White-box Cryptography Revisited: Space-Hard Ciphers

Andrey Bogdanov
Technical University of Denmark, Denmark
anbog@dtu.dk

Takanori Isobe
Sony Corporation, Japan
takanori.isobe@jp.sony.com

ABSTRACT

The need for software security in untrusted environments is ever increasing. White-box cryptography aims to ensure the security of cryptographic algorithms when the attacker has full access to their implementations. However, there is no secure white-box implementation of standard block ciphers such as DES and AES known to date: All published techniques have been practically broken.

In this paper, we revisit white-box cryptography and propose a family of *white-box secure block ciphers* SPACE with several novel features. The design of SPACE is such that the key-extraction security in the white box reduces to the well-studied problem of key recovery for block ciphers (AES in our example) in the standard black-box setting. Moreover, to mitigate code lifting, we introduce the notion of *space hardness*. It measures the difficulty of compressing the white-box implementation of a cipher, and quantifies security against code lifting by the amount of code that needs to be extracted from the implementation by a white-box attacker to maintain its functionality. SPACE includes several variants with different white-box code sizes. Therefore, it is applicable to a wide range of environments and use cases. One of the variants called N-SPACE can be implemented with different code sizes while keeping the cipher itself unchanged.

SPACE offers a high level of space hardness: It is difficult to find a compact but still functional representation of SPACE given its white-box implementation. This property has several useful consequences for applications. First, it gets more challenging for a DRM attacker (e.g. in a pay TV setting) to scale a code-lifting attack and to distribute the break. Moreover, this paves the way for *mass-surveillance resistant cryptography*: If a large proportion of users dedicates a significant part of their computers' storage (e.g. HDD) to white-box SPACE implementations, it will be much more complex or even infeasible for governmental agencies to deal with the keys of all users simultaneously due to the limited storage available, forcing them to focus on targeted attacks instead. This consequence is especially important given Snowden's revelations on the extent of the mass surveillance practice by NSA and GCHQ. Finally, the usage of SPACE ciphers can mitigate the damage of having malware in security-critical systems such as networks processing top-secret data: As those are typically insulated from the Internet, the capacity of the communication channel from inside to outside the system is often limited, making it infeasible for Trojans to transmit the necessary key material.

CCS'15, October 12–16, 2015, Denver, Colorado, USA.

ACM 978-1-4503-3832-5/15/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2810103.2813699.

Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: Security and Protection

Keywords

white-box cryptography; space-hard cipher; code lifting; decomposition; key extraction; DRM; pay TV; mass surveillance; Trojans; malware

1. INTRODUCTION

1.1 Background

White-box cryptography, introduced by Chow et al. in 2002, aims to protect software implementations of cryptographic algorithms in untrusted environments [13, 14]. An increasing number of applications are emerging that require substantial security in purely software environments, e.g. set-top boxes, PCs, tablets and smartphones, even if hardware-assisted security mechanisms are available such as the ARM TrustZone. Here, the attacker has full control over the execution environment of a cryptographic algorithm, both in static and dynamic ways by decompiler and debugger tools, e.g. IDA Pro and IL DASM.

The major goal of white-box cryptography is to protect the *confidentiality of secret keys* in such a white-box environment. In addition, *code lifting* is a threat [13, 42, 33], where the attacker attempts to isolate the program code from the implementation environment and directly uses the code itself as a larger key, instead of finding the underlying compact secret key.

Given the spread of software-only applications in embedded as well as desktop and server systems, it comes as no surprise that white-box cryptography receives a lot of attention from industry, especially in pay TV and other DRM settings. As it inherently addresses resistance to malware and Trojans, white-box cryptography will find more and more

applications in banking and other security-critical settings as well.

1.2 Previous Work

White-box implementations of DES and AES were first proposed by Chow et al. in [13, 14]. Their approach was to find a representation of the algorithm as a network of look-ups in randomized and key-dependent tables. In the wake of these seminal papers, several further variants of white-box implementations for DES and AES were proposed [12, 44, 23, 26]. However, all published white-box solutions for DES and AES to date have been *practically* broken by key extraction and table-decomposition attacks [3, 43, 35, 34, 25].

