$4E_x$	$A9_x$	$6C_x$	56_x	$F4_x$	EA_x	65_x	$7A_x$	AE_x	08_x
C ·	BA_x	78_x	25_x	$2E_x$	$1C_x$	$A6_x$	$B4_x$	$C6_x$	$E8_x$	DD_x	74_x	$1F_x$	$4B_x$	BD_x	$8B_x$	$8A_x$
D ·	70_x	$3E_x$	$B5_x$	66_x	48_x	03_x	$F6_x$	$0E_x$	61_x	35_x	57_x	$B9_x$	86_x	$C1_x$	$1D_x$	$9E_x$
E ·	$E1_x$	$F8_x$	98_x	11_x	69_x	$D9_x$	$8E_x$	94_x	$9B_x$	$1E_x$	87_x	$E9_x$	CE_x	55_x	28_x	DF_x
F ·	$8C_x$	$A1_x$	89_x	$0D_x$	BF_x	$E6_x$	42_x	68_x	41_x	99_x	$2D_x$	$0F_x$	$B0_x$	54_x	BB_x	16_x

Appendix C

MDS matrix

In Section 2.3.5, the algebraic description of the `MixColumns` function is presented. This function can be decomposed into the application of the function `H` to each column of the state. Since this function is linear, there exists a linear matrix $M_{\mathtt{H}}$ such that $\mathtt{H}(x) = M_{\mathtt{H}}x$ for all $x \in \mathbb{F}_2^{32}$. This matrix is presented in Equation C.1. To its readability, all one-entries in this matrix are printed in bold, while all zero-entries are represented with a dot. Moreover, three horizontal and three vertical lines have been included to aid in understanding the effect of applying this matrix to a four-byte column of the state.

$$M_{\mathtt{H}} = \left[\begin{array}{cccccccc|cccccccc|cccccccc|cccccccc}
\cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot \\
\mathbf{1} & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot \\
\mathbf{1} & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot \\
\mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot \\
\mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} \\
\hline
\mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} \\
\hline
\mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} \\
\hline
\mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot \\
\mathbf{1} & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot \\
\mathbf{1} & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot \\
\mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} \\
\mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \mathbf{1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot
\end{array}\right] \tag{C.1}$$

Appendix D

Masks

Table D.1: The set $\Omega_v = \{\omega_1, \omega_2, \omega_3, \omega_4, \omega_5\}$ for every $v \in \mathbb{F}_2^8 \setminus \{0\}$.

