Masters Thesis in the course of study "Security & Safety Engineering"

Symmetric block ciphers with a block length of 32 bit

Literature research, rescaling of the AES, analysis of 32 bit block length

Furtwangen, August 30, 2018

prepared by:

Stephan Rothe

Luisenstraße 22, 78120 Furtwangen

Matriculation Number: 256160

supervised by:

Prof. Dr. habil. Olaf Neiße

Prof. Dr. Dirk Koschützki

Declaration of Originality

I hereby declare that I have written the work independently and have not used any sources and aids other than those indicated and have marked quotations. I agree that the work may be viewed by third parties and quoted in compliance with copyright principles.

Furtwangen, August 30, 2018

Stephan Rothe

Abstract

Subject of the thesis at hand is the analysis of symmetric block ciphers with a block length of 32 bit. It is meant to give a comprising overview over the topic of 32 bit block ciphers. The topic is divided in the examination of three questions. It contains a list of state of the art block ciphers with a block length of 32 bit. The block ciphers are being described, focussing on the encryption function. An SPN-based cipher with 32 bit block length is being proposed by rescaling the AES cipher. The 32 bit block length results in certain security issues. These so called risk factors are analysed and mitigating measures are proposed. The result of the thesis is, that 32 bit block ciphers can be implemented in a secure manner. The use of 32 bit ciphers should be limited to specific use-cases and with a profound risk analysis, to determine the protection class of the data to be encrypted.
Key words: Symmetric block cipher; Block length 32 bit; State of the art 32 bit block ciphers; Rescaling Rijndael/AES; Analysis 32 bit block length

Contents

List of Figures

List of Tables

Abbreviations

3TDEA	Three-key Triple Data Encryption Algorithm
AES	Advanced Encryption Standard
ARX	Addition, Rotation, XOR
CBC	Cipher Block Chaining
CTR	Counter
DES	Data Encryption Standard
DLBCA	Design 32-bit Lightweight Block Cipher Algorithm
ECB	Electronic Code Book
FFX	Format-preserving, Feistel-based Encryption Mode
FPE	Format-preserving Encryption
FSR	Feedback Shift Register
GE	Gate Equivalent
GF	Galois Field
IDEA	International Data Encryption Algorithm
IoT	Internet of Things
LFSR	Linear Feedback Shift Register
MAC	Message Authentication Code
NIST	National Institute of Standards and Technology
NLF	Non-Linear Function
NLFSR	Non-Linear Feedback Shift Register
NSA	National Security Agency
RC5	Rivest Cipher 5
RFID	Radio Frequency Identification
SEA	Scalable Encryption Algorithm
SPN	Substitution Permutation Network
STS	Statistical Test Suite

1 Introduction

This thesis is dealing with block ciphers with a block length of 32 bit. Currently National Institute of Standards and Technology (NIST) approves two block ciphers: Three-key Triple Data Encryption Algorithm (3TDEA) and Advanced Encryption Standard (AES) [1, p. 4]. 3TDEA with a block length of 64 bit and AES with a block length of 128 bit. Even though there is no 32 bit block cipher yet specified by NIST, the need for so called ’lightweight cryptography’ rises. On the one side, developments such as the Internet of Things (IoT) result in computing devices getting smaller. An increasing amount of devices implement a processor and decent hardware to satisfy increasing demands, a tendency referred to as ’ubiquitous computing’. On the other side, there is a growing demand of these devices to communicate with each other, making them vulnerable against external influences. The data exchanged by these tiny devices can, if not confidential for itself, be confidential if a lot of data is aggregated by an adversary. Therefore a need to secure the communication is mandatory. A secure communication can be achieved by the use of cryptographic methods. In order to work properly in resource constraint environments new methods for encryption need to be investigated.

One solution might be to reduce the block length, as a large block length "[...] is more costly to implement" [2, p. 254]. Rescaling the block length to 32 bit is the topic of this thesis.

A distinct use-case for 32 bit block ciphers is a museum implementing so called beacons. These beacons own a 32 bit unique number, making it possible to address $2^{32} \sim 4,29$ billion different devices. A visitor to the museum can achieve more information on a specific exhibit, by using the smart-phone to receive the product number from the beacon, sending it to a server, which returns the desired information. Due to changed conditions the need to encrypt the product number arose. Changing the other parts of the specification, such as saving, sending and processing the 32 bit blocks of data should not be conducted. Therefore a 32 bit block cipher needs to be implemented.

The content of this thesis is structured following three questions:

1. Are there 32 bit block ciphers already existing?
   Answered in chapter 3 "State of the art 32 bit block ciphers"

2. Are there other possibilities to create a block cipher with 32 bit?
   Answered in chapter 4 "Rescaling the Advanced Encryption Standard"

3. What kind of security issues derive from 32 bit block length and how can these be mitigated?
   Answered in chapter 5 "Analysis of 32 bit block ciphers from a security perspective"

The size of a block will always be referred to as ’block length’, whereas the size of a key will always be referred to as ’key size’. As this is not done consistently in the literature, it helps to distinguish the exact reference, if only ’size’ or ’length’ is mentioned.

2 Terms & Definitions

In this chapter Terms and Definitions are being established to be used later on in the thesis. The following figure 2.1 shows the hierarchy and connection of some of the terms being explained later on in this chapter.

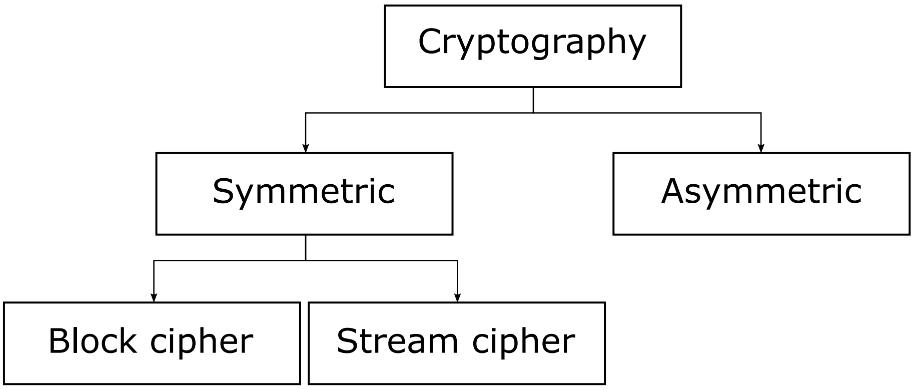


Figure 2.1: Modern Cryptography Hierarchy (adapted from [3, p. 3])

2.1 Symmetric vs. Asymmetric

The field of cryptography splits into two parts: Symmetric and asymmetric cryptography. The difference between both of them is the utilization of the key. In *symmetric* cryptography, also often referred to as 'secret-key cryptography', essentially the same key is being used for encryption and for decryption. In order to maintain confidentiality provided by a symmetric cipher, the key has to be kept secret. When an adversary gets to know the key, the ciphertext is compromised. In *asymmetric* encryption, also often referred to as 'public-key cryptography', the keys for encryption and decryption are different and it should be impossible (computationally infeasible) to calculate the decryption key out of the encryption key. [4]
Symmetric cryptography can be distinguished in two different cipher-methods (even though the distinction is not definitive): Stream ciphers and Block ciphers [4].

2.2 Stream cipher

Stream ciphers process the plaintext bit by bit. An encryption algorithm selects one plaintext bit, applies a couple of simple functions and returns a ciphertext bit. This way, every single bit is treated by itself, resulting in a consecutive ciphertext. Due to the internal state of a stream cipher, every output bit depends on the previously computed bits (this is also the main difference to block ciphers, where every block is calculated on its own and independent from previous computed blocks). A stream cipher therefore does maintain an inner state for encryption. The key bits usually flow in with an XOR to the currently processed plaintext bit. Decrypting the ciphertext would mean to take every bit and apply the respective inverse functions [4, p. 117f].

The later on presented Non-Linear Feedback Shift Registers (NLFSRs) are part of the stream ciphers, even though they have an input and output length of n bit. In the underlying design, individual bits of the plaintext are being manipulated and an inner state is maintained. After a certain number of rounds, the output is the manipulated plaintext, referred to as the ciphertext.

2.3 Block cipher

Block ciphers process plaintext in blocks comprising a fixed number of bits. The number of bits within a block is referred to as *block length.* The block length is only one attribute of a block ciphers characteristics. *Key size* and *number of rounds* are important for the design and the security of a block cipher as well.

The block length refers to the respective number of bits of the plaintext and the ciphertext. The following figure 2.2 shows a block cipher with a block length n and key size k.

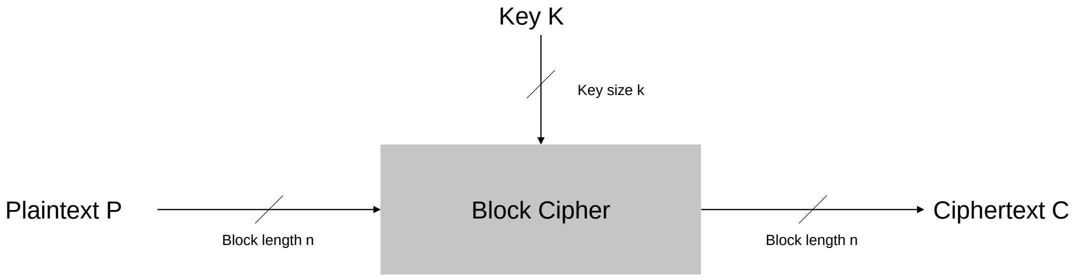


Figure 2.2: Model of a Block Cipher

For encryption, the block cipher performs functions on a block of data and outputs the respective ciphertext. Decryption often works the same way reverse, taking the appropriate number of bits of the ciphertext, applying the respective inverse functions and returning the plaintext P. The key K is part of the encryption and decryption process as it flows in during the application of the functions.

A plaintext block of size n can consist of 2^n different Strings of $0,1$. To map this input to a respective output, $2^n!$ possibilities are at hand. This number refers to the number of permutations, where the mapping of an arbitrary input to an arbitrary output is called a *permutation.* A permutation is specified by the associated key [2, p. 11].

If the number of bits of a plaintext is not a multiple of the block length, the last block has to be filled up with a chosen set of bits. This procedure is referred to as *padding* [2]. It will not be examined in the following thesis, since it is assumed, that only plaintext of exactly 32 bit is being encrypted.

Modes of operation can be used with any block cipher and are being implemented, when the plaintext is bigger than one block (which is the case in most applications). Implementing a mode of operation can result in meeting certain security goals such as raised *confidentiality* (for example Cipher Block Chaining (CBC)) or raised integrity (for example CBC-Message Authentication Code (MAC)) [2].

Most of the block ciphers comprise simple functions, since these often can be better implemented in software and also can be better realized in hardware. In order to maintain a decent level of confusion and diffusion (explanation in chapter 2.4) compared to the use of complicated functions, these simple functions need to be repeated several times. This is the reason why block cipher are often referred to as *iterated block cipher.* The characteristic *number of rounds* gives an indication how often the functions are being repeated during the process of encryption [2].

Block ciphers can be realised in a number of different designs. This thesis is going to differentiate the following three designs-types, described in the following chapters: Feistel, NLFSR and Substitution Permutation Network (SPN).

2.3.1 Feistel

Typically a cipher based on the Feistel design splits the block into two halves and applies a so called *round function* on one of the halves. When both halves have the same length it is called a *balanced Feistel.* If the length is not equal, it is called an *unbalanced Feistel.* The following basic steps are usually being implemented [4, p. 131]:

1. Separation of the plaintext in two halves left L_i and right R_i
2. Apply a so called round function f with a key K_i to the left side L_i
3. XOR (⊕) the output of the round function to R_i
4. Switch and rename both blocks: $L_{i+1} = R_i$ and $R_{i+1} = L_i \oplus f(R_i, K)$
5. Repeat a defined number of rounds i

Compared to SPN, Feistel-based ciphers have the advantage, that encryption and decryption operations are usually very similar, only the derived subkeys are used in a reverse order [2, p. 250]. On the other hand, the round function needs to be designed very carefully, since the security of the cipher fundamentally depends on it. A Feistel-based cipher often implements a higher number of rounds than SPN-based ciphers in order to fully confuse and diffuse the plaintext.