Security against key extraction and code lifting in those white-box implementations is based upon *external encodings*, which are randomly drawn bijections added to the input and output of the target block cipher. Such a block cipher becomes an *encoded* variant of the original algorithm, which is inappropriate when standard encryption schemes are required for interoperability, e.g. on the standard DRM platform Marlin [27] or in banking.

Dedicated white-box block ciphers have recently been proposed by Biryukov et al. in [4]. They are based on the ASASA structure that consists of two secret nonlinear layers (S) and three secret affine layers (A), with affine and nonlinear layers interleaved. Similarly to [13, 14], the white-box implementation of ASASA uses table look-ups. The security of ASASA against key extraction in the white-box setting relies on the hardness of the decomposition problem for ASASA. To estimate the security against code lifting attacks without external encodings, the work [4] introduces a security requirement of *weak white-box security*: It should be computationally hard for an attacker to find any compact equivalent representation of the cipher, i.e. table decomposition in the white-box environment should be computationally hard. Indeed, this makes code lifting attacks difficult in terms of the amount of data that needs to be extracted from the white-box environment. Unfortunately, efficient decomposition attacks on ASASA have been proposed [19, 32, 22].

To summarize the design approaches so far, the security against key recovery and table-decomposition attacks of most existing white-box implementations relies on the hardness of the decomposition problem given multiple secret nonlinear and linear layers. It is pointed out in [4] that the white-box implementations of AES and DES in [13, 14] can be considered as the 3-layer ASA, which is much weaker than the 5-layer ASASA. However, the decomposition of secret nonlinear and linear layers is a relatively new problem with only a few papers [7, 11, 40] dedicated to its study. Indeed, although more layers make the construction more secure, recent cryptanalysis [6] suggests that even as many as 9 layers (SASASASAS) are susceptible to attacks. Thus, the assurance on the security of $(\mathrm{AS})^i$ against decomposition is yet to be provided.

In this paper, we take a different approach and base the decomposition security of our ciphers in the white box on the problem of key recovery for block ciphers in the standard black-box setting.

1.3 Our Contributions

In this paper, we propose a family of white-box secure block ciphers. Our ciphers are designed to satisfy the following properties:

White-box security is based on black-box security: In white-box environments, the security of our constructions relies on the well-studied problem of key recovery for block ciphers such as AES. Thus, *key extraction and table-decomposition attacks are computationally infeasible as long as the underlying block cipher is secure against key recovery attacks in the standard black-box setting.*

Space hardness: To quantitatively evaluate the difficulty of code lifting attacks, we introduce a security requirement called (M, Z)*-space hardness* which is a generalization of the weak white-box security notion of [4]. The notion of (M, Z)-space hardness allows us to claim that *if the amount of code to be isolated from the white-box implementation by an attacker is less than M, a construction is secure against code lifting. Namely, the success probability that the code correctly encrypts (or decrypts) a random input is less than 2^{-Z}.* Indeed, weak white-box security corresponds to the case of $(M, 0)$-space hardness.

Furthermore, even if the attacker succeeds in code lifting, the property of space hardness discourages him from illegally distributing the code due to its large size, as it is infeasible to find any compact implementation unless the secret key is known.

No external encoding: To be applicable to the wide range of situations and use cases, our ciphers do not require any external operations such as external encodings for their white-box security.

Variable white-box implementation size: In order to provide a high degree of compatibility across platforms and resource restrictions, our constructions include several variants with different but fixed code sizes as well as a variant with variable code sizes while keeping the cipher itself unchanged.

Our family of white-box secure block ciphers consists of two types of constructions: SPACE and N-SPACE.

SPACE includes four variants: SPACE-8, -16, -24 and -32, which are implementable in different but fixed sizes of code, ranging from a few KB to some GB. The table sizes of SPACE-8, -16, -24 and -32 are suited for L1/L2 cache (e.g. 32 KB to 256 KB), L3 cache (e.g. 8 MB), RAM (e.g. a few GB) and HDD (e.g. many GB), respectively. Moreover, we propose 4-SPACE as an example of N-SPACE. It offers four implementation variants with different code sizes from a few KB to several GB, while keeping the cipher itself unchanged: 4-SPACE-8, -16, -24 and -32.