v	ω_1	ω_2	ω_3	ω_4	ω_5	v	ω_1	ω_2	ω_3	ω_4	ω_5	v	ω_1	ω_2	ω_3	ω_4	ω_5
01_x	84_x	63_x	$F3_x$	$E7_x$	77_x	25_x	54_x	$3B_x$	$D9_x$	$6F_x$	$8D_x$	49_x	53_x	35_x	ED_x	66_x	BE_x
02_x	$7C_x$	$A2_x$	$D2_x$	DE_x	AE_x	26_x	43_x	85_x	94_x	$C6_x$	$D7_x$	$4A_x$	AA_x	$1D_x$	EC_x	$B7_x$	46_x
03_x	42_x	31_x	$F9_x$	73_x	BB_x	27_x	CF_x	$C2_x$	21_x	$0D_x$	EE_x	$4B_x$	68_x	34_x	$A8_x$	$5C_x$	$C0_x$
04_x	$3E_x$	69_x	$D1_x$	57_x	EF_x	28_x	$C5_x$	$3A_x$	93_x	FF_x	56_x	$4C_x$	$0B_x$	$A4_x$	$B1_x$	AF_x	BA_x
05_x	$5A_x$	CD_x	FD_x	97_x	$A7_x$	29_x	CD_x	30_x	$B0_x$	FD_x	$7D_x$	$4D_x$	21_x	$C2_x$	CA_x	$E3_x$	EB_x
06_x	$D4_x$	$9A_x$	61_x	$4E_x$	$B5_x$	$2A_x$	$2B_x$	54_x	$D9_x$	$7F_x$	$F2_x$	$4E_x$	$E7_x$	$E1_x$	90_x	06_x	77_x
07_x	$A1_x$	18_x	FC_x	$B9_x$	$5D_x$	$2B_x$	$2A_x$	$6D_x$	51_x	47_x	$7B_x$	$4F_x$	12_x	90_x	$F5_x$	82_x	$E7_x$
08_x	80_x	97_x	DA_x	17_x	$5A_x$	$2C_x$	DA_x	CD_x	30_x	17_x	EA_x	50_x	41_x	18_x	$E4_x$	59_x	$A5_x$
09_x	$9F_x$	AB_x	68_x	34_x	$F7_x$	$2D_x$	$D3_x$	$F1_x$	64_x	22_x	$B7_x$	51_x	$E2_x$	$C9_x$	$9D_x$	$2B_x$	$7F_x$
$0A_x$	AD_x	53_x	CB_x	FE_x	66_x	$2E_x$	65_x	$E1_x$	$F3_x$	84_x	96_x	52_x	66_x	98_x	$D8_x$	FE_x	BE_x
$0B_x$	$4C_x$	89_x	93_x	$C5_x$	DF_x	$2F_x$	$F5_x$	71_x	82_x	84_x	77_x	53_x	49_x	$0A_x$	$9E_x$	43_x	$D7_x$
$0C_x$	AE_x	$A2_x$	$A0_x$	$0C_x$	$0E_x$	30_x	$C2_x$	29_x	$2C_x$	EB_x	EE_x	54_x	25_x	$2A_x$	$7B_x$	$0F_x$	$5E_x$
$0D_x$	EA_x	CD_x	$B0_x$	27_x	$5A_x$	31_x	AB_x	$A8_x$	68_x	03_x	$C3_x$	55_x	95_x	AA_x	79_x	$3F_x$	EC_x
$0E_x$	$D0_x$	DC_x	AE_x	$0C_x$	$7E_x$	32_x	$3F_x$	79_x	$5B_x$	46_x	64_x	56_x	15_x	23_x	28_x	36_x	$3D_x$
$0F_x$	$C9_x$	44_x	54_x	$8D_x$	$9D_x$	33_x	$6F_x$	10_x	$E2_x$	$7F_x$	$8D_x$	57_x	18_x	04_x	$E4_x$	$1C_x$	FC_x
10_x	47_x	33_x	$7B_x$	74_x	$3C_x$	34_x	$8A_x$	09_x	$C1_x$	83_x	$4B_x$	58_x	ED_x	$8B_x$	98_x	66_x	75_x
11_x	40_x	$8B_x$	ED_x	CB_x	AD_x	35_x	$8F_x$	43_x	49_x	CC_x	$C6_x$	59_x	$B8_x$	50_x	$D6_x$	$E8_x$	$6E_x$
12_x	$4F_x$	$9A_x$	FB_x	$D5_x$	$B4_x$	36_x	$3A_x$	$6C_x$	89_x	56_x	$B3_x$	$5A_x$	05_x	08_x	CA_x	$0D_x$	CF_x