2.3.2 Non Linear Feedback Shift Register

A NLFSR is a Linear Feedback Shift Register (LFSR) "[...] whose current state is a Non-Linear Function (NLF) of its previous state [...]" [5, p. 1]. LFSRs are well suited for hardware implementation and provide good statistical properties. The disadvantage is, that the output sequences of a LFSR are usually easily predictable because of this linearity. Functions providing non-linearity can be applied during encryption to mitigate these kind of problems. These functions can be implemented using various techniques, usually combining multiple LFSR[1].

Before a NLFSR can be used for encryption, it has to be *initialized*. So called initialization vectors are processed by the cipher to set up the internal state of the feedback shift registers for encryption. After initialization, the plaintext flows in bit by bit.

NLFSRs are commonly counted to the Stream ciphers, but can be used to encrypt blocks of data as well [4].

2.3.3 Substitution Permutation Network

SPN-based ciphers usually split the block in several equally sized, small blocks (referred to as *words*) and apply a so called Substitution Box (S-Box) to every word. Within the S-Box the value of the word is exchanged (substituted) with another value. Afterwards a Permutation step is applied, combining a certain amount of words and 'mixing' the bits. Whereas the substitution step causes *confusion*, the permutation step creates *diffusion*. The concatenation of these two steps results in a strong encryption, as long as enough rounds are being implemented. The following figure 2.3 shows the basic concept of an SPN (with the S-Box (S) and the Permutation step (P)). The AES is the most prominent cipher implementing this design [4].

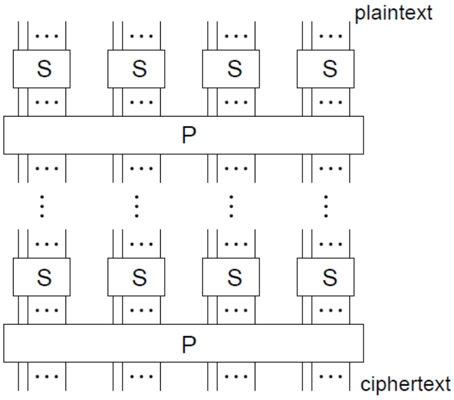


Figure 2.3: Substitution Permutation Network [2, p. 251]

The advantage of an SPN is, that in every round the whole datapath is subject to confusion and

[1] for more information refer to [2, p. 202ff]

diffusion. Therefore the number of rounds for an SPN-based cipher is usually smaller than for Feistel- or NLFSR-based ciphers.

2.4 Confusion and Diffusion

Confusion and Diffusion describe two basic principles for realising secure ciphers. The terms have been defined by Claude Shannon in 1948 [6]. Confusion describes the obfuscation of the relationship between the key and the ciphertext. Today this is often achieved with Substitution Boxes. Diffusion describes the influence of a plaintext bit on a lot of ciphertext bits to obfuscate any statistical characteristics within the plaintext [3, p. 65].

2.5 Security of a cipher

For the security of a cipher many definitions are existent. This chapter gives a brief overview of definitions, which are being used later on this thesis. Since this thesis is dealing with symmetric block cipher, the main security goal of block ciphers is used for the definition of security, which is confidentiality.

Often the security of a cipher is reduced to the key size. The longer the key, the better. But: Key sizes for symmetric ciphers are only decisive if the full key search is the best known attack (*brute forcing the key*). As shown later in chapter 5 "Analysis of 32 bit block ciphers from a security perspective", for 32 bit block ciphers, other attacks might be better than brute forcing the key, when the block length is as short as 32 bit. Other characteristics of a cipher, such as block length and number of rounds are therefore important security parameters as well [2].

In the context of this thesis, security of block ciphers follows two distinct approaches. Firstly, Menezes et al. defined the perfect secrecy for a cipher with the following words: "A cipher provides perfect secrecy (unconditional security) if the ciphertext and plaintext blocks are statistically independent" [2, p. 251]. Therefore the output of a block cipher should be hard to distinguish from a random sequence. In other words: a cipher should be able to provide good confusion and diffusion.

Secondly, a hint for the security of a cipher is the presence of any attacks on a *round reduced* version of this cipher, which leads to disclosure of the key or plaintext. Reducing the rounds can benefit cryptanalysis resulting in certain attacks being more sufficient. On the way to break a plaintext encrypted with the full number of rounds, usually attacks on round-reduced versions are published previously [2].

2.6 Collision

A block cipher is a pseudo-random permutation. Therefore, if the ciphertexts match, the plaintexts are the same as well. If two or more blocks match in the output of a cipher, a *collision* is found, which can be exploited by an adversary. The probability of the occurrence of collisions can be calculated.

3 State of the art 32 bit block ciphers

Part of this chapter is the presentation of the results of a profound literature research on already existing, state of the art 32 bit block ciphers. In the following, each of the identified block ciphers is getting described in more detail. At the end of this chapter, the presented ciphers are summarized in table 3.6.

It is important to notice, that the list is by no means exhaustive. Even though this literature research has been conducted thoroughly, it might turn out, that other ciphers with a 32 bit block length emerge.

3.1 Lightweight Block Ciphers with 32 bit block length

In the following, the identified 32 bit block ciphers are presented and briefly described. The focus of this chapter is the description of the encryption algorithm and the listing of a selection of published attacks. The key-expansion and decryption are not in the main focus. This is due to the fact, that the key-expansion is mostly not or indirectly connected to the block length. The decryption algorithm mostly consists of inverse functions conducted in opposite order, compared to the encryption algorithm.

Some of the presented ciphers have a design implementing a NLFSR. Even though they are technically counted to the stream ciphers, they are listed here because they also allow the input of whole blocks into the encryption algorithm.

Where possible, only the 32 bit version of a cipher will be explained in more detail. Parameters for other versions of the respective families can be investigated in detail in the referenced publication. The ciphers are presented in a chronological order starting with the oldest.

KeeLoq

KeeLoq encrypts a block of 32 bit with a secret 64 bit key. It was designed in the mid 1980s. As can be seen in figure 3.1 KeeLoq uses a 32 bit Shift Register for the Plaintext and a 64 bit Shift Register for the Key. KeeLoq gets initialized by the use of a 32 bit initialization vector.

For encryption, a NLF is applied to specific positions of the 32 bit Shift Register (for encryption positions 31, 26, 20, 9, 1 (refer to figure 3.1). The resulting bit is XORed to the rightmost bit of the key and positions 16 and 0 and inserted into the 32 Shift Register bit from the left side. This causes

a shifting of one bit to the right. The 64 bit Key Register is also shifted one bit to the right and the NLF is applied again. This procedure is repeated 528 times, which is also the number of rounds of KeeLoq. After 528 rounds, the content of the 32 bit register is called the ciphertext. [7, p. 9ff]
The main case of use of KeeLoq was the remote keyless entry system for automobiles. Since multiple attacks have been published, such as [8] and [9], KeeLoq is mostly not in use any more [7, p. 9ff].

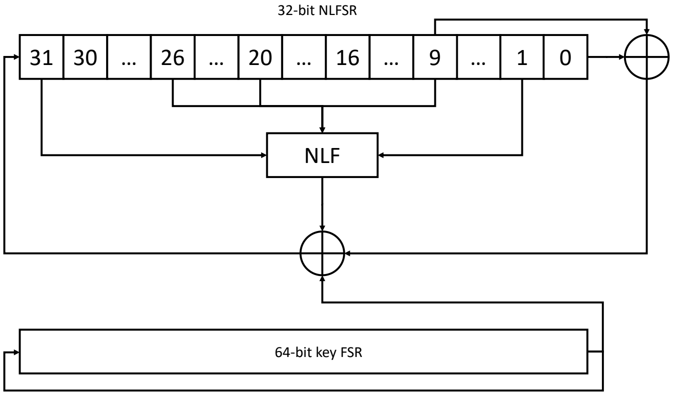


Figure 3.1: KeeLoq encryption (adapted from [7, p. 11])

RC5

Rivest Cipher 5 (RC5) is a Feistel-based cipher proposed by Ronald Rivest in 1995. It is designed for variable block lengths (originally proposed: 32, 64, 128 bit), variable key sizes (0 - 255 Byte) and variable number of rounds (0 - 255, 12 initially proposed for 32 bit block length). For the encryption, the plaintext is separated in two halves of size ω. Figure 3.2 shows a full round of RC5 encryption, consisting of two half rounds using three simple operations [10]:

- Circular shifting to the left by j bit ($\lll (j)$), where j corresponds to the $\log_2 \omega$-least significant bits of the other word (For a block length of 32 bit, ω is 16 bit long, and $j = \log_2 \omega = 4$) - this is referred to as *data-dependent rotation* [10, p. 90]
- addition modulo ω ($\boxplus$)
- bitwise XOR ($\oplus$)

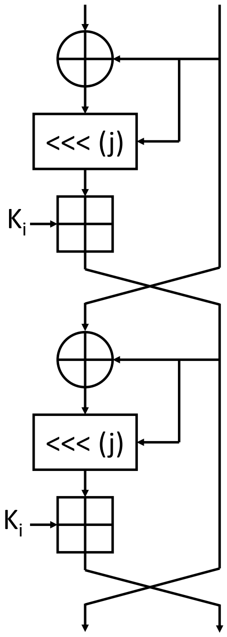


Figure 3.2: One round of RC5 encryption

In 2006 an attack based on differential cryptanalysis was presented, which requires 2^{44} chosen plaintexts [11]. It is also shown in this paper, that the 128 bit version of RC5 is weaker than initially expected. Rivest himself described the 32 bit version of RC5 "primarily for researchers who wish to examine the security properties of a natural 'scaled-down' RC5" [10, p. 89]

Skip32

Skip32 is a 32 bit Block cipher based on the SkipJack cipher proposed by the National Security Agency (NSA). Skip32 is designed with an 80 bit secret key and 24 rounds. It was presented in 1999 by Greg Rose, but never published officially. Therefore no description of the exact structure can be presented herein. Skip32 turned out to be useful in obfuscating small values, for example database ID's used in URLs [12].

The underlying cipher SkipJack is based on the design of an unbalanced Feistel. SkipJack is reported to be vulnerable to differential attacks, resulting in revoking the usage of SkipJack by the NIST: "[...] approval for the use of SKIPJACK is being withdrawn, as its security strength is now considered inadequate." [1, p. 4]. Since Skip32 is based on SkipJack, the conclusion can be drawn to not use Skip32 anymore too.

Scalable Encryption Algorithm

Scalable Encryption Algorithm (SEA) is a block cipher with a variable block length, key size and round number. It is a balanced Feistel based cipher, where b is the processor or word size and n_b the number of words per Feistel branch. The block length and key size are both computed as a multiple of $6b$. Consequently the block length and key size are equal. A block length of 32 bit can not be achieved, since 32 is not a multiple of 6. On the other hand, even smaller block length can be derived theoretically with this cipher, although the authors suggested a minimum word size $b \geqq 6$. The number of rounds n_r is computed by the use of the following equation, which the authors suggested in their proposal [13, p. 11]:

$$n_r = \frac{3n}{4} + 2 * (n_b + \lfloor \frac{b}{2} \rfloor)$$

With the number of rounds calculated that way, linear and differential attacks ought to be resisted. The authors also suggested, that the number of rounds has to be odd. So if this equation results in an even number, the number 1 has to be added. Figure 3.3 shows the Feistel structure of the cipher. The following functions are being used during the encryption with SEA [13, p. 3f]:

- bitwise XOR ($\oplus$)
- Applying the following three bit S-Box (S):

Table 3.1: S-Box of SEA

x	0	1	2	3	4	5	6	7
S(x)	0	5	6	7	4	3	1	2

- Rotating words of size n_b (R) (R^{-1})
- Bit rotation (r)
- addition modulo 2^b ($\boxplus$)

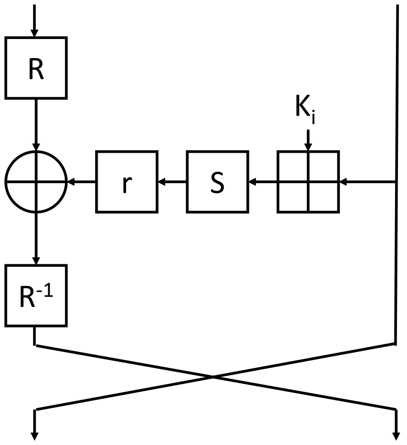


Figure 3.3: One round of SEA encryption (adapted from [13, p. 4])

The authors tested SEA against various attacks, including differential and linear cryptanalysis with the result of being provable secure against these kind of attacks. Even though SEA does not make use of 32 bit block length, SEA is part of this listing, because of its variability, and the fact, that even significantly smaller block lengths than 32 bit are conceivable.