Our constructions offer implementation advantages over known white-box AES implementations and are competitive to $(\mathrm{AS})^i$ structures in white-box environments. In particular, at the comparable levels of (M,Z)-space hardness, the white-box implementations of SPACE-16, -24, and -32 require exactly the same number of table look-ups as ASASA-1, -2, and -3, respectively, see Table 4.

Since the underlying internal block cipher can be freely chosen depending on the user requirements, a wide range of

implementation properties in the black box can be attained. If one chooses a software-oriented lightweight block cipher such as PRIDE [1] and SIMON/SPECK [2] as the underlying block cipher, an implementation with very low RAM and code size requirements is possible [18]. With AES inside, the black-box performance can be optimized by using bit-sliced implementations or AES-NI.

1.4 Related Work: Memory Hardness

The concept of *memory hardness* was proposed in the context of password hashing [37, 21, 5]: It forces the attacker to consume a large amount of memory while computing a target function. The purpose is to prevent efficient parallel brute-force attacks by dedicated password-cracking hardware and GPUs.

As opposed to that, the goal of (M, Z)-space hardness is to mitigate the copying of functionality. It states a bound on the data required to be extracted from the white-box environment for successfully processing a random input with probability of more than 2^{-Z}. In other words, it aims to effectively increase the key size to M.

2. ATTACK MODELS

In this paper, we deal with two attack models: *black-box* model and *white-box model*.

2.1 In the Black Box

The black-box model is a classical attack model in the field of symmetric-key cryptography.

2.1.1 Attacker's Abilities

This model assumes that the attacker is able to access inputs and outputs of the cipher with known- or chosen-plaintexts or ciphertexts. Adaptive queries can be allowed.

2.1.2 Security Requirements

As the attacker aims to recover the secret key or to distinguish the block cipher from a randomly drawn permutation, some standard security requirements in the black-box model can be informally summarized as follows.

Key recovery security: It is computationally hard to recover the key of the block cipher.

Distinguishing security: It is computationally hard to distinguish the block cipher from a randomly drawn permutation.

The hardness of a key recovery is evaluated by the time complexity accompanied by data and memory complexities of finding the key. For instance, a 128-bit security implies a time complexity of at least 2^{128} encryptions. The complexity of a distinguishing attack is formally evaluated by the number of queries necessary.

2.2 In the White Box

The white-box model originates from the seminal results by Chow et al. [13, 14].

2.2.1 Attacker's Abilities

This model assumes that the attacker has full control over the execution environment of a cipher, both in static and dynamic ways with the aid of arbitrary trace execution, examination of sub-results and keys in memory, insertion of break-points, modification of internal variables, and many more.

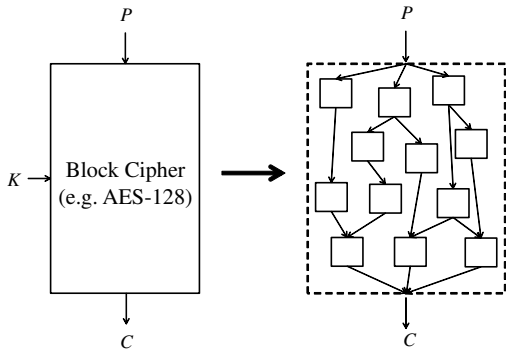

Figure 1: Table-based white-box implementations: The key K is scrambled by a network of table look-ups

2.2.2 Security Requirements

The main goal of the white-box attacker is to extract the secret key given the full access to the cipher's implementation and its internals. Therefore, typical security requirements in the white-box setting are as follows.

Key extraction security: It is computationally hard to extract the secret key of the block cipher.

Code lifting security: Instead of a secret key, the attacker can directly use the implementation itself as a larger effective key. In particular, he can isolate the program code where the key is embedded in order to copy the functionality of encryption/decryption routines and to utilize it in a stand-alone manner. In some aspects, this is also referred to as global deduction by De Mulder [33].

If a code lifting attack succeeds, the attacker gets the advantage which is almost the same as key extraction, i.e. he can encrypt/decrypt any plaintext/ciphertext. Unless a public-key primitive is used or external encodings are involved, it is challenging to completely prevent code lifting attacks. To evaluate the difficulty of this attack, the notions of weak white-box security and incompressibility have been introduced in [33, 4]. The details of these will be given in the next section. In this paper, we adopt a more general security notion: *space hardness*.