13_x	$9E_x$	$1B_x$	43_x	85_x	DD_x	37_x	$C8_x$	38_x	$B2_x$	$F0_x$	$7A_x$	$5B_x$	$E9_x$	32_x	78_x	DB_x	91_x
14_x	$9A_x$	60_x	61_x	FA_x	FB_x	38_x	$F4_x$	$C3_x$	AB_x	37_x	$5F_x$	$5C_x$	$B2_x$	$F0_x$	$F9_x$	42_x	$4B_x$
15_x	56_x	$A9_x$	$B3_x$	FF_x	$E5_x$	39_x	$B9_x$	$F8_x$	$E4_x$	41_x	$5D_x$	$5D_x$	86_x	81_x	39_x	07_x	BF_x
16_x	$A6_x$	$E2_x$	$C9_x$	44_x	$6F_x$	$3A_x$	36_x	28_x	$B1_x$	$1E_x$	87_x	$5E_x$	$F2_x$	$C9_x$	54_x	$3B_x$	$A6_x$
17_x	CA_x	08_x	$2C_x$	$C2_x$	$E6_x$	$3B_x$	62_x	25_x	$3C_x$	47_x	$5E_x$	$5F_x$	$7A_x$	38_x	$C1_x$	42_x	BB_x
18_x	57_x	50_x	$D1_x$	07_x	86_x	$3C_x$	$8D_x$	10_x	$3B_x$	$9D_x$	$B6_x$	60_x	$E1_x$	14_x	96_x	$F5_x$	77_x
19_x	$7F_x$	$F2_x$	$C9_x$	$8D_x$	$B6_x$	$3D_x$	89_x	20_x	56_x	$A9_x$	DF_x	61_x	77_x	71_x	63_x	06_x	14_x
$1A_x$	$1E_x$	99_x	92_x	87_x	$8C_x$	$3E_x$	04_x	$B9_x$	$E0_x$	BD_x	$E4_x$	62_x	$3B_x$	$9D_x$	$D9_x$	$A6_x$	$E2_x$
$1B_x$	75_x	13_x	$D8_x$	66_x	AD_x	$3F_x$	32_x	55_x	91_x	67_x	$A3_x$	63_x	$D5_x$	01_x	61_x	$D4_x$	$B4_x$
$1C_x$	$E8_x$	BF_x	86_x	57_x	$6E_x$	40_x	11_x	$9E_x$	CC_x	$8F_x$	DD_x	64_x	$1F_x$	32_x	$A3_x$	$2D_x$	BC_x
$1D_x$	$C4_x$	$4A_x$	78_x	$8E_x$	BC_x	41_x	50_x	39_x	$D6_x$	69_x	86_x	65_x	$2E_x$	$9B_x$	FA_x	$B5_x$	$D4_x$
$1E_x$	$1A_x$	20_x	$6C_x$	$3A_x$	76_x	42_x	03_x	$5C_x$	68_x	$5F_x$	$6B_x$	66_x	52_x	58_x	49_x	$0A_x$	$1B_x$
$1F_x$	64_x	22_x	AA_x	46_x	CE_x	43_x	26_x	13_x	53_x	35_x	75_x	67_x	$B7_x$	88_x	$F1_x$	$3F_x$	46_x
20_x	23_x	$3D_x$	99_x	$1E_x$	BA_x	44_x	$7B_x$	74_x	$6D_x$	$0F_x$	16_x	68_x	$4B_x$	09_x	31_x	42_x	$7A_x$
21_x	$4D_x$	$6A_x$	EA_x	27_x	$A7_x$	45_x	69_x	$E8_x$	86_x	81_x	EF_x	69_x	45_x	41_x	$E0_x$	04_x	$A5_x$
22_x	$F6_x$	$E9_x$	DB_x	$1F_x$	$2D_x$	46_x	78_x	67_x	32_x	$1F_x$	$4A_x$	$6A_x$	$C7_x$	21_x	$E3_x$	$E6_x$	24_x
23_x	20_x	76_x	$E5_x$	56_x	$C5_x$	47_x	10_x	$3B_x$	$E2_x$	$2B_x$	$F2_x$	$6B_x$	$C1_x$	42_x	$B2_x$	83_x	73_x
24_x	$A7_x$	$6A_x$	DA_x	CD_x	$7D_x$	48_x	$D9_x$	$B6_x$	$A6_x$	$6F_x$	$7F_x$	$6C_x$	BA_x	$A4_x$	$8C_x$	$1E_x$	36_x