KATAN / KTANTAN

In 2009 a new family of lightweight Block ciphers was presented, consisting of six different ciphers divided in two so called flavors: KATAN and KTANTAN. Both are separated into three block lengths: 32, 46 and 64 bit. For all the six ciphers, a key size of 80 bit is intended. KTANTAN is optimized for the use in Hardware. The Key is burnt into the device and can not be altered once initialized. That makes this flavor more compact and therefore faster. The main difference between the two families therefore is the key schedule [14].
A design goal of KATAN and KTANTAN was to develop a very lean cipher, which needs the smallest possible amount of Gate Equivalents (GEs). This makes the cipher family very useful for the implementation in highly resource constrained environments, such as Radio Frequency Identification (RFID) tags [14].
KATAN and KTANTAN make use of two LFSR implementing two simple functions [14]:

- bitwise XOR ($\oplus$)
- bitwise AND ($\wedge$)

Non-linearity is applied by the use of an irregular update rule (IR) in a few specific rounds. The exact sequence is described in Appendix A of the proposal of KATAN and KTANTAN [14, p. 287]. For encryption, the plaintext is separated into two registers: L_1 and L_2 (figure 3.4). For the 32 bit versions, $|L_1|$ is 13 bit, $|L_2|$ is 19 bit. Every round, two NLF f_a and f_b are applied [14]:

$$
\begin{aligned}
f_a(L_1) =& L_1[x_1] \oplus L_1[x_2] \oplus (L_1[x_3] \cdot L_1[x_4]) \oplus (L_1[x_5] \cdot IR) \oplus K_a \\
f_b(L_2) =& L_2[y_1] \oplus L_2[y_2] \oplus (L_2[y_3] \cdot L_2[y_4]) \oplus (L_2[y_5] \cdot L_2[y_6]) \oplus K_b
\end{aligned}
$$

After 254 rounds, the combined contents of L_1 and L_2 form the ciphertext. The Parameters K_a and K_b represent the applied subkey bit. The values given in table 3.2 have to be used for x and y [14, p. 272ff]:

Table 3.2: Parameters for 32 bit versions of KATAN / KTANTAN [14]

Parameter	x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3	y_4	y_5	y_6
value for 32 bit versions	12	7	8	5	3	18	7	12	10	8	3

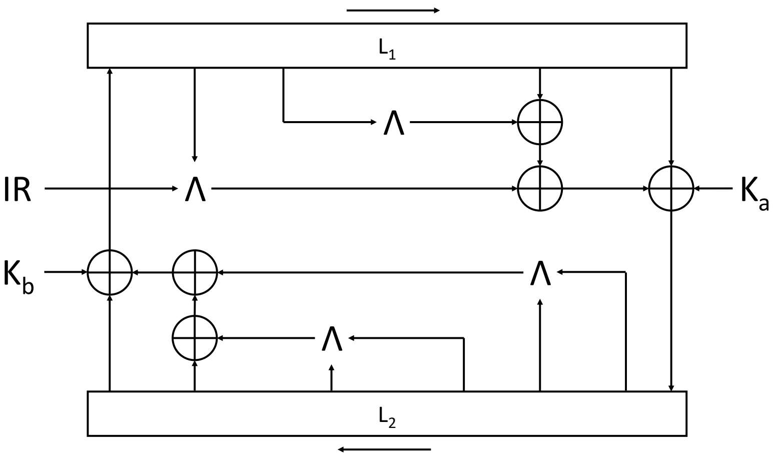


Figure 3.4: KATAN32/KTANTAN32 encryption (adapted from [14, p. 279])

KTANTAN with a block length of 32 bit is reported broken, only "[...] requiring $2^{28.44}$ time (half a minute on a current CPU) to recover the full 80 bit key" [15, p. 213ff]. KATAN with a block length of 32 bit is suspected of being vulnerable to Meet-in-the-middle Attacks: Zhu and Gong proposed an attack on 175 rounds, recovering the master key [16, p. 313ff].

Hummingbird 2

Hummingbird 2 is the successor of the Hummingbird cipher and encrypts blocks of 16 bit length. It can be used to encrypt 32 bit blocks as well by concatenating 2 16 bit blocks. Therefore it is getting listed in this collection.

Due to its compact hardware implementation and low power consumption, it is very useful for resource constraint environments, such as RFID tags or wireless sensors. As can be seen in the following figure 3.5 the cipher does not implement a key schedule, but keeps an internal state R^i of size 128 bit throughout the process of encrypting larger texts. Before encryption, four rounds are being processed, where the internal state is getting initialized with a 64 bit initialization vector. After initialization follow four rounds of encryption and another four rounds to update the internal state again (figure 3.5). The key K of size 128 bit is split into eight subkeys K_i with $i \in \{1,2,3,4,5,6,7,8\}$ of 16 bit size. The subkeys are being used for primal initialization and encryption of the plaintext P^i into ciphertext C^i [17, p. 19ff].

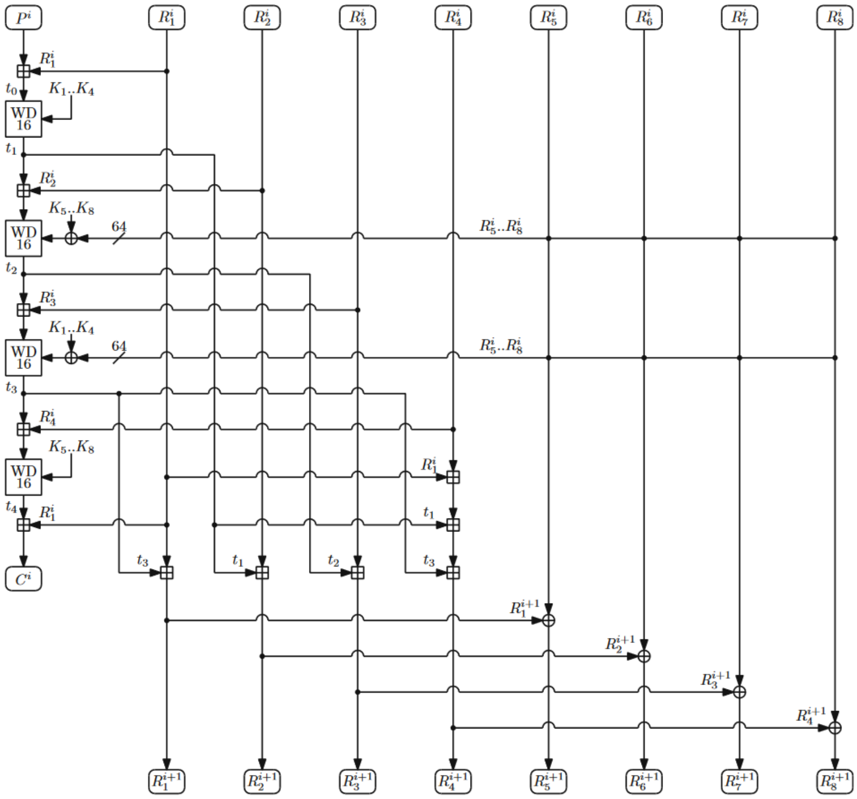


Figure 3.5: One round of Hummingbird 2 encryption [18, p. 472]

The herein focussed Hummingbird 2 encryption consists entirely of three simple functions [17, p. 20]

- addition $mod 2^{16}$ ($\boxplus$)
- bitwise XOR ($\oplus$)
- round function WD16

The round function WD16 can be seen as a block cipher itself, with a block length of 16 bit and a key size of 64 bit. It is based on an SPN and is getting used during initialization and encryption of Hummingbird 2. The 64 bit key is split into four 16 bit subkeys K_i with $i \in \{1,2,3,4\}$ and is XORed to the 16 bit datapath before each of the 4 rounds. The following figure 3.6 shows one round of WD16.

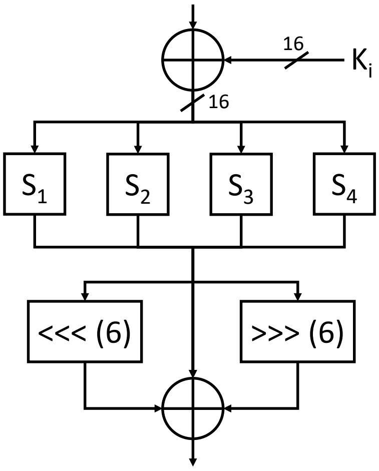


Figure 3.6: One round of WD16 (adapted from [18, p. 470])

Three functions are being used during one round of WD16

- bitwise XOR (⊕)
- applying the following four different 4 bit Substitution Boxes S_1 to S_4:

Table 3.3: S-Boxes of WD16

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S_1(x)$	7	C	E	9	2	1	5	F	B	6	D	0	4	8	A	3
$S_2(x)$	4	A	1	6	8	F	7	C	3	0	E	C	5	9	B	2
$S_3(x)$	2	F	C	1	5	6	A	D	E	8	3	4	0	B	9	7
$S_4(x)$	F	4	5	8	9	7	2	1	A	3	0	E	6	C	D	B

- Circular shifting to the left by 6 bit ($\lll$ (6)) and to the right by 6 bit ($\ggg$ (6))

In 2014 Saarinen published a related key attack against Hummingbird 2, "[...] which effectively halves the cipher's key size" [18, p. 480].

FFX

Format-preserving, Feistel-based Encryption Mode (FFX) describes a mode of operation for an underlying symmetric block cipher. It enables the user to encrypt plaintext with arbitrary block length. Additionaly it has the advantage to preserve the format of the plaintext, resulting in the ciphertext having the same format as the plaintext. Therefore the term Format-preserving Encryption (FPE) is also often referred. FFX can be used to encrypt Credit Card numbers, so that the size and the format is the same in plaintext and ciphertext. By doing so, an already existing specification does not need to be changed in order to process the Credit Card number in an encrypted way.

In the context of this thesis, it is also conceivable to use FPE to encrypt a plaintext of 32 bit, resulting in a ciphertext of 32 bit.

NIST Special Publication 800-38G describes two different modes of FPE (called FF1 and FF3) [19]. The main difference between the two is the number of rounds of the mode of operation. FF1 uses 10 rounds, FF2 uses 8 rounds.

FF1 and FF2 both use a Feistel-based scheme. The following figure 3.7 shows the first round of FFX. For the round function F_K, an approved 128 bit block cipher is used in CBC mode to deduce a MAC (currently only AES meets these requirements). The resulting MAC is added modular to the left side of the plaintext. In the second round, the MAC is created of the left side and added modular to the right side and so on. The input parameters for the round function are Key K, length n, Tweak T and the round number starting with 0. The Tweak is an arbitrary number, which does not necessarily need to be kept secret. After 10 (respective 8) rounds, the resulting string is the ciphertext and thus the output of the mode of operation [19].

The downside of using the FFX mode of operation is the use of two encryption schemes and therefore its bad performance. The round function does not consist of simple transformations as in the other presented ciphers, instead an AES is used. The FFX mode of operation can therefore not be declared really useful for the implementation in resource constrained environments.