3. KNOWN WHITE-BOX TECHNIQUES

The white-box techniques published so far can be divided into two groups: white-box implementations of existing block ciphers such as DES and AES on the one hand, and dedicated designs of block ciphers for the white-box environment on the other.

3.1 White-box Implementation of DES/AES

White-box implementations of DES and AES were first proposed by Chow et al. in [13, 14]. Their approach was to represent the block cipher E_K as a network of look-ups in randomized key-dependent tables, see Fig. 1. The key value

is masked by random variables and is scrambled into the tables.

Each table is protected by applying secret invertible encodings before and after the table. For example, the white-box implementation of AES [14] employs secret nonlinear and linear components as secret encodings to protect the tables. In order to preserve the functionality of the cipher, the output encoding g^i of the i-th table is the inverse function of the input encoding f^{i+1} of the $(i+1)$-th input table, that is, $(g^i)^{-1} = f^{i+1}$.

Finally, external encodings IN and OUT are added to the input and output of the block cipher E_K. Therefore, the action of the composite transform is $OUT \circ E_K \circ IN^{-1}$, where IN and OUT are secret bijections. The purpose of external encodings is to protect the tables of the first and last rounds. This also mitigates code lifting by hiding the actual block cipher E_K between encodings.

For example, in the DRM setting, the external encoding IN is first applied on the server side after encrypting the content P and returns $C' = IN \circ E_K(P)$. Then, the DRM client software, running on the user's device, decrypts C' to $P' = OUT \circ D_K \circ IN^{-1}(C')$ and outputs the encoded content $P' = OUT(P)$, where D_K is the inverse of the block cipher E_K. The remaining encoding is removed in the user's content player which is placed closer to the playback device to obtain the original content P, see Fig. 2.

Following the papers by Chow et al. [13, 14], several further variants of white-box implementations of DES and AES were proposed [12, 44, 23, 26].

3.1.1 Problems with Security

All published white-box implementations of DES and AES are *practically* broken [3, 43, 35, 34, 25]. In addition to these dedicated attacks, Michiels et al. [31] proposed a generic table decomposition attack on a wide class of white-box implementations of SPN ciphers.

As mentioned in [30], since AES and DES were designed with the black-box security in mind, it seems difficult to provide white-box security at the same time. This is still an open problem.

3.1.2 On External Encodings

The crucial drawback of this approach is the usage of external encodings. Due to external encodings at the input and output of the block cipher E_K, the algorithm becomes an *encoded* variant of the cipher, i.e. a *different* cipher. This can be inappropriate when interoperability is necessary as it is the case for the standard DRM platform Marlin which specifies AES-128 as the content protection algorithm [27]. Banking is also an application where interoperability plays an important role.

Furthermore, to obtain original plaintexts after white-box decryption, the additional decoding operation has to be performed in a *secure* environment. The natural question arises as to why the entire decryption is not performed in this secure environment in the first place. Therefore, as also mentioned in [42, 33], the applications of white-box implementations with external encodings are mainly restricted to proprietary DRM settings.

However, if the external encodings are removed to avoid the issues mentioned above, the white-box implementation becomes much weaker [14] because the first- and last-round tables become directly accessible to the attacker.

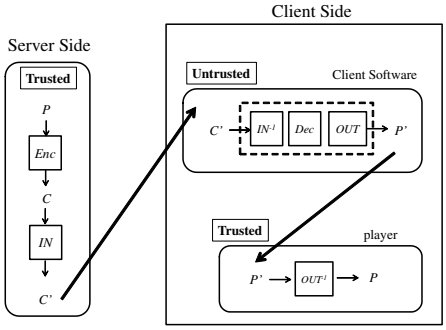

Figure 2: External encodings for DRM: IN and OUT are external encodings. Enc is the block cipher E_K and Dec is its inverse D_K

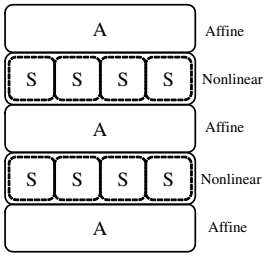

Figure 3: ASASA construction: 5 layers of interleaved secret affine and secret nonlinear (S-boxes) operations