v	ω_1	ω_2	ω_3	ω_4	ω_5
$6D_x$	$9D_x$	$2B_x$	$D9_x$	$B6_x$	44_x
$6E_x$	$E4_x$	$F8_x$	59_x	$1C_x$	BD_x
$6F_x$	33_x	25_x	48_x	16_x	$7B_x$
70_x	AC_x	$D0_x$	$D2_x$	$7C_x$	$7E_x$
71_x	FA_x	$D5_x$	61_x	$2F_x$	$9B_x$
72_x	DC_x	72_x	$A0_x$	AE_x	$7C_x$
73_x	$9C_x$	03_x	$6B_x$	$9F_x$	$F7_x$
74_x	$B6_x$	10_x	$F2_x$	$A6_x$	44_x
75_x	$1B_x$	58_x	94_x	43_x	$8F_x$
76_x	$B1_x$	$1E_x$	92_x	AF_x	23_x
77_x	61_x	60_x	$4E_x$	01_x	$2F_x$
78_x	46_x	$5B_x$	88_x	$1D_x$	CE_x
79_x	BC_x	32_x	$E9_x$	$8E_x$	55_x
$7A_x$	$5F_x$	68_x	$9F_x$	37_x	$C0_x$
$7B_x$	44_x	10_x	$2B_x$	54_x	$6F_x$
$7C_x$	02_x	70_x	DC_x	72_x	DE_x
$7D_x$	$E6_x$	$C2_x$	29_x	24_x	CF_x
$7E_x$	DE_x	$D0_x$	70_x	$0E_x$	AE_x
$7F_x$	19_x	$2A_x$	48_x	33_x	51_x
80_x	08_x	CF_x	$E6_x$	$C7_x$	EE_x
81_x	$F8_x$	45_x	$5D_x$	BD_x	$A5_x$
82_x	$B5_x$	$9A_x$	FA_x	$2F_x$	$4F_x$
83_x	$A8_x$	$9C_x$	$6B_x$	34_x	$C3_x$
84_x	01_x	$2E_x$	$B4_x$	$2F_x$	$B5_x$
85_x	98_x	13_x	BE_x	$8B_x$	26_x
86_x	$5D_x$	18_x	41_x	45_x	$1C_x$
87_x	93_x	89_x	$A9_x$	$1A_x$	$3A_x$
88_x	$8E_x$	$E9_x$	78_x	67_x	$F6_x$
89_x	$3D_x$	$0B_x$	BA_x	36_x	87_x
$8A_x$	34_x	$C0_x$	$C3_x$	$F4_x$	$F7_x$
$8B_x$	94_x	11_x	58_x	85_x	CC_x
$8C_x$	FF_x	$1A_x$	93_x	$E5_x$	$6C_x$
$8D_x$	$3C_x$	33_x	19_x	$0F_x$	25_x
$8E_x$	88_x	$1D_x$	79_x	95_x	$F1_x$
$8F_x$	35_x	40_x	$D8_x$	75_x	ED_x
90_x	$9B_x$	$D5_x$	$D4_x$	$4E_x$	$4F_x$
91_x	EC_x	$D3_x$	$5B_x$	$3F_x$	$B7_x$
92_x	$A9_x$	$1A_x$	76_x	$B3_x$	DF_x
93_x	87_x	$0B_x$	28_x	$8C_x$	AF_x
94_x	$8B_x$	AD_x	75_x	26_x	FE_x
95_x	55_x	DB_x	$A3_x$	$8E_x$	$F6_x$
96_x	$B4_x$	$9A_x$	60_x	$2E_x$	$D4_x$
97_x	EB_x	05_x	08_x	EE_x	$E3_x$
98_x	85_x	52_x	58_x	$D7_x$	DD_x
99_x	DF_x	$1A_x$	20_x	$C5_x$	FF_x
$9A_x$	14_x	12_x	82_x	06_x	96_x
$9B_x$	90_x	$F5_x$	71_x	65_x	$E1_x$
$9C_x$	73_x	$F0_x$	$C8_x$	83_x	BB_x
$9D_x$	$6D_x$	62_x	51_x	$0F_x$	$3C_x$
$9E_x$	13_x	40_x	AD_x	53_x	BE_x
$9F_x$	09_x	$7A_x$	$C8_x$	73_x	$C1_x$
$A0_x$	$A0_x$	AC_x	$D2_x$	$0C_x$	72_x
$A1_x$	07_x	$B8_x$	$D1_x$	BF_x	$D6_x$
$A2_x$	$D2_x$	$D0_x$	$0C_x$	02_x	DE_x
$A3_x$	$F1_x$	CE_x	64_x	$3F_x$	95_x
$A4_x$	$B3_x$	$4C_x$	$6C_x$	FF_x	DF_x
$A5_x$	$D1_x$	50_x	$B8_x$	81_x	69_x
$A6_x$	16_x	48_x	62_x	$5E_x$	74_x
$A7_x$	24_x	21_x	EB_x	05_x	CF_x
$A8_x$	83_x	31_x	$C8_x$	$B2_x$	$4B_x$
$A9_x$	92_x	15_x	$3D_x$	87_x	AF_x
AA_x	$4A_x$	55_x	BC_x	$1F_x$	$F6_x$
AB_x	31_x	38_x	$C8_x$	09_x	$F9_x$
AC_x	70_x	$A0_x$	DC_x	$D0_x$	AC_x
AD_x	$0A_x$	11_x	94_x	$1B_x$	$9E_x$
AE_x	$0C_x$	02_x	72_x	$0E_x$	$7E_x$
AF_x	$E5_x$	$A9_x$	93_x	$4C_x$	76_x
$B0_x$	EE_x	$E3_x$	29_x	$0D_x$	$C7_x$
$B1_x$	76_x	$3A_x$	$B3_x$	$4C_x$	$C5_x$
$B2_x$	$5C_x$	$6B_x$	$A8_x$	37_x	$F4_x$
$B3_x$	$A4_x$	$B1_x$	92_x	15_x	36_x
$B4_x$	96_x	12_x	63_x	84_x	$F5_x$
$B5_x$	82_x	84_x	65_x	06_x	$E7_x$