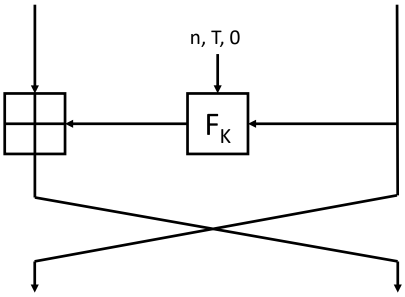


Figure 3.7: First Round of the FFX mode of operation encryption (adapted from [19, p. 10])

SIMON / SPECK

In 2013 two new families of lightweight block ciphers were proposed by the NSA called SIMON and SPECK. Both consist of ciphers with variable block lengths (32, 48, 64, 96, 128 bit) and key sizes (64, 72, 96, 128, 144, 192, 256 bit). The number of rounds depends on the chosen block length and key size and is therefore variable as well (The 32 bit versions use a 64 bit key and 32 rounds (SIMON) respective 22 rounds (SPECK). All in all 20 different ciphers are proposed within these families. For an overview of all the proposed block lengths, key sizes and number of rounds, refer to [20, p. 10 & p. 15].

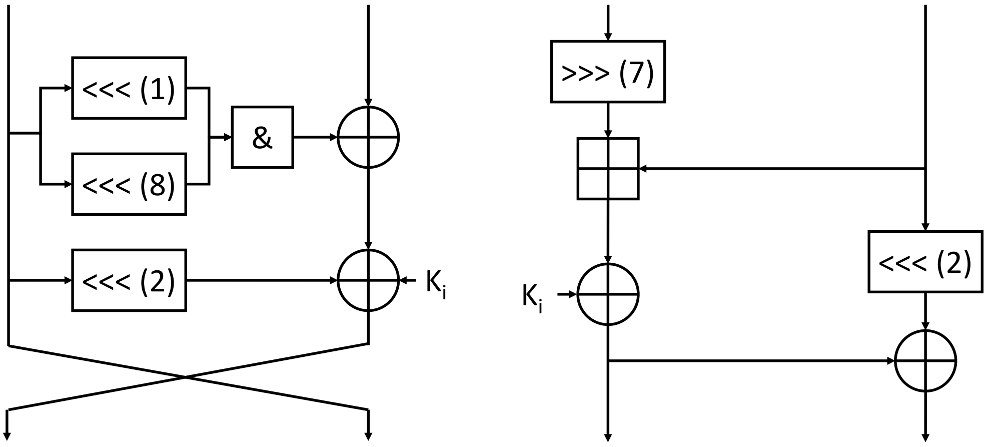
Both families, SIMON and SPECK, are very performant and effective compared to the formerly presented block ciphers. Their very simple implementation results in a high throughput and low energy consumption, making these ciphers very useful in resource constrained environments [21, p.

9]. The difference between SIMON and SPECK is their designated use in software and hardware. Though both of the ciphers can be used in hard- and software, SIMON is optimized for the performance in hardware, whereas SPECK is optimized for the performance in software. This satisfies the design-criterion of cross-platform flexibility, which the authors think, needs to be a goal for lightweight block ciphers. The flexibility is realized by the use of a balanced Feistel-based structure for SIMON. SPECK on the other hand can be visualized by a composition of two Feistel-like maps. The flexibility is also realized by using three simple functions: Addition, Rotation, XOR (ARX). This is illustrated in the following table 3.4:

Table 3.4: Functions used for encryption with SIMON and SPECK

SIMON	**SPECK**
• Circular shifting to the left by j bit ($\lll (j)$)	• Circular shifting to the left by 2 bit ($\lll$ (2)) and to the right by 7 bit ($\ggg$ (7))
• bitwise AND (&)	• addition $mod 2^n$ ($\boxplus$)
• bitwise XOR ($\oplus$)	• bitwise XOR ($\oplus$)

The following figure 3.8 shows one round of SIMON and SPECK using the notation above. K_i denotes the respective subkey for each round. After 32 rounds for SIMON and 22 rounds for SPECK the ciphertext is generated by the combination of the output strings of the two branches.



(a) SIMON (b) SPECK

Figure 3.8: One round of SIMON32 and SPECK32 encryption [22, p. 556]

SIMON and SPECK both have been heavily analysed by cryptographers around the world. Currently, the best known attacks on SIMON and SPECK with a block length of 32 bit are linear and differential cryptanalyses. The attacks mainly use the fact, that SIMON and SPECK base on ARX [22] [23] [24]. So far, no full round attack has been published.

SIMECK

SIMECK is a family of ciphers presented in 2015 and based on SIMON and SPECK. The SIMECK family of block ciphers consists of three different ciphers, distinguished by their respective block length, key size and number of rounds. The focus herein will be set on the cipher with the block length of 32 bit. As well as SIMON and SPECK, SIMECK implements a balanced Feistel-based structure and makes use of ARX. The following simple functions are being used during the encryption:

- Circular shifting to the left by j bit ($\lll (j)$)
- bitwise AND (&)
- bitwise XOR ($\oplus$)

SIMECK has a 64 bit secret key and requires 32 rounds for encryption. K_i denotes the subkey XORed every round. As can be observed in the following figure 3.9, the round function of SIMECK is very similar to the previously presented round function of SIMON, varying only in the number of circular shifts to the left [25].

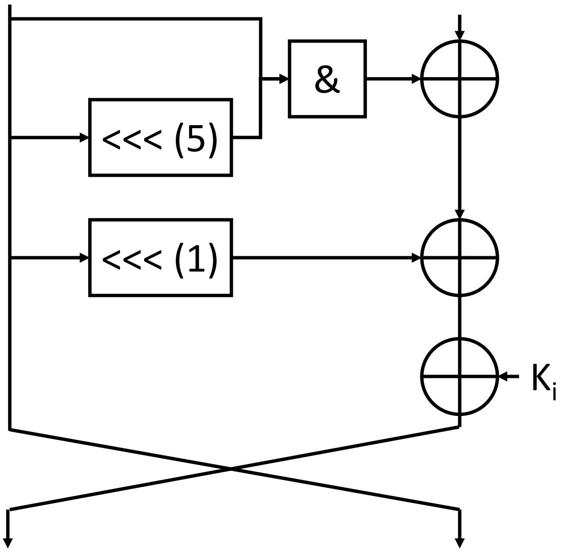


Figure 3.9: One round of SIMECK encryption (addapted from [25, p. 5])

Similar to SIMON and SPECK, only attacks on round-reduced version of SIMECK are yet published [26] [27] [28]. No attack on the full number of rounds of the three proposed versions has yet been published.

Design 32-bit Lightweight Block Cipher Algorithm

Design 32-bit Lightweight Block Cipher Algorithm (DLBCA) is a cipher presented in 2017 using a balanced Feistel-based structure with elements of an SPN [29]. DLBCA makes use of an 80 bit secret key and 15 rounds. The round function consists of the following four functions:

- bitwise XOR (⊕)
- applying the following 4 bit S-Box (S):

Table 3.5: S-Box of DLBCA

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S(x)	F	C	2	7	9	0	5	A	1	B	E	8	6	D	3	4

- Bit Permutation
- circular shifting to the left by 12 bit ($\lll (12)$)

The whole round function is shown in the following figure 3.10

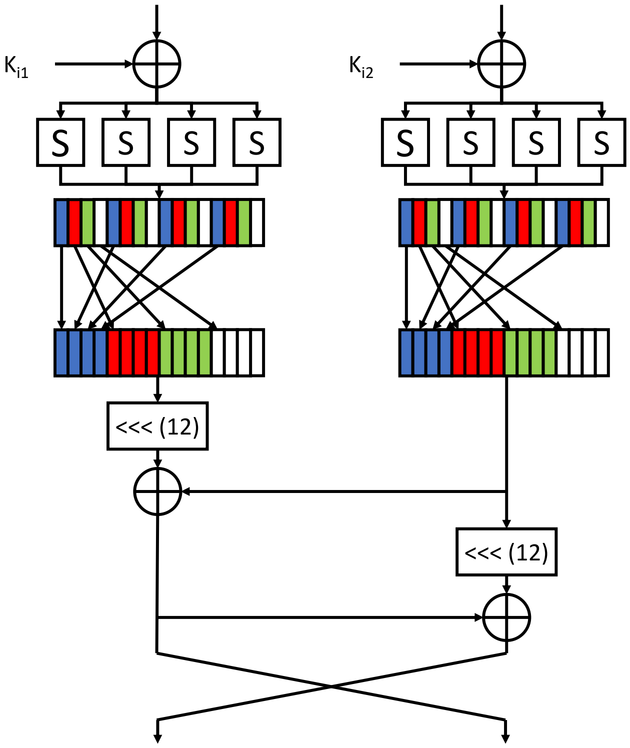


Figure 3.10: One round of DLBCA encryption [29, p. 2]

The cipher DLBCA has been developed recently, so it has not been subject to a lot of cryptanalyses yet. In their publication [29], the authors state, "that DLBCA is the most secure than other algorithms considered in this paper in the terms of differential and boomerang attacks" (compared ciphers are TWINE [30], Lblock [31], PRESENT [32] and KLEIN [33]). Independent research has not been published yet.

Table 3.6: List of state of the art 32 bit block ciphers

Name	Year	Block length	Key size	Type	Number of Rounds	Implementation
KeeLoq	mid 1980s	32	64	NLFSR	528	Hardware
RC5	1994	32, 64, 128 (64 suggested)	0 - 2040 (128 suggested)	Feistel, ARX	0 - 255 (12 originally suggested, for now 18-20 rounds are recommended)	Software
Skip32	1999	32	80	Feistel	24	Software
SEA	2006	variable	variable	Feistel	variable	
KATAN	2009	32, 48, 64	80	NLFSR	254	Software
KTANTAN	2009	32, 48, 64	80	NLFSR	254	Hardware (Key is burned into device - can not be changed)
Hummingbird 2	2011	16	128	hard to classify, WD16 is an SPN	4+4+4	Hard- and Software
AES FFX	2010	variable	128, 192, 256 (AES as underlying block cipher)	Feistel (Round Function: SPN)	10, 8 (FF1, FF3)	
SIMON	2013	32, 48, 64, 96, 128	64, 72, 96, 128, 144, 192, 256	Feistel, ARX	22, 23, 26, 27, 28, 29, 32, 33, 34	Hardware
SPECK	2013	32, 48, 64, 96, 128	64, 72, 96, 128, 144, 192, 256	Composition of two Feistel-like maps, ARX	32, 36, 42, 44, 52, 54, 68, 69, 72	Software
SIMECK	2015	32, 48, 64	64, 96, 128	Feistel, ARX	32, 36, 44	Hard- and Software
DLBCA	2017	32	80	Feistel	15	

3.2 Conclusion

As presented in this chapter, there are quite a few block ciphers with a block length of 32 bit. Ciphers with even smaller block lengths are conceivable, for example SEA (16bit), Hummingbird 2 (16bit) and FFX (variable). Some of the presented ciphers are vulnerable to various attacks, such as linear or differential cryptanalysis and some are considered broken.

All ciphers implement some kind of Feistel structure or a NLFSR. This is surprising, since modern block ciphers, such as AES make use of SPN. A reason for this might be, that Feistel and NLFSR round functions are considered to be simpler and usually offer a leaner implementation than SPN. The round function of Feistel Networks consists mainly of a combination of simple functions, such as XOR, modular addition and rotation/shifting. These functions are easily implemented in hardware and are therefore useful in resource constraint environments, making the ciphers implementing those functions suitable for lightweight designs. The downside of the use of simple functions is, that the number of rounds needs to be high in order to apply sufficient confusion and diffusion. This can take time and also costs a lot of resources. For SPN the number of rounds usually is smaller than for Feistel Networks or NLFSRs.

The next chapter 4 "Rescaling the Advanced Encryption Standard" shows a proposal of a 32 bit block cipher based on SPN in order to fill this gap and to investigate, if it is reasonable to implement an SPN-cipher, even for resource constraint environments.