3.2 Dedicated Cipher: ASASA

Dedicated block ciphers for white-box environments were proposed by Biryukov et al. in [4]. They are based on the ASASA structure consisting of two secret nonlinear layers (S-boxes, S) and three secret affine layers (A) in the interleaved order, see Fig. 3. The security of the block ciphers against key recovery attacks relies on the hardness of decomposing the ASASA structure. Unlike the white-box implementations of AES and DES, the ASASA structure does not require external encodings. To estimate the security of ASASA against code lifting attacks, the notion of *weak white-box security* was introduced.

Definition 1 (Weak White-box Security [4]). *The function F is an T-secure weak white-box implementation of E_K if it is computationally hard to obtain an equivalent key of size less than T given full access to F.*

In other words, it should be computationally hard for an attacker to find any compact equivalent function which is smaller than T. Accordingly, an attacker requires code of size T to copy the functionality of the cipher completely. Weak white-box security enables the estimation of the difficulty of code lifting by the amount of data needed to be extracted from the white-box environment. This property is also called *incompressibility* by De Mulder [33].

Biryukov et al. also define strong white-box security. This property corresponds to MQ-problems used in public-key cryptography and is related to one-wayness as defined by De Mulder [33].

3.2.1 Security Issues

Decomposition and key recovery attacks on ASASA structures have recently been proposed [19, 32, 22]. The security of constructions based on multiple secret nonlinear and linear layers is still to be explored and seems hard to evaluate, despite several cryptanalytic efforts [7, 11, 40]. Recent attacks point out that even the 9-layer variant SASASASAS does not offer a sufficient security level [6]. The assurance on the security of $(\mathrm{AS})^i$ against decomposition attacks is yet to be provided.

4. OUR DESIGN GOALS

In this section, we outline our design goals for a new family of white-box secure block ciphers.

4.1 Security

To quantitatively evaluate the difficulty of code lifting without relying on external encodings, one could take the notion of weak white-box security [4], which can assess the amount of data required to copy the *full* functionality. However, the white-box security of a cipher when the size of the available code (table) is less than M is unclear.

To reveal the tradeoff between the data available and attacker's advantage, we introduce a novel security notion coined (M, Z)*-space hardness*.

Definition 2 ((M, Z)-space hardness). *The implementation of a block cipher E_K is (M, Z)-space hard if it is infeasible to encrypt (decrypt) any randomly drawn plaintext (ciphertext) with probability of more than 2^{-Z} given any code (table) of size less than M.*

(M, Z)-space hardness enables us to estimate the code size M to be isolated from white-box environments to encrypt (decrypt) any plaintext (ciphertext) with a success probability larger than 2^{-Z} as well as to derive more fine-grained security claims. Weak white-box security [4] can be seen as a special case of (M,Z)-space hardness and corresponds to $(M, 0)$-space hardness.

Security requirements in the black- and white-box environments are given as follows.

Security in the black box: The cipher should be secure against *key recovery attacks* and *distinguishing attacks*, i.e. there are no attacks more efficient than generic attacks such as brute force.

Security in the white box: The cipher should be security against *key extraction attacks*, and mitigation of *code lifting attacks* in terms of **(M,Z)-space hardness**: An attacker needs to obtain codes (tables) whose size is larger than M to compute any plaintext or ciphertext with probability larger than 2^{-Z}.

4.2 Functionality

To be applicable to a wide range of situations and use cases, the cipher should not require any additional functions such as external encodings.

4.3 Performance

In both black-box and white-box environments, the performance of the cipher should be competitive to known primitives such as whitebox AES [14, 44] and ASASA constructions [4].