$B6_x$	74_x	19_x	48_x	$6D_x$	$3C_x$
$B7_x$	67_x	$4A_x$	91_x	$2D_x$	$F6_x$
$B8_x$	59_x	$F8_x$	FC_x	$A1_x$	$A5_x$
$B9_x$	39_x	$3E_x$	$D6_x$	07_x	EF_x
BA_x	$6C_x$	20_x	89_x	$4C_x$	$E5_x$
BB_x	$C3_x$	$C0_x$	$9C_x$	03_x	$5F_x$
BC_x	79_x	64_x	AA_x	$1D_x$	$D3_x$
BD_x	BF_x	81_x	$D1_x$	$3E_x$	$6E_x$
BE_x	CC_x	49_x	52_x	85_x	$9E_x$
BF_x	BD_x	$A1_x$	$E0_x$	$1C_x$	$5D_x$
$C0_x$	$F0_x$	$4B_x$	$7A_x$	BB_x	$8A_x$
$C1_x$	$6B_x$	34_x	$F4_x$	$5F_x$	$9F_x$
$C2_x$	30_x	27_x	$7D_x$	17_x	$4D_x$
$C3_x$	BB_x	31_x	38_x	$8A_x$	83_x
$C4_x$	$1D_x$	$D3_x$	$F1_x$	CE_x	EC_x
$C5_x$	28_x	23_x	99_x	$0B_x$	$B1_x$
$C6_x$	FE_x	$D8_x$	CB_x	26_x	35_x
$C7_x$	$6A_x$	80_x	$B0_x$	EA_x	DA_x
$C8_x$	37_x	AB_x	$9F_x$	$9C_x$	$A8_x$
$C9_x$	$0F_x$	19_x	51_x	16_x	$5E_x$
CA_x	17_x	$5A_x$	EA_x	$4D_x$	FD_x
CB_x	$D7_x$	$0A_x$	11_x	DD_x	$C6_x$
CC_x	BE_x	$8B_x$	40_x	35_x	FE_x
CD_x	29_x	$2C_x$	24_x	05_x	$0D_x$
CE_x	DB_x	$C4_x$	78_x	$1F_x$	$A3_x$
CF_x	27_x	$5A_x$	80_x	$7D_x$	$A7_x$
$D0_x$	$0E_x$	70_x	$A2_x$	$7E_x$	AC_x
$D1_x$	$A5_x$	$A1_x$	18_x	04_x	BD_x
$D2_x$	$A2_x$	$A0_x$	70_x	02_x	$D2_x$
$D3_x$	$2D_x$	91_x	$E9_x$	BC_x	$C4_x$
$D4_x$	06_x	63_x	90_x	65_x	96_x
$D5_x$	63_x	71_x	90_x	12_x	$F3_x$
$D6_x$	$E0_x$	41_x	59_x	$A1_x$	$B9_x$
$D7_x$	CB_x	ED_x	98_x	26_x	53_x
$D8_x$	DD_x	$1B_x$	52_x	$C6_x$	$8F_x$
$D9_x$	48_x	$6D_x$	$2A_x$	25_x	62_x
DA_x	$2C_x$	08_x	EB_x	24_x	$C7_x$
DB_x	CE_x	22_x	$5B_x$	EC_x	95_x
DC_x	72_x	$7C_x$	AC_x	$0E_x$	DE_x
DD_x	$D8_x$	13_x	40_x	CB_x	98_x
DE_x	$7E_x$	02_x	$A2_x$	$7C_x$	DC_x
DF_x	99_x	92_x	$A4_x$	$0B_x$	$3D_x$
$E0_x$	$D6_x$	$E8_x$	69_x	$3E_x$	BF_x
$E1_x$	60_x	$4E_x$	$9B_x$	$2E_x$	FB_x
$E2_x$	51_x	16_x	62_x	47_x	33_x
$E3_x$	FD_x	$B0_x$	$6A_x$	$4D_x$	97_x
$E4_x$	$6E_x$	39_x	50_x	57_x	$3E_x$
$E5_x$	AF_x	BA_x	23_x	15_x	$8C_x$
$E6_x$	$7D_x$	$6A_x$	80_x	17_x	FD_x
$E7_x$	$4E_x$	01_x	FB_x	$4F_x$	$B5_x$
$E8_x$	$1C_x$	59_x	$E0_x$	45_x	FC_x
$E9_x$	$5B_x$	79_x	88_x	22_x	$D3_x$
EA_x	$0D_x$	21_x	CA_x	$2C_x$	$C7_x$
EB_x	97_x	30_x	DA_x	$A7_x$	$4D_x$
EC_x	91_x	$4A_x$	$C4_x$	DB_x	55_x
ED_x	58_x	49_x	$D7_x$	11_x	$8F_x$
EE_x	$B0_x$	97_x	30_x	27_x	80_x
EF_x	FC_x	$F8_x$	$B9_x$	04_x	45_x
$F0_x$	$C0_x$	$F7_x$	$9C_x$	37_x	$5C_x$
$F1_x$	$A3_x$	$2D_x$	$C4_x$	$8E_x$	67_x
$F2_x$	$5E_x$	19_x	$2A_x$	47_x	74_x
$F3_x$	FB_x	01_x	$D5_x$	FA_x	$2E_x$
$F4_x$	38_x	$B2_x$	$C1_x$	$8A_x$	$F9_x$
$F5_x$	$2F_x$	60_x	$9B_x$	$4F_x$	$B4_x$
$F6_x$	22_x	88_x	95_x	AA_x	$B7_x$
$F7_x$	$F9_x$	$F0_x$	$8A_x$	09_x	73_x
$F8_x$	81_x	$B8_x$	$6E_x$	39_x	EF_x
$F9_x$	$F7_x$	03_x	AB_x	$F4_x$	$5C_x$
FA_x	71_x	14_x	82_x	65_x	$F3_x$
FB_x	$F3_x$	$E1_x$	14_x	12_x	$E7_x$
FC_x	EF_x	$E8_x$	$B8_x$	07_x	57_x
FD_x	$E3_x$	05_x	29_x	$E6_x$	CA_x
FE_x	$C6_x$	$0A_x$	52_x	CC_x	94_x
FF_x	$8C_x$	99_x	28_x	15_x	$A4_x$