4 Rescaling the Advanced Encryption Standard

As seen before, in the list of found 32 bit ciphers, there are no 32 bit ciphers yet implementing an SPN. To investigate this in more detail, part of this thesis is to rescale the tried and tested block cipher AES from a proposed minimum block length of 128 bit to 32 bit. For this purpose, AES is being described briefly followed by an explanation, how it is rescaled. Statistical testing of the output of the rescaled version provides indication on the ability of the cipher to obfuscate the input and therefore how fast the cipher applies confusion and diffusion.

4.1 Analysis of AES

AES is the most prominent representative of SPN block ciphers. The AES cipher is based on the Rijndael proposal by Vincent Rijmen and Joan Daemen. The difference is, that AES supports only three members of the 25 originally proposed ciphers in the Rijndael familiy: The block length of 128 bit and variable key sizes of 128, 192, 256 bit. Depending on the key size, either 10, 12 or 14 rounds are standardized. The Rijndael family supports variable block lengths and key sizes, ranging from 128 bit to 256 bit in 32 bit steps (refer to table 4.1). The associated number of rounds n_r is a function of the block length n and the key size k as shown in table 4.1.

Table 4.1: Number of rounds n_r as a function of the block length n and key size k for Rijndael (adapted from [34, p. 42])

n_r		n				
		128	160	192	224	256
k	128	10	11	12	13	14
	160	11	11	12	13	14
	192	12	12	12	13	14
	224	13	13	13	13	14
	256	14	14	14	14	14

The standardization process of AES has been conducted with openness and care. The fair winner has been Rijndael, due to its performance over the design criteria, compared to the other candidates. One of the main aspects is its simplicity, based on the property of Rijndael to be completely describable in mathematical formulas. Due to the fact, that every single transformation can be retraced with mathematics, there are no possibilities for the implementation of any kind of fraud

[1]. Therefore Rijndaels and also AESs acceptance is very high.
The following figure 4.1 shows one round of AES with a block length of 128 bit. The first and the last round differ from the pictured figure. The first round consists only of an initial key addition step, in the last round the Mix column step is omitted. AES operates on words of 8 bit[2].

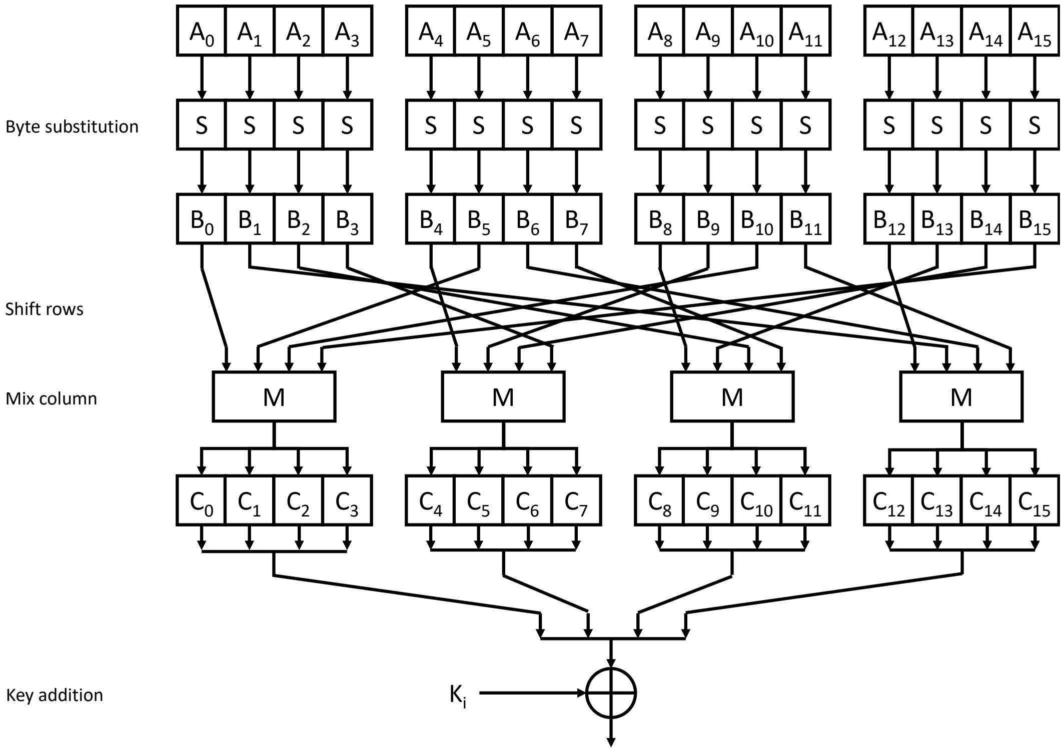

A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 A12 A13 A14 A15
Byte substitution
S
B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15
Shift rows
Mix column
M
C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 C10 C11 C12 C13 C14 C15
Key addition
Ki

Figure 4.1: One round of AES encryption (adapted from [3, p. 117])

4.2 Rescaling AES to 32 bit block length

AES has been chosen to be rescaled out of the following reasons:

- Simplicity: every transformation can be reproduced mathematically. Even the selection of the S-Box, as the main source of confusion and non-linearity, is reproducible.
- Acceptance: Rijndael won the public AES call for tenders. AES is one of the only block ciphers approved by NIST.
- Security: AES is provable secure against all yet known kinds of attack.

[1] As it happened for example with the Data Encryption Standard (DES)

[2] A detailed description of the four different steps and their mathematical background is available in the literature (for the full description of Rijndael for example [34])

- Variable block lengths with steps of 32 bit: Rijndael, as the basis for AES, is designed for variable block lengths in 32 bit steps. Therefore it is possible to rescale AES to 32 bit.

Rescaling AES from the above presented figure 4.1 with a block length of 128 bit to 32 bit can be obtained theoretically by two different approaches:

1. Reducing the word-size and therefore also the size of the S-Box from 8 bit to 2 bit resulting in 16 S-Boxes with 2 bit each ($16 * 2bit = 32bit$).
2. Reducing the number of words and therefore the number of S-Boxes resulting in 4 S-Boxes with 8 bit each ($4 * 8bit = 32bit$).

Approach number one is only a theoretical possibility. As the S-Box is the only source of non-linearity, reducing it to 2 bit would make it vulnerable to cryptanalysis. For a 2 bit S-Box there would be theoretically 24 permutations to choose from, disregarding the selection of the S-Box with mathematical formulas (as a main feature of AES). For comparison, an 8 bit S-Box (as proposed in AES) has $\log_2(2^8!) \simeq 2^{1684}$ permutations available.

Approach number two is the preferred choice, as changing the number of words is the procedure within the Rijndael-family to obtain the variable block lengths. For example the Rijndael cipher with block length of 160 bit is operating on five 32 bit words. Whereas the minimum proposed block length is 128 bit, it is possible to shorten the block length to 32 bit by the means of decreasing the number of 32 bit words from 4 to 1. The following figure 4.2 shows the updated round function developed in this thesis.

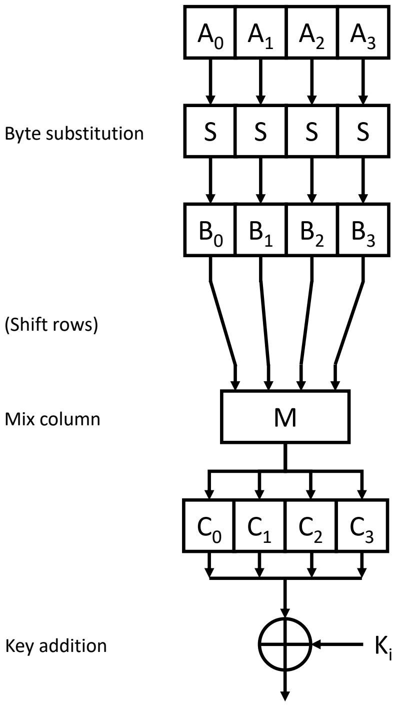


Figure 4.2: One round of rescaled AES encryption

Some of the steps of AES are affected of this rescaling and these changes to the round function. The effects are described in the following subchapters.

Key addition step

The key addition step XORs the subkey to the whole datapath. For AES, a 128 bit subkey is XORed to the datapath. Key addition is conducted as the first and the last step during encryption and is a common approach in modern cryptography [3, p. 163ff]. By rescaling to 32 bit block length, only a 32 bit subkey is XORed to the datapath. Assuming a 128 bit key, it would take 4 rounds to use only one subkey during encryption. The problems arising from that will be analysed in more detail in chapter 4.4 "Conclusion".

Byte substitution step

The Byte substitution step consists of a bijective mapping of the input values to the output values. It is a non-linear function and the main source of confusion within AES. The well probed S-Box of AES, with its underlying arithmetic in the Galois Field (GF)(2^8)[3], does not have to be changed. All the work, that has been done before, concerning the S-Box and its cryptanalysis, is still valid for the rescaled version of AES. It is assumed, that confusion still works as good as in the original AES proposal.

Shift rows step

The Shift rows step is one of the sources of diffusion within AES. Within this step, 16 8 bit words are concatenated to 4 32 bit words, which are the input to the Mix column step. As the datapath of the rescaled AES consists of only one word with a length of 32 bit, the Shift rows step is obsolete. This can be observed in figure 4.2. Therefore, the Shift rows step, as one of the two diffusion steps, has to be omitted due to the rescaling. An analysis of the effects is shown in chapter 4.3.3 "Analysis of the diffusion process".

Mix column step

The main source of diffusion, the Mix column step, can be implemented for 32 bit block length as proposed originally in AES. The Mix column step is realized as a matrix multiplication in GF(2^8). The following equation shows the matrix multiplication in the Mix column step:

[3] Calculating the Inverse in GF(2^8) with the irreducible polynomial $P(x) = x^8 + x^4 + x^3 + x + 1$ followed by an affine transformation. More information in [3, p. 118ff] or [34]

$$\begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \end{pmatrix}$$

Since there is only one Mix column 'box' for a block length of 32 bit, the whole datapath is subject to diffusion during one round. This results in every state byte S_i being dependent on every plaintext byte P_i with $i \in \{0,1,2,3\}$ after just two rounds, considering that the first round of AES only encompasses the Key addition step (Matrix Multiplication above).

Key schedule

The Key schedule for the rescaled version of AES has not been adapted in this thesis. Therefore a 128 bit key is needed for the input, the related subkeys are derived by the use of the proposed Key schedule of Daemen and Rijmen.
The number of permutations of the rescaled version of AES is approx. $\log_2(2^{32}!) \simeq 2^{131242625453}$ which is way bigger than the amount of possible keys 2^{128}. It is therefore still possible, that a random permutation can be specified by a single key. However, statistically, the possibility of two keys specifying the same permutation, is higher when the block length is reduced to 32 bit compared to a block length of 128 bit.
After rescaling the AES, qualities of the cipher are tested. This is shown in the following chapter.

4.3 Statistical analysis on the rescaled AES

During the AES selection procedure, statistical tests have been conducted on all candidates, to test the confusion and diffusion qualities of the respective ciphers. These can be shown by analysing the outputs for specific input values (variable plaintexts and keys) and testing the presence of any statistical anomalies, which give an indication on the dependencies between input and output. Part of the Statistical Test Suite (STS) are 188 different statistical tests (for an overview refer to table 4.3) [35].

4.3.1 Applying the statistical test suite to the rescaled AES

The test suite being used during the AES selection process is used for the rescaled version of Rijndael as well[4]. Statistical testing is conducted to prove that the output is still statistically independent from the input. This is also an important security indication (refer to chapter 2.5 "Security of

[4] NIST STS has been improved by Landon Curt Noll, Riccardo Paccagnella, Tom Gilgan and is available in version 3.2.6 on https://github.com/arcetri/sts

a cipher"). A second goal of statistical testing in the context of this thesis is the probing, if it can be used to determine, how many rounds the cipher needs to apply full confusion and diffusion. This can be an indication to find out, how many rounds are needed for the rescaled AES to be secure. Therefore different numbers of rounds are used for the encryption of the test data during testing. Compared to the AES selection process, statistical testing for the rescaled AES is conducted with a smaller amount of test data, as modes of operations are not considered. The following test data is being generated, with a file size of 37.5 MByte each:

1. Plaintext counted up, encrypted with 128 bit keys of all zeroes.
2. Plaintext of all zeroes, encrypted with counted up 128 bit keys.
3. Plaintext counted up, encrypted with counted up 128 bit keys.