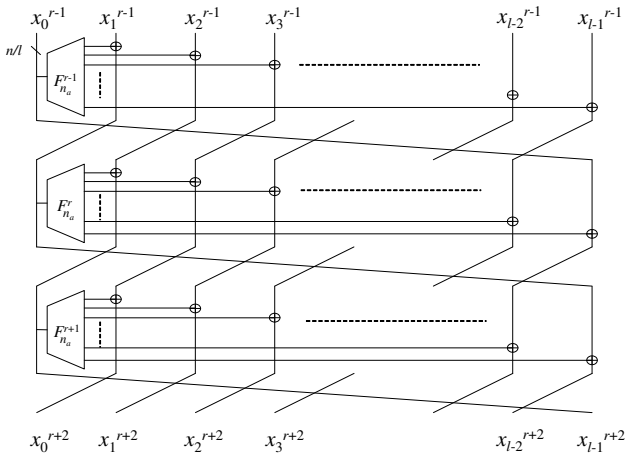

Figure 4: Ciphers with fixed code size: SPACE

To make our ciphers implementable in multiple settings including those with restricted resources, we provide several variants with different code sizes (SPACE), and a variant enabling multiple code sizes while keeping the cipher itself unchanged (N-SPACE).

5. SPACE: FIXED SPACE

This section proposes a block cipher called SPACE, attaining our design goals in the black-box and white-box settings as well as offering several variants with different but fixed code sizes.

5.1 The Design

SPACE is an ℓ-line target-heavy generalized Feistel network [38] which encrypts an n-bit plaintext under a k-bit secret key to an n-bit ciphertext, where the size of each line is n_a $(= n/\ell)$ bits, as shown in Fig. 4.

Let the n-bit state of round r be $X^r = \{x_0^r, x_1^r, \ldots, x_{\ell-1}^r\}$, $x_i^r \in \{0,1\}^{n_a}$. X^0 and X^R are a plaintext and a ciphertext, respectively, where R is the number of rounds. Each round updates the state as:

$$X^{r+1} = \left(F_{n_a}^r(x_0^r) \oplus (x_1^r||x_2^r||\ldots||x_{\ell-1}^r)\right)||x_0^r,$$

where $F_{n_a}^r : \{0,1\}^{n_a} \to \{0,1\}^{n_b}$. Here $||$ denotes the concatenation, and $n_b = n - n_a$. The function $F_{n_a}^r(x)$ is defined as

$$F_{n_a}^r(x) = (msb_{n_b}(E_K(C_0||x))) \oplus r,$$

where E_K is a block cipher with n-bit block and k-bit key, $msb_u(x)$ selects the most significant u bits of x, and C_0 is an $(n_b\ (= n - n_a))$-bit binary zero value. The last XOR of r plays the role of a round constant (see Fig. 5).

Let $F'_{n_a}(x) = msb_{n_b}(E_K(C_0||x))$. It is an n_a-bit to n_b-bit function. Each round updates the state by looking up the leftmost line value in the table for $F'_{n_a}(x)$, adding the constant r to the result to compute $F_{n_a}^r(x)$, XORing it to the other lines, and rotating the lines by one position to the left.

In the white-box environment, $F'_{n_a}(x)$ is implemented by table look-ups. SPACE has only one table of $F'_{n_a}(x)$ that is reused in each round. This single-table implementation makes the evaluation of (M, Z)-space hardness easier.

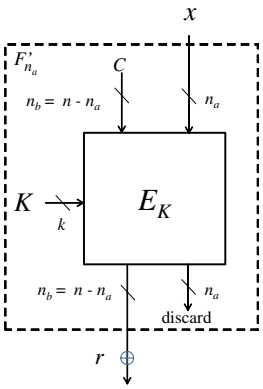

Figure 5: F-function: $F_{n_a}(x) = F'_{n_a}(x) \oplus r$

We instantiate the SPACE family with four concrete block ciphers for $k = 128$ and AES-128 as the underlying block cipher E_K:

- SPACE-$(8, R)$: $n = 128$, $\ell = 16$, $n_a = 8$, $F_8^r : \{0,1\}^8 \to \{0,1\}^{120}$
- SPACE-$(16, R)$: $n = 128$, $\ell = 8$, $n_a = 16$, $F_{16}^r : \{0,1\}^{16} \to \{0,1\}^{112}$
- SPACE-$(24, R)$ [*1] : $n = 128$, $\ell = 16$, $n_a = 24$, $F_{24}^r : \{0,1\}^{24} \to \{0,1\}^{104}$
- SPACE-$(32, R)$: $n = 128$, $\ell = 4$, $n_a = 32$, $F_{32}^r : \{0,1\}^{32} \to \{0,1\}^{96}$

5.2 Feistel

We aim to show security in the white box as defined in Section 4.1. Hence our approach is to construct the table $F'_{n_a}(x)$ from a well-studied standard block cipher such as AES by constraining the plaintext and truncating the ciphertext. Then, the hardness of extracting the key from the table and finding a compact description of the table in the white-box model relies on the difficulty of key recovery for the underlying block cipher in the black-box model.