Appendix E

Experiment code

In this appendix we present the code used to run the experiment presented in Section 4.4. For this experiment, the PCG [48] random number generator was used. The source code of this random number generator can be found in [49]. Lastly, the project was compiled using `g++` with the `-O3`, `-flto` and `-fopenmp` flags.

Experiment.cpp

```
#include "math.h"
#include "newToolbox.h"
#include "pcg_random.hpp"

// Test settings
static constexpr uint16_t NREXPERIMENTS = 0x4;
static constexpr uint16_t NRTRIALS = 0x10;
static constexpr uint64_t SAMPLESIZE = 0xC35C80000;
static constexpr State IPM = {0x803CA993, 0xAAA99C80, 0x3CC158A9, 0xC1D9A9AA};
static constexpr State OPM = {0x00000039, 0x00003400, 0x002d0000, 0x2d000000};
static constexpr State W1 = {0x7D6293E5, 0x7993C07D, 0x627A9493, 0x7A9D9379};
static constexpr State W6 = {0xEA19203A, 0xCE206BEA, 0x19F94920, 0xF9F220CE};

// Randomization
pcg64 globalRandom(time(0));


// Compute ID for correct plaintext class
uint32_t getCorrectPlaintextClassID(Key key)
{
    uint32_t plaintextClass = 0;
    for (uint16_t i = 0; i < 16; i++)
    {
        uint32_t w1Parity = P8(key.bytes[i] & W1.bytes[i]);
        plaintextClass ^= w1Parity << (2 * i + 1);

        uint32_t w6Parity = P8(key.bytes[i] & W6.bytes[i]);
        plaintextClass ^= w6Parity << (2 * i);
    }
    return plaintextClass;
}

// Compute W1 parity value, given plaintext class ID and index of plaintext byte
uint8_t getW1parity(uint64_t plaintextClassId, uint16_t byteIdx)
{
    return (plaintextClassId >> (2 * byteIdx + 1)) & 0x01;
}

// Compute W6 parity value, given plaintext class ID and index of plaintext byte
uint8_t getW6parity(uint64_t plaintextClassId, uint16_t byteIdx)
{
    return (plaintextClassId >> (2 * byteIdx)) & 0x01;
}

// Construct all possible values of all bytes in the plaintext, given the plaintext class.
void constructInputBytes(uint8_t *stateBytes, uint32_t plaintextClass)
{
    for (uint8_t byteIdx = 0; byteIdx < 16; byteIdx++)
    {
        // Select w1 and w6 masks
        uint8_t w1 = W1.bytes[byteIdx];
```

```
        uint8_t w6 = W6.bytes[byteIdx];

        // Extract parity bits for w1 and w6
        uint8_t w1p = getW1parity(plaintextClass, byteIdx);
        uint8_t w6p = getW6parity(plaintextClass, byteIdx);

        // Only select bytes with correct parities
        uint64_t parityBitVector = 0;
        uint8_t idx = 0;
        for (uint16_t pt = 0; pt < 256; pt++)
            if (P8(w1 & pt) == w1p && P8(w6 & pt) == w6p)
            {
                stateBytes[64 * byteIdx + idx] = pt;
                idx += 1;
            }
    }
}

// Compute sample correlation for a given plaintext class under a given key
// Computes sample correlation, scaled up by a factor SAMPLESIZE / 2.
// Computes output balance, scaled up by a factor SAMPLESIZE / 2.
void experiment(Key key, uint32_t plaintextClass,
    int64_t &scaledCorrelation, int64_t &scaledBalance)
{
    // Randomizer
    pcg64 random(globalRandom() + plaintextClass);

    // Compute plaintext bytes for each class
    uint8_t stateByteVector[16 * 64];
    constructInputBytes(stateByteVector, plaintextClass);

    scaledCorrelation = (int64_t)(SAMPLESIZE / 2);
    scaledBalance = (int64_t)(SAMPLESIZE / 2);
    for (uint64_t t = 0; t < SAMPLESIZE; t++)
    {
        // Seed random plaintext
        Seed plaintextSeed;
        plaintextSeed.high = random();
        plaintextSeed.low = random();

        // Generate random state (after subbytes) and compute input parity.
        State state;
        for (uint8_t i = 0; i < 16; i++)
        {
            // Select random index in [0, 63]
            uint8_t idx = plaintextSeed.bytes[i] & 0x3F;

            // Select corresponding byte (after subbyte)
            state.bytes[i] = stateByteVector[64 * i + idx];
        }

        // Compute input parity
        uint8_t inputParity = 0;
        for (uint8_t i = 0; i < 4; i++)
            inputParity += __builtin_parityl(IPM.longs[i] & state.longs[i]);
        inputParity &= 1;

        // Encrypt state with one round
        AddRoundKey(state, key, 0);
        SubBytes(state);
        ShiftRows(state);
        MixColumns(state);
        AddRoundKey(state, key, 1);

        // Compute output parity
        uint8_t outputParity = 0;
        for (uint8_t i = 0; i < 4; i++)
            outputParity += __builtin_parityl(OPM.longs[i] & state.longs[i]);
        outputParity &= 1;

        // Update correlation
        scaledCorrelation -= inputParity ^ outputParity;
        scaledBalance -= outputParity;
    }
}


int main()
{
    std::cout << "=====␣CONDITIONAL␣EXPERIMENTS␣=====\n";
    std::cout << "SETTINGS:\n";
    std::cout << "-␣NR␣TRIALS:␣" << NRTRIALS << "\n";
    std::cout << "-␣SAMPLESIZE:␣2^" << std::log2(SAMPLESIZE) << "\n\n";
```

```
    for (uint16_t e = 0; e < NREXPERIMENTS; e++)
    {
        std::cout << "<====␣EXPERIMENT␣" << e << "␣====>\n";

        // Generate master key
        uint32_t master_key[4];
        for (uint16_t i = 0; i < 4; i++)
            master_key[i] = globalRandom();

        // Compute correct plaintext class
        Key expandedKey;
        ExpandKeyTwoRounds(master_key, expandedKey);
        std::cout << "master␣key:\n";
        for (uint8_t i = 0; i < 4; i++)
            printHex(expandedKey.longs[i]);
        uint32_t correctPlaintextClass = getCorrectPlaintextClassID(expandedKey);
        std::cout << "correct␣plaintext␣class:\n";
        printHex(correctPlaintextClass);

        // Generate plaintext classes
        // Note: first plaintext class is the correct class.
        uint32_t classes[NRTRIALS];
        classes[0] = correctPlaintextClass;
        for (uint16_t c = 1; c < NRTRIALS; c++)
            classes[c] = globalRandom();

        std::cout << std::endl;
        // Execute experiment
        # pragma omp parallel for num_threads(8)
        for (uint32_t plaintextClass : classes)
        {
            // Generate encryption key (for each thread)
            Key key;
            ExpandKeyTwoRounds(master_key, key);

            // Run experiment
            int64_t scaledCorrelation = 0;
            int64_t scaledBalance = 0;
            experiment(key, plaintextClass, scaledCorrelation, scaledBalance);

            // Compute correlation
            double correlation = (double) scaledCorrelation / (SAMPLESIZE / 2);
            double corr_exp = std::log2(std::abs(correlation));

            // Compute balance
            double balance = (double) scaledBalance / (SAMPLESIZE / 2);
            double ball_exp = std::log2(std::abs(balance));

            // Print result
            std::cout << "\n<------------->\n";
            std::cout << "plaintext␣class:";
            printHex(plaintextClass);
            std::cout << "scaled␣correlation:␣" << scaledCorrelation << "\n";
            std::cout << "correlation␣mag.:␣␣2^" << corr_exp << "\n";
            std::cout << "scaledBalance:␣" << scaledBalance << "\n";
            std::cout << "balance␣mag.:␣␣2^" << ball_exp << "\n";
            std::cout << "<------------->" << std::endl;
        }
        std::cout << "\n==========================\n\n";
    }
    return 0;
}
```