The test data is being concatenated in Electronic Code Book (ECB)-Mode. The resulting ciphertext is analysed with the STS. A so called p-value is generated for every single test out of the 188 tests, which gives an indication about the statistical quality of the ciphertext. The STS compares these p-values to previously defined limits. If the limit is surpassed, the outcome of the test is PASSED otherwise FAILED. For the testing of the rescaled version, the default settings of the STS have been applied.

4.3.2 Results of statistical testing

The results of the statistical testing is shown in table 4.2. The three different test cases are shown in 3 columns, each separated into PASSED and FAILED. The numbers indicate the number of tests passed and failed for the specified number of rounds.

Table 4.2: Results of the statistical testing - number of tests PASSED and FAILED

	Test data 1		**Test data 2**		**Test data 3**	
Number of Rounds	PASSED	FAILED	PASSED	FAILED	PASSED	FAILED
1	1	187	30	158	1	187
2	132	56	186	2	186	2
3	188	0	187	1	188	0
4	185	3	183	5	186	2
8	188	0	188	0	187	1
12	185	3	187	1	188	0
16	188	0	187	1	183	5
20	188	0	187	1	188	0
24	187	1	188	0	178	10
26	188	0	188	0	183	5
28	187	1	185	3	187	1
30	184	4	186	2	186	2
31	186	2	188	0	185	3
32	186	2	188	0	188	0
56	186	2	186	2	187	1
64	186	2	188	0	185	3
128	188	0	186	2	188	0

After one round, most of the statistical tests fail, but after 3 rounds most of the tests pass successfully. It is important to say, that this is not a constant process. If all tests pass for one round, it does not say, that after one more round, all the tests will pass again (compare Test data 1 for 3 and 4 rounds). That statistical tests might fail is a normal behaviour and can be observed for the statistical testing of random numbers as well[5]. A reason for this behaviour might be the fact, that pseudo-random strings are being analysed. Due to the pseudo-randomness it might happen, that tests fail, even for a very high number of rounds (for example $n_r = 128$).

If the oscillation of failing and passing the tests is connected to the pseudo randomness, one would assume, that all 188 tests have the same probability of failure. An analysis has been conducted and the result is shown in the following table 4.3. The result is, that all test do fail at least once. It is not the case, that only specific tests fail. For the analysis of even more tests, it can be assumed, that the probability of failure (illustrated in the last column) will be more or less the same for all tests.

[5] refer to the test-case on a random input to the STS on the enclosed CD

Table 4.3: Statistical tests and their respective probability of failure

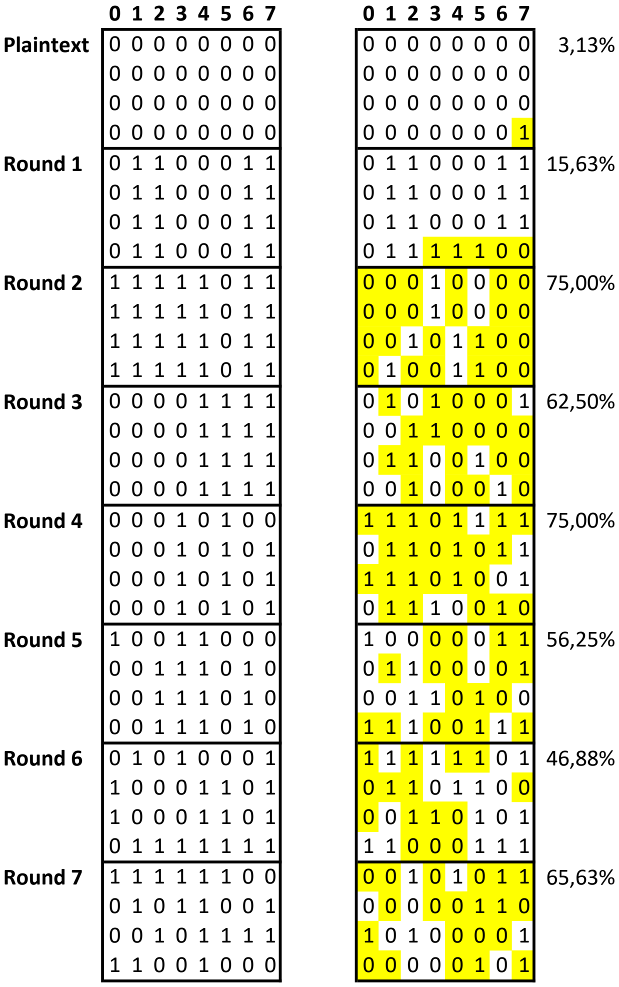
Test	Number of Tests	FAILED	Probability of Failure (%)
Frequency	51	4	7,84%
Block Frequency	51	4	7,84%
Cumulative Sums (forward)	51	5	9,80%
Cumulative Sums (backwards)	51	5	9,80%
Runs	51	4	7,84%
Longest Run of Ones	51	3	5,88%
Binary Matrix Rank	51	4	7,84%
Discrete Fourier Transform	51	7	13,73%
Non-overlapping Template Matching (148)	7548	481	6,370%
Overlapping Template Matching	51	3	5,88%
Maurer's Universal Statistical	51	4	7,84%
Approximate Entropy	51	5	9,80%
Random Excursions (8)	408	37	9,07%
Random Excursions Variant (18)	918	88	9,59%
Serial (2)	102	8	7,84%
Linear Complexity	51	1	1,96%

The goal is achieved, to prove that the output is still statistically independent from the input. The second goal to determine the number of rounds needed for a good confusion and diffusion during encryption with the rescaled AES could not be obtained with statistical testing. This is due to the oscillation of passing and failing of the tests (refer to table 4.2).

4.3.3 Analysis of the diffusion process

The rescaled AES version lacks one diffusion step compared to the 'standard' AES. To analyse the diffusion capabilities in more detail, two ciphertexts are compared with each other, where the difference between both is only one bit changed in the plaintext. This is often referred to as the *flipping* of one bit. A good diffusion capability of a cipher results in the flipping of around 50% of the bits statistically.

Figure 4.3 shows the results for 7 rounds of this flipping. On the left side, a zero-plaintext of 32 bit (32 zeros) and zero-key of 128 bit (128 zeroes) has been used to create a ciphertext. The output of the state up until round 7 is shown. On the right side, the least significant bit in the plaintext flips from 0 to 1, the key stays the same (128 zeroes). The difference between the state on the left and right is colored with yellow markers and gives an indication how fast diffusion is applied in the rescaled version of AES. This so called *avalanche effect* can be observed in figure 4.3 and results in flipping of multiple bits after just two rounds. The graphic shows that diffusion is still working for the rescaled AES. After more rounds, the percentage of flipped bits is expected to even out closer to 50%.



	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	
Plaintext	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3,13%
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	
Round 1	0	1	1	0	0	0	1	1	0	1	1	0	0	0	1	1	15,63%
	0	1	1	0	0	0	1	1	0	1	1	0	0	0	1	1	
	0	1	1	0	0	0	1	1	0	1	1	0	0	0	1	1	
	0	1	1	0	0	0	1	1	0	1	1	**1**	**1**	**1**	**0**	**0**	
Round 2	1	1	1	1	1	0	1	1	**0**	**0**	**0**	1	**0**	0	**0**	**0**	75,00%
	1	1	1	1	1	0	1	1	**0**	**0**	**0**	1	**0**	0	**0**	**0**	
	1	1	1	1	1	0	1	1	**0**	**0**	1	**0**	1	**1**	**0**	**0**	
	1	1	1	1	1	0	1	1	**0**	1	**0**	**0**	1	**1**	**0**	**0**	
Round 3	0	0	0	0	1	1	1	1	0	**1**	0	**1**	**0**	**0**	**0**	1	62,50%
	0	0	0	0	1	1	1	1	0	0	**1**	**1**	**0**	**0**	**0**	**0**	
	0	0	0	0	1	1	1	1	0	**1**	**1**	0	**0**	1	**0**	**0**	
	0	0	0	0	1	1	1	1	0	0	**1**	0	**0**	**0**	1	**0**	
Round 4	0	0	0	1	0	1	0	0	**1**	**1**	**1**	**0**	**1**	1	**1**	**1**	75,00%
	0	0	0	1	0	1	0	1	0	**1**	**1**	**0**	**1**	**0**	**1**	1	
	0	0	0	1	0	1	0	1	**1**	**1**	**1**	**0**	**1**	**0**	0	1	
	0	0	0	1	0	1	0	1	0	**1**	**1**	1	0	**0**	**1**	**0**	
Round 5	1	0	0	1	1	0	0	0	1	0	0	**0**	**0**	0	**1**	**1**	56,25%
	0	0	1	1	1	0	1	0	0	**1**	1	**0**	**0**	0	**0**	**1**	
	0	0	1	1	1	0	1	0	0	0	1	1	**0**	**1**	**0**	0	
	0	0	1	1	1	0	1	0	**1**	**1**	1	**0**	**0**	**1**	1	**1**	
Round 6	0	1	0	1	0	0	0	1	**1**	1	**1**	1	**1**	**1**	0	1	46,88%
	1	0	0	0	1	1	0	1	**0**	**1**	**1**	0	1	1	0	**0**	
	1	0	0	0	1	1	0	1	**0**	0	**1**	**1**	**0**	1	0	1	
	0	1	1	1	1	1	1	1	1	1	**0**	**0**	**0**	1	1	1	
Round 7	1	1	1	1	1	1	0	0	**0**	**0**	1	**0**	1	**0**	**1**	**1**	65,63%
	0	1	0	1	1	0	0	1	0	**0**	0	**0**	**0**	**1**	**1**	**0**	
	0	0	1	0	1	1	1	1	**1**	0	1	0	**0**	**0**	**0**	1	
	1	1	0	0	1	0	0	0	**0**	**0**	0	0	**0**	**1**	0	**1**	

Figure 4.3: Diffusion process for 7 rounds of encryption with the rescaled AES: yellow bits represent flipped bits, on the right side the percentage of flipped bits is indicated

4.4 Conclusion

In this chapter, AES has been rescaled to show, that SPN based ciphers are also able to implement a block length of 32 bit. It turns out, that a general statement on how many rounds are needed for the rescaled AES to be secure, can not be derived from statistical testing. Other methodologies to determine the adequate number of rounds, such as findings short cut attacks ought to bring better results (for example: determine the amount of rounds for which cryptanalysis does work and add a sufficient security margin). By the use of statistical testing, it was shown, that the rescaled AES

still applies sufficient confusion and diffusion.
As the key is XORed during the Key addition step, only one round key with 32 bit is used in every round. This may lead to problems with weak keys, if the key size is too long. A 128 bit key does need 4 rounds to be used during encryption (32 bit datapath). For the above presented version without changing the key schedule, a high number of rounds needs to be chosen in order to apply sufficient confusion and diffusion during encryption and to mitigate the threat of weak keys. A solution for this might be, to adapt the key schedule to the rescaled version of AES.

5 Analysis of 32 bit block ciphers from a security perspective

This chapter is analysing ciphers with a block length of 32 bit from a security perspective. The state of the art ciphers mentioned above and the rescaled AES can be used as examples.
Compared to the 'standard' block length of 128 bit, a block length of 32 bit results in changes of other characteristics of a cipher. These changes can lead to vulnerabilities, as, for example, an adversary may use them to break the cipher or extract the key. In the context of this thesis, the changes of characteristics are being referred to as *risk factors*. These are basic characteristics of a cipher and do not have to be vulnerabilities per definition[1].
During the creation of this thesis, it has been tried to do a classical risk analysis after ISO 27005. This approach failed out of two reasons:

- Though risks can be determined, they are more or less applicable to all block ciphers (such as "Break of a cipher", "Extracting the key"). The problem is to analyse these risks on an abstract level. As it is hard to determine risks in information security (likelihood * severity), it is even harder for ciphers, where only the block length of 32 bit is specified.
- The terminology does not fit properly for block ciphers. For example it is not suitable to mark a fundamental characteristic of a cipher as a vulnerability.