Since restricting the input and output of any secure underlying block cipher is unlikely to deliver a permutation, a Feistel-type construction is a natural candidate. We note that the SPN structure adopted by the ASASA construction does require secret permutations as building blocks and, therefore, cannot be based on the truncation of a standard block cipher with a 128-bit block length such as AES-128 directly.

We also considered type-1, -2, -3 generalized Feistel construction and source-heavy construction as the underlying construction and opted for the target-heavy Feistel construction for performance reasons.

5.3 Security in the White Box

5.3.1 Key Extraction

In the white-box model, the attacker is able to fully access inputs and outputs of tables in any round. To extract the key from the table $F'_{n_a}(x) = msb_{n_b}(E_K(C_0||x))$ means to recover the key of E_K in the black-box model, with plaintexts from a restricted space and truncated ciphertexts. The underlying block cipher E_K of our SPACE instantiation is AES-128, for which no efficient key recovery attack has been published so far despite considerable cryptanalytic efforts over 15 years [9, 17, 20]. Thus, key extraction is computationally hard in the white-box model as long as the underlying block cipher is secure against key recovery attacks in the black-box model.

More formally, the advantage of the key extraction in the white-box model for SPACE, $\mathrm{Adv}_{\text{KE-WB}}$, is upper-bounded by the advantage of the key recovery for the underlying block cipher in the black-box model, $\mathrm{Adv}_{\text{KR-BB}}$:

$$\mathrm{Adv}_{\text{KE-WB}} \leq \mathrm{Adv}_{\text{KR-BB}}.$$

[*1] Only in this variant, each round updates the state as $X^{r+1} = (x_1^r||x_2^r)||\big(F_{24}^r(x_0^r||x_1^r||x_2^r) \oplus (x_3^r||x_4^r||\ldots||x_{\ell-1}^r)\big)||x_0^r$ to keep the 128-bit block size.

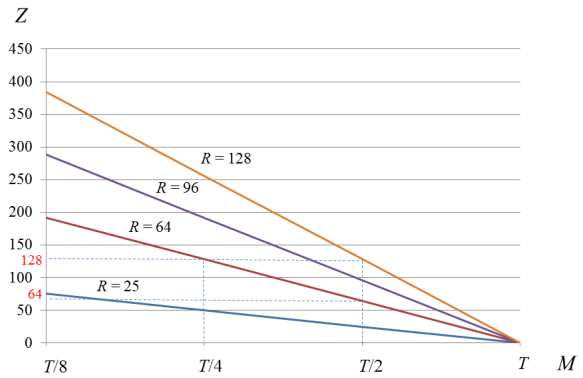

Figure 6: A compression attack on SPACE with $R \in \{25, 64, 96, 128\}$ in terms of (M, Z)-space hardness, with $T = (2^{n_a} \times n_b)$ bits

5.3.2 Code Lifting: Space Hardness

As the attacker is unable to compute $E_K(C_0||x)$ without the knowledge of K if the underlying block cipher E_K is secure, it is computationally hard to find any compact representation of $E_K(C_0||x)$. The table of $E_K(C_0||x)$ consists of 2^{n_a} entries of n_b bits each, and the total table (code) size T is estimated as $(2^{n_a} \times n_b)$ bits. In other words, this provides weak whitebox security at the level of $(2^{n_a} \times n_b)$ bits [4].