newToolbox.h

```
#pragma once
#include <stdint.h>

union Seed
{
    struct
    {
        uint64_t low;
        uint64_t high;
    };
    uint8_t bytes[16];
};
static_assert(sizeof(Seed) == sizeof(uint8_t) * 16);

union State
{
    uint32_t longs[4];
    uint8_t bytes[16];
```

```
};
static_assert(sizeof(State) == sizeof(uint32_t) * 4);

union Key
{
    uint32_t longs[8];
    uint8_t bytes[32];
};
static_assert(sizeof(Key) == sizeof(uint32_t) * 8);


// Section-Wise (8 bits) CYclic LEft SHift over by one bit.
#define sw8cylesh32_1(word) (((word & 0x7F7F7F7F) << 1 ^ (word & 0x80808080) >> 7))
#define sw8cylesh64_1(word) ((word & 0x7F7F7F7F7F7F7F7F) << 1 ^ (word & 0x8080808080808080) >> 7)

// CYclic LEft SHift uint32_t's by X bits
#define cylesh32_8(word) (word << 8 ^ word >> 24)
#define cylesh32_16(word) (word << 16 ^ word >> 16)
#define cylesh32_24(word) (word << 24 ^ word >> 8)

// Usefull tools
uint8_t P8(uint8_t v);
uint32_t SubByteCol(uint32_t c);
void printHex(uint32_t w);

// AES key expansion
void ExpandKeyTwoRounds(const uint32_t* masterKey, Key &key);

// AES encryption steps
void AddRoundKey(State &s, Key k, uint8_t round);
void SubBytes(State &s);
void ShiftRows(State &s);
void MixColumns(State &s);
```