Therefore the ciphers with a block length of 32 bit are being analysed by their changed characteristics and how these characteristics can be exploited by an adversary.

5.1 Risk Factors

By reducing the characteristic block length to 32 bit, the following characteristics of a cipher do change or need to be adapted compared to block ciphers with a higher block length. A detailed analysis and the impact concerning the security of a cipher is given in the following sub chapters:

- **Block length** set to 32 bit
- **Number of permutations** decreases
- **Number of collisions** increases
- **Number of rounds** of a cipher needs to be adapted (usually decreases)
- **Key size** needs to be adapted (usually decreases)

[1] "Vulnerability: weakness of an asset or control that can be exploited by one or more threats" [36]

5.1.1 Block length

Today it is commonly accepted, that the block length of a block cipher must be at least 128 bit (refer to NIST AES call for tenders). On the one hand, if the block length is shorter than 128 bit, the security is decreased, because certain attacks can be performed better on a smaller block length. Such attacks can aim on statistical analysis resulting for example in text dictionary attacks and matching ciphertext attacks. This is specifically the case, if some plaintext-ciphertext pairs happen to come to be known to an adversary. When 2^n plaintext-ciphertext pairs are known to the adversary, the whole dictionary is compromised within that specific permutation. On the other hand, if the block length is too big (bigger than 256 bit), the complexity and therefore resource requirements grow with every bit. For implementing such big block lengths in practice, easy functions are necessary to not make the implementation take too many resources.

Therefore, the characteristic 'block length' can be seen as a trade off. If it is getting too small, problems arise, if it is getting to big, there are problems as well. Clearly, a block length of 32 bit is way shorter than the preferred 128 bit. This definitely is the weaker side of the trade off with partly negative side effects. These effects are explained in the following sub chapters.

5.1.2 Number of permutations

The number of permutations n_p is defined by $n_p = 2^n!$ where n is the block length. The following table 5.1 shows the number of permutation in relation to the block length.

Table 5.1: Comparison of block lengths and their respective number of permutations

Block length b	Number of permutations n_p
128	$\sim 10^{10^{40.11}}$
64	$\sim 10^{10^{20.54}}$
32	$\sim 2.22 * 10^{39507966976}$
16	$\sim 5.16 * 10^{287193}$
4	$\sim 2.09 * 10^{13}$
2	24

The smaller the block length, consequently the smaller the number of permutations. Note, that the number of permutations is calculated as a factorial. By reducing the block length, the number of permutations is reduced very quickly as can be seen in the table 5.1. The shorter the number of permutations, the chance of an adversary rises, to try every single permutation and therefore to brute force the ciphertext. This succeeds with a higher probability, if the plaintext actually consists of meaningful strings or does have high entropy. If the plaintext is a pseudo-random number (for example a product number), the adversary has limited chance to determine, if the brute-forced plaintext is in fact the correct plaintext. For this kind of attacks, the adversary does not need to

know the key. They are based on statistical approaches on the ciphertext and do not aim on breaking the cipher itself.
When implementing a lightweight cipher, it should be considered, that the computing power rises every year, raising the opportunity to brute-force a small block in a couple of years, which has a sufficient block length meeting today's conditions.
The full number of permutations usually is not used within a cipher, since the key space[2], defined by $K = 2^k$ is limiting the number of actually used permutations (for most block ciphers: $n_p >> K$), as a permutation is specified by the associated key (refer to chapter 2.3 "Block cipher"). This is because a permutation is specified by the associated key. Therefore it can be easier for an adversary to brute-force all the keys of a cipher, since this number is far less. An analysis of the characteristic key size is given in chapter 5.1.5 "Key Size".

5.1.3 Number of collisions

As the birthday paradox states, the probability of two people having birthday on the same day in a group of 23 people is bigger than 0.5. This idea can be applied to block ciphers as well [3, p. 341ff].
The encryption function of a block cipher should be a pseudo-random permutation. Therefore it might happen, that in an output stream, two ciphertext blocks encrypted under the same key occur. In this case, due to the pseudo-random permutation, the plaintexts for these two ciphertext-blocks are the same as well.
Following the birthday paradox, it is possible to find a collision well before all the possible ciphertexts have been send. This barrier is $2^{\frac{n}{2}}$ blocks of data encrypted under the same key and is commonly known as the *birthday bound.* For 32 bit block length it is just 256 kB[3]. The following table 5.2 shows this calculation in comparison to other block lengths and their respective birthday bound.

Table 5.2: Comparison of different block lengths and their respective birthday bound

Block length	Birthday Bound	Calculation
128	256 EByte	$\frac{2^{64}*128bit}{8*1024^6}$
64	32 GByte	$\frac{2^{32}*64bit}{8*1024^3}$
32	256 kByte	$\frac{2^{16}*32bit}{8*1024}$
16	512 Byte	$\frac{2^{8}*16bit}{8*1024}$
4	2 Byte	$\frac{2^{2}*4bit}{8}$
2	4 bit	2^1*2bit

[2] The set of all possible keys is called the *key space* [3, p. 5]

[3] this is the amount of data, an adversary needs to intercept, to have a chance of finding a collision. 256 kB can be send over the internet in one second (assuming a data rate of 2 MBit/s)

After $2^{\frac{n}{2}}$ blocks of data encrypted under the same key, according to the following formular, at least one collision is expected with a probability of $p \approx 0.3935$ where $D = 2^{\frac{n}{2}} = 2^{16}$ and $N = 2^n = 2^{32}$ [37]:

$$p = 1 - \prod_{i=0}^{D-1} \frac{N-i}{N} \approx 0,3935$$

The above stated equation is only depending on the block length itself. The occurrence of collisions therefore is only based upon the characteristic block length.

For a cipher not using any modes of operation or using modes of operation without chaining, such as ECB, collisions might appear with a high probability well before the birthday bound [37, p. 4]. This is enhanced by plaintexts containing a lot of redundancy. If this is known to an adversary, it can help to successfully attack a cipher.

When using modes of operation, attacks using the birthday bound are practical as well. As AES is designed "*to be secure even if the adversary obtains nearly* 2^{128} *input-output pairs*" [38, p. 126], most common modes of encryption are not proven to be secure after 2^{64} blocks of data being encrypted under the same key [38, p. 126]. Bhargavan and Leurent proposed an attack on the modes of operation (specifically the most used CBC and Counter (CTR) modes) for block ciphers, they called 'Sweet32'. For the example of the CBC mode, it is aimed at the occurrence of collisions of two ciphertext-blocks: $c_i = c_j$. Because the encrpytion function of an arbitrary block cipher is a permutation, a collision of two ciphertext-blocks means, that the inputs to the respective encryption function E are the same too. This results in the revealing of the XOR between two plaintext blocks [37]:

$$\begin{aligned} c_i &= c_j \qquad \text{with } c_i = E_i(m_i \oplus c_{i-1}) \text{ and } c_j = E_j(m_j \oplus c_{j-1}) \\ m_i \oplus c_{i-1} &= m_j \oplus c_{j-1} \\ m_i \oplus m_j &= c_{i-1} \oplus c_{j-1} \end{aligned}$$

The knowledge of an XOR between two plaintext blocks can be used by an adversary to mount certain attacks. After Bhargavan and Leurent, these attacks have a higher probability to succeed if the following conditions are true [37]:

- a fixed secret is sent repeatedly
- some fraction of the plaintext is known

In this case, there is a chance, that the XOR between the two plaintext blocks reveals the secret assuming the condition that one plaintext block contains the fixed secret and the other a known plaintext.

For the CTR mode, it is harder to achieve fractions of the plaintext. It can be done as well, when the adversary knows parts of the plaintext before (more information on that attack can be found here [39]).

Interestingly enough, the attacks described above, are based on the ciphertext itself, the attack can be mounted without the adversary requiring to learn the key. The only condition for these kind of attacks is, that the plaintexts must be encrypted under the same key.

5.1.4 Number of rounds

The number of rounds is an important security characteristic of a cipher. It strongly depends on the speed of the cipher to apply sufficient diffusion [40, p. 91].
The number of rounds for a cipher should be chosen carefully, considering the trade-off between efficiency and security. Whereas the goal is to use as minimum rounds as possible out of performance reasons, the resistance of a cipher against cryptanalytic attacks increases with the number of rounds. In cryptanalysis it is a common approach, to try to break a round-reduced version of a cipher in order to learn more about the full-round version (refer to chapter 3 "State of the art 32 bit block ciphers").
Comparing block ciphers with a variable block lengths in general leads to the following tendency: The higher the block length, the higher the number of rounds (for example refer to table 4.1). This is to ensure full confusion and diffusion on the whole datapath, depending on the block length. The previously presented table 3.6 also lists ciphers with a variable block length. These ciphers give a good indication on the change of the number of rounds compared to the block length.
Differences in the number of rounds result from the different underlying designs of the respective ciphers (Feistel, NLFSR, SPN).

5.1.5 Key size

The key in a block cipher is used to specify a single permutation out of the whole number of possible permutations. As discussed before, the number of permutations is strongly connected to the block length and decreases as the block length is decreased. Consequently, as the number of permutations is reduced, the key space should be reduced as well. Otherwise, the chance that two different keys are specifying a single permutation is higher.
Therefore a smaller key space is needed for block ciphers with 32 bit block length compared to 128 bit. This can be observed in the previously presented table 3.6 of 32 bit block ciphers. For example RC5, SIMON and SPECK support variable block lengths and variable key sizes. The tendency herein is: For smaller block lengths, smaller key sizes are proposed as well. An exception to this tendency marks the proposal of Rijndael, since every proposed block length can be combined with any proposed key size to form one of the 25 ciphers.
A reduced key space offers the possibility of exhaustive key search. This should be considered, when developing a new lightweight block cipher. In the list of previously presented ciphers shown in table 3.6, the key size does not fall under 64 bit. For the selection of the key size, the respective confidentiality level of the data to be encrypted should be considered, as well as an accumulation of data. Also, the prospective development of the computing power needs to be taken into account. It rises every year, making key spaces vulnerable for exhaustive key search in the future, if

the prospective development is not considered during development.
The key size being large enough to preclude exhaustive key search is "[...] a necessary, but usually not sufficient, condition for an encryption scheme to be secure" [2, p. 21]. When other ways to break the cipher are more efficient, the key size is not as crucial for the security of a cipher, as discussed before in chapters block length, number of permutations and number of collisions.

5.2 Mitigating measures

To mitigate the above evaluated risk factors for block ciphers with a block length of 32 bit, certain measures have to be considered and implemented within the specification of a product. In the following, these measures are identified and described in more detail.

1. **The Key needs to be changed before the birthday bound is reached (also referred to as *Rekeying*).**
   Considering the specific use-case, rekeying can be achieved by the use of different methods, which are heavily based on the amount of participants needing to know the new key. Procedures for rekeying can be divided in symmetric and asymmetric methods. Often it can be useful to establish a central key-server with the task to manage all the keys for all participants in a network.
   As shown above, the key needs to be changed well before the birthday bound is reached. For a 32 bit block cipher this threshold is well before 256 kByte of data.

2. **When modes of operation are being used, they should be selected considering their capability of providing security even when the birthday bound is reached**.
   To mitigate the problem with the birthday bound when using modes of operation, a lot of modes of operation have been proposed, providing security beyond the birthday bound. These modes of operation are not described here, as this topic leaves the focus of this thesis and as new modes are being proposed constantly.

3. **The number of rounds needs to be adapted** according to

   - the best known shortcut attack plus a sufficient security margin and
   - the efficient implementation of the lightweight block cipher making it suitable for resource constraint environments.

   This point refers to the previously described trade-off between security on the one side and performance on the other side. For example, as the security would benefit from a high number of rounds, the performance of the cipher would lack, making it impractical for the application in resource constraint environments.

4. **The key size should be chosen adequately to the security-level of the cipher**, considering the reduced key space based on the reduced number of permutations.