newToolbox.cpp

```
#include "newToolbox.h"
#include <iostream>
#include <iomanip>
#include <string>

static const uint8_t sbox[256] = {
    0x63, 0x7C, 0x77, 0x7B, 0xF2, 0x6B, 0x6F, 0xC5,
    0x30, 0x01, 0x67, 0x2B, 0xFE, 0xD7, 0xAB, 0x76,
    0xCA, 0x82, 0xC9, 0x7D, 0xFA, 0x59, 0x47, 0xF0,
    0xAD, 0xD4, 0xA2, 0xAF, 0x9C, 0xA4, 0x72, 0xC0,
    0xB7, 0xFD, 0x93, 0x26, 0x36, 0x3F, 0xF7, 0xCC,
    0x34, 0xA5, 0xE5, 0xF1, 0x71, 0xD8, 0x31, 0x15,
    0x04, 0xC7, 0x23, 0xC3, 0x18, 0x96, 0x05, 0x9A,
    0x07, 0x12, 0x80, 0xE2, 0xEB, 0x27, 0xB2, 0x75,
    0x09, 0x83, 0x2C, 0x1A, 0x1B, 0x6E, 0x5A, 0xA0,
    0x52, 0x3B, 0xD6, 0xB3, 0x29, 0xE3, 0x2F, 0x84,
    0x53, 0xD1, 0x00, 0xED, 0x20, 0xFC, 0xB1, 0x5B,
    0x6A, 0xCB, 0xBE, 0x39, 0x4A, 0x4C, 0x58, 0xCF,
    0xD0, 0xEF, 0xAA, 0xFB, 0x43, 0x4D, 0x33, 0x85,
    0x45, 0xF9, 0x02, 0x7F, 0x50, 0x3C, 0x9F, 0xA8,
    0x51, 0xA3, 0x40, 0x8F, 0x92, 0x9D, 0x38, 0xF5,
    0xBC, 0xB6, 0xDA, 0x21, 0x10, 0xFF, 0xF3, 0xD2,
    0xCD, 0x0C, 0x13, 0xEC, 0x5F, 0x97, 0x44, 0x17,
    0xC4, 0xA7, 0x7E, 0x3D, 0x64, 0x5D, 0x19, 0x73,
    0x60, 0x81, 0x4F, 0xDC, 0x22, 0x2A, 0x90, 0x88,
    0x46, 0xEE, 0xB8, 0x14, 0xDE, 0x5E, 0x0B, 0xDB,
    0xE0, 0x32, 0x3A, 0x0A, 0x49, 0x06, 0x24, 0x5C,
    0xC2, 0xD3, 0xAC, 0x62, 0x91, 0x95, 0xE4, 0x79,
    0xE7, 0xC8, 0x37, 0x6D, 0x8D, 0xD5, 0x4E, 0xA9,
    0x6C, 0x56, 0xF4, 0xEA, 0x65, 0x7A, 0xAE, 0x08,
    0xBA, 0x78, 0x25, 0x2E, 0x1C, 0xA6, 0xB4, 0xC6,
    0xE8, 0xDD, 0x74, 0x1F, 0x4B, 0xBD, 0x8B, 0x8A,
    0x70, 0x3E, 0xB5, 0x66, 0x48, 0x03, 0xF6, 0x0E,
    0x61, 0x35, 0x57, 0xB9, 0x86, 0xC1, 0x1D, 0x9E,
    0xE1, 0xF8, 0x98, 0x11, 0x69, 0xD9, 0x8E, 0x94,
    0x9B, 0x1E, 0x87, 0xE9, 0xCE, 0x55, 0x28, 0xDF,
    0x8C, 0xA1, 0x89, 0x0D, 0xBF, 0xE6, 0x42, 0x68,
    0x41, 0x99, 0x2D, 0x0F, 0xB0, 0x54, 0xBB, 0x16};


void ExpandKeyTwoRounds(const uint32_t *masterKey, Key &key)
{
    for (uint8_t j = 0; j < 4; j++)
        key.longs[j] = masterKey[j];
    key.longs[4] = key.longs[0] ^ cylesh32_8(SubByteCol(key.longs[3])) ^ 0x01;
    for (uint8_t c = 5; c < 8; c++)
        key.longs[c] = key.longs[c - 4] ^ key.longs[c - 1];
```

```
}

void AddRoundKey(State &s, Key key, uint8_t roundIdx)
{
    for (uint16_t i = 0; i < 4; i++)
        s.longs[i] ^= key.longs[4 * roundIdx + i];
}

uint32_t SubByteCol(uint32_t c)
{
    uint8_t *cb = (uint8_t *)&c;
    for (uint8_t i = 0; i < 4; i++)
        cb[i] = sbox[cb[i]];
    return c;
}

void SubBytes(State &s)
{
    for (uint16_t i = 0; i < 16; i++)
        s.bytes[i] = sbox[s.bytes[i]];
}

void ShiftRows(State &s)
{
    uint32_t tmp[4];
    tmp[0] = (s.bytes[15] << 24) ^ (s.bytes[10] << 16) ^ (s.bytes[ 5] << 8) ^ s.bytes[ 0];
    tmp[1] = (s.bytes[ 3] << 24) ^ (s.bytes[14] << 16) ^ (s.bytes[ 9] << 8) ^ s.bytes[ 4];
    tmp[2] = (s.bytes[ 7] << 24) ^ (s.bytes[ 2] << 16) ^ (s.bytes[13] << 8) ^ s.bytes[ 8];
    tmp[3] = (s.bytes[11] << 24) ^ (s.bytes[ 6] << 16) ^ (s.bytes[ 1] << 8) ^ s.bytes[12];
    for (uint8_t i = 0; i < 4; i++)
        s.longs[i] = tmp[i];
}

// Treat all bytes in word as elements in F_{2^n}
// and multiply with 2, mod 0x11A.
uint32_t mulp2(uint32_t a)
{
    uint32_t x = sw8cylesh32_1(a);
    return x ^ ((x & 0x01010101) * 0x1A);
}

uint32_t MixColumn(uint32_t s)
{
    uint32_t tmp = mulp2(s);
    tmp ^= cylesh32_24(tmp);
    tmp ^= cylesh32_24(s);
    tmp ^= cylesh32_16(s);
    tmp ^= cylesh32_8(s);
    return tmp;
}

void MixColumns(State &s)
{
    for (uint8_t i = 0; i < 4; i++)
        s.longs[i] = MixColumn(s.longs[i]);
}

uint8_t P8(uint8_t v)
{
    v ^= v >> 4;
    v &= 0xF;
    return (0x6996 >> v) & 0x1;
}

void printHex(uint32_t val)
{
    std::cout << "␣␣0x"
              << std::setw(8)
              << std::setfill('0')
              << std::hex
              << val
              << std::dec
              << "\n";
}
```