5.3 Conclusion

This chapter showed, that by reducing the characteristic block length to 32 bit, a lot of related characteristics change as well or need to be adapted. These changes can lead to crucial security issues (risk factors), where the security of a cipher can not be assured any longer under all possible conditions. To mitigate these issues, measures have to be implemented, resulting in the cipher being secure under certain conditions and for a designated use.

It is important, that for such a short block length, the characteristic key size is not as important as for block ciphers with a bigger block length. An adversary does not necessarily need to know the key to be able to extract the plaintext. As this can be mitigated for ciphers with a block length of 32 bit, it is expected to be more severe when the block length is reduced even further.

6 Conclusion and Outlook

This thesis focussed on block ciphers with a block length of 32 bit. As a conclusion the three basic questions from the introduction are being recalled with their respective short answers:

1. Are there 32 bit block ciphers already existing?
   As of today, state of the art 32 bit block ciphers are already existing. Some of them are broken and some of them are suspected to be vulnerable to cryptanalytic attacks. An overview can be found in table 3.6. Whereas most of the ciphers implement a Feistel- or NLFSR-based scheme, no cipher has been found implementing an SPN. This fact is used for the creation of an own cipher.

2. Are there other possibilities to create a block cipher with 32 bit?
   Within this thesis, it has been shown, that AES can be rescaled to a 32 bit block length. According to the output of specific tests, the rescaled version seems to be a good alternative to the other presented, state of the art ciphers. It applied good confusion and diffusion. Further cryptanalysis needs to be conducted in order to determine the minimal number of rounds for which no shortcut attacks exist. A certain security margin needs to be evaluated and added to this calculated number of rounds, bearing in mind the predicted development of computing power and the unpredictable development of new attacks to break a cipher. Also it might be necessary to adapt the key-schedule and the key size of the cipher as well, to make this cipher even more suitable for lightweight cryptography.

3. What kind of security issues derive from 32 bit block length and how can these be mitigated?
   The methodology proposed in ISO 27005 for risk assessment could not be applied to conduct a security analysis of block ciphers with a block length of 32 bit. If the block length is reduced to 32 bit, other characteristics of a cipher change or need to be adapted. Adapting specific characteristics may lead to the cipher being more vulnerable against certain kinds of attacks. Risk factors have been derived, resulting in the cipher being insecure if not properly mitigated.

This thesis showed, that 32 bit block ciphers can be implemented in a secure way. It should be always considered, that such a short block length should be only used for a specific use case, and not for 'normal', everyday cryptography. The data to be encrypted should be analysed and decided, which lightweight encryption scheme ought to be used, according to their respective level of confidentiality. For this analysis an aggregation of data must be considered, meaning, that a block of data may be of not much use to an adversary, but an aggregation of a lot of blocks might give in

fact valuable information.
For the use-case 'Museum' presented in the introduction, multiple possibilities are at hand. On the one side, a state of the art 32 bit cipher can be implemented. As some of the ciphers presented in this thesis are broken, the secure ones should be preferred (such as SIMON, SPECK or SIMECK). The cipher DLBCA is too recent and too little cryptanalysis was conducted on it. Another possibility could be, to implement the herein presented rescaled version of AES. It has the advantage, that its S-Box is developed by independent researches and has been subject to substantial public cryptanalysis.

Bibliography

[1] *Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths*, Std. NIST Special Publication 800-131A, November 2015. [Online]. Available: http://dx.doi.org/10.6028/NIST.SP.800-131AR1

[2] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*, ser. CRC Press series on discrete mathematics and its applications. Boca Raton, Fla. and London: CRC, 2001.

[3] C. Paar and J. Pelzl, *Kryptografie verständlich: Ein Lehrbuch für Studierende und Anwender*, ser. eXamen.press. Berlin and Heidelberg: Springer Vieweg, 2016. [Online]. Available: http://dx.doi.org/10.1007/978-3-662-49297-0

[4] K. Martin, *Everyday cryptography*. New York, NY, USA: Oxford University Press, 2017.

[5] E. Dubrova, M. Teslenko, and H. Tenhunen, "On Analysis and Synthesis of (N, K)-non-linear Feedback Shift Registers," in *Proceedings of the Conference on Design, Automation and Test in Europe*, ser. DATE '08. New York, NY, USA: ACM, 2008, pp. 1286–1291. [Online]. Available: http://doi.acm.org/10.1145/1403375.1403686

[6] C. E. Shannon, "Communication Theory of Secrecy Systems," 1948.

[7] G. V. Bard, *Algebraic cryptanalysis*. London: Springer, 2009. [Online]. Available: https://link.springer.com/content/pdf/10.1007%2F978-0-387-88757-9.pdf

[8] S. Indesteege, N. Keller, O. Dunkelman, E. Biham, and B. Preneel, "A Practical Attack on KeeLoq," in *Advances in Cryptology – EUROCRYPT 2008*, N. Smart, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 1–18.

[9] M. Kasper, T. Kasper, A. Moradi, and C. Paar, "Breaking KeeLoq in a Flash: On Extracting Keys at Lightning Speed," in *Progress in Cryptology – AFRICACRYPT 2009*, B. Preneel, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 403–420.

[10] R. L. Rivest, "The RC5 encryption algorithm," in *Fast Software Encryption*, B. Preneel, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 86–96.

[11] A. Biryukov and E. Kushilevitz, "Improved cryptanalysis of RC5," in *Advances in Cryptology — EUROCRYPT'98*, K. Nyberg, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 85–99.

[12] E. Hammond, “Skip32,” 1999. [Online]. Available: https://metacpan.org/pod/release/ESH/Crypt-Skip32-0.17/lib/Crypt/Skip32.pm

[13] F.-X. Standaert, G. Piret, N. Gershenfeld, and J.-J. Quisquater, “SEA: A Scalable Encryption Algorithm for Small Embedded Applications,” in *Smart Card Research and Advanced Applications*, J. Domingo-Ferrer, J. Posegga, and D. Schreckling, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 222–236.

[14] C. de Cannière, O. Dunkelman, and M. Knežević, “KATAN and KTANTAN — A Family of Small and Efficient Hardware-Oriented Block Ciphers,” in *Cryptographic Hardware and Embedded Systems - CHES 2009*, C. Clavier and K. Gaj, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 272–288.

[15] M. Ågren, “Some Instant- and Practical-Time Related-Key Attacks on KTANTAN32/48/64,” in *Selected Areas in Cryptography*, A. Miri and S. Vaudenay, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 213–229.

[16] B. Zhu and G. Gong, “Multidimensional meet-in-the-middle attack and its applications to KATAN32/48/64,” *Cryptography and Communications*, vol. 6, no. 4, pp. 313–333, 2014.

[17] D. Engels, M.-J. O. Saarinen, P. Schweitzer, and E. M. Smith, “The Hummingbird-2 Lightweight Authenticated Encryption Algorithm,” in *RFID. Security and Privacy*, A. Juels and C. Paar, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 19–31.

[18] M.-J. O. Saarinen, “Related-Key Attacks Against Full Hummingbird-2,” in *Fast Software Encryption*, S. Moriai, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 467–482.

[19] *Recommendation for Block Cipher Modes of Operation: Methods for Format Preserving Encryption*, Std. NIST Special Publication 800-38G, March 2016. [Online]. Available: http://dx.doi.org/10.6028/NIST.SP.800-38G

[20] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, “The SIMON and SPECK Families of Lightweight Block Ciphers,” Cryptology ePrint Archive, Report 2013/404, 2013. [Online]. Available: eprint.iacr.org/2013/404

[21] ——, “SIMON and SPECK: Block Ciphers for the Internet of Things,” Cryptology ePrint Archive, Report 2015/585, 2015. [Online]. Available: eprint.iacr.org/2015/585.pdf

[22] A. Biryukov, A. Roy, and V. Velichkov, “Differential Analysis of Block Ciphers SIMON and SPECK,” in *Fast Software Encryption*, C. Cid and C. Rechberger, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 546–570.

[23] K. Kondo, Y. Sasaki, Y. Todo, and T. Iwata, “On the Design Rationale of SIMON Block Cipher: Integral Attacks and Impossible Differential Attacks against SIMON Variants,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E101.A, no. 1, pp. 88–98, 2018.

[24] L. Song, Z. Huang, and Q. Yang, "Automatic Differential Analysis of ARX Block Ciphers with Application to SPECK and LEA," in *Information Security and Privacy*, J. K. Liu and R. Steinfeld, Eds. Cham: Springer International Publishing, 2016, pp. 379–394.

[25] G. Yang, B. Zhu, V. Suder, M. D. Aagaard, and G. Gong, "The Simeck Family of Lightweight Block Ciphers," Cryptology ePrint Archive, Report 2015/612, 2015. [Online]. Available: eprint.iacr.org/2015/612

[26] K. Zhang, J. Guan, B. Hu, and D. Lin, "Security evaluation on Simeck against zero-correlation linear cryptanalysis," *IET Information Security*, vol. 12, no. 1, pp. 87–93(6), 2018. [Online]. Available: http://digital-library.theiet.org/content/journals/10.1049/iet-ifs.2016.0503

[27] M. Zaheri and B. Sadeghiyan, "SMT-based Cube Attack on Simeck32/64," Cryptology ePrint Archive, Report 2018/130, 2018. [Online]. Available: eprint.iacr.org/2018/130

[28] B. Ryabko and A. Soskov, "The distinguishing attack on Speck, Simon, Simeck, HIGHT and LEA," Cryptology ePrint Archive, Report 2018/047, 2018.

[29] S. Salim, "Design 32-bit Lightweight Block Cipher Algorithm (DLBCA)," *International Journal of Computer Applications*, vol. 166, no. 8, pp. 17–20, 2017.

[30] E. Kobayashi, T. Suzaki, K. Minematsu, and S. Morioka, "TWINE: A Lightweight Block Cipher for Multiple Platforms," Researchgate, 2012. [Online]. Available: https://www.researchgate.net/publication/281267456_TWINE_A_Lightweight_Block_Cipher_for_Multiple_Platforms

[31] W. Wu and L. Zhang, "LBlock: A Lightweight Block Cipher," in *Applied Cryptography and Network Security*, J. Lopez and G. Tsudik, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 327–344.

[32] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoe, "PRESENT: An Ultra-Lightweight Block Cipher," in *Cryptographic Hardware and Embedded Systems - CHES 2007*, P. Paillier and I. Verbauwhede, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 450–466.

[33] Z. Gong, S. Nikova, and Y. W. Law, "KLEIN: A New Family of Lightweight Block Ciphers," in *RFID. Security and Privacy*, A. Juels and C. Paar, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 1–18.

[34] J. Daemen and V. Rijmen, *The design of Rijndael: AES–The Advanced Encryption Standard*. Berlin and Heidelberg and New York: Springer, 2002.

[35] *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*, Std. NIST Special Publication 800-22 Rev. 1a., April 2010. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-22r1a.pdf

[36] *Information technology – Security techniques – Information security management systems – Overview and vocabulary*, Std. ISO/IEC 27 000:2018, February 2018. [Online]. Available: https://www.iso.org/obp/ui/#iso:std:iso-iec:27000:ed-5:v1:en

[37] K. Bhargavan and G. Leurent, “On the Practical (In-)Security of 64-bit Block Ciphers: Collision Attacks on HTTP over TLS and OpenVPN,” 2016. [Online]. Available: https://sweet32.info/

[38] T. Iwata, “Authenticated Encryption Mode for Beyond the Birthday Bound Security,” in *Progress in Cryptology – AFRICACRYPT 2008*, S. Vaudenay, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 125–142.

[39] D. McGrew, “Impossible plaintext cryptanalysis and probable-plaintext collision attacks of 64-bit block cipher modes,” Cryptology ePrint Archive, Report 2012/623, 2012. [Online]. Available: https://eprint.iacr.org/2012/623

[40] R. Anderson, *Security Engineering: A guide to building dependable distributed systems*, ser. Wiley computer publishing. New York and Weinheim: Wiley, 2001.