Let us consider the case where a part of the table is leaked, i.e. $i \leq 2^{n_a}$ entries of table are extracted by the attacker, where the leaked-table size M is $(i \times n_b)$ bits. The probability that a random input of the table is among the extracted subset of entries is estimated as $i/2^{n_a}$ $(= (i \times n_b)/(2^{n_a} \times n_b))$. Thus, given a random plaintext/ciphertext, the corresponding output after R rounds can be computed with i entries of the table with a probability of about $(i/(2^{n_a}))^R$. Fig. 6 shows this relation between M and Z in terms of (M, Z)-space hardness for SPACE with $R \in \{25, 64, 96, 128\}$. This evaluation is a basic *white-box compression attack* and its results should be seen as an upper bound on the actual space hardness.

5.4 Security in the Black Box

5.4.1 Key Recovery

In the black-box model, the attacker is unable to directly access the inputs and outputs of the internal tables $F'_{n_a}(x) = msb_{n_b}(E_K(C_0||x))$. Hence, a key recovery for SPACE in the black-box setting is at least as complex as a key recovery for the underlying block cipher. Thus, in the black box, dedicated distinguishing attacks with possible subsequent table recovery are of more concern than key recovery attacks.

5.4.2 Generic Attacks

Generic attacks on target-heavy generalized Feistel constructions were proposed in [36, 41]. None of these properties spans more than 47, 23 and 11 rounds of SPACE-8, -16 and -32, respectively.

5.4.3 Differential Cryptanalysis

Here we analyze the differential properties of an n_a-bit to n_b-bit function $F'_{n_a}(x) : \{0,1\}^{n_a} \rightarrow \{0,1\}^{n_b(=n-n_a)}$.

DEFINITION 3. *The cardinality of a differential $N(a,b)$ for a function f is the number of pairs with input difference a that have output difference b:*

$$N(a,b) = \#\{(v,u)|u \oplus v = a \text{ and } f(v) \oplus f(u) = b\}$$

The distribution of $N(a,b)$ over functions has been shown to be binomial for sufficiently large n_a and n_b [16, 8].

LEMMA 1. *[16] For a non-trivial differential (a,b) with fixed a and b, the distribution of $N(a,b)$ over n_a-bit to n_b-bit functions is binomial:*

$$Pr(N(a,b) = i) = (2^{-n_b})^i (1 - 2^{-n_b})^{2^{n_a-1}-i} \binom{2^{n_a-1}}{i}.$$

Assuming that the differentials over a fixed randomly drawn permutation have a similar distribution of expected $N(a,b)$ and using the proof techniques of Theorem 2 in [8], we obtain the following for a random function:

THEOREM 1. *Assuming that the distribution of $N(a,b)$ for a function $F'_{n_a}(x)$ is binomial (Lemma 1), the probability q_B that $N(a,b)$ is at most B over all non-trivial values of a and b can be lower-bounded by*

$$q_B > \left(1 - \frac{(2^{n_a-1} \cdot 2^{-n_b})^{B+1}}{(B+1)!}\right)^{2^{n+1}}.$$

Proof. The number of combinations of a and b is estimated as $2^{n\ (=n_a+n_b)}$, hence:

$$q_B = (1 - Pr(N(a,b) > B))^{2^n},$$

where we have:

$$\begin{aligned} Pr(N(a,b) > B) &= \sum_{j=B+1}^{2^{n_a-1}} (2^{-n_b})^i (1 - 2^{-n_b})^{2^{n_a-1}-i} \binom{2^{n_a-1}}{i} \\ &< \sum_{j=B+1}^{2^{n_a-1}} (2^{-n_b})^i \binom{2^{n_a-1}}{i} \\ &< \sum_{j=B+1}^{2^{n_a-1}} (2^{-n_b})^i \frac{(2^{n_a-1})^i}{(B+1)!} \\ &< 2 \cdot \frac{(2^{n_a-1} \cdot 2^{-n_b})^{B+1}}{(B+1)!}. \quad \square \end{aligned}$$

Table 1: Lower bound on q_B: The probability that $N(a,b)$ is at most B over all non-trivial values of a and b for $B = 1, 2$ in F'_8, F'_{16}, F'_{24} and F'_{32}

	q_1	q_2	q_3
F'_8	$1 - 2^{-96}$	$1 - 2^{-209}$	$1 - 2^{-323}$
F'_{16}	$1 - 2^{-64}$	$1 - 2^{-161}$	$1 - 2^{-259}$
F'_{24}	$1 - 2^{-32}$	$1 - 2^{-113}$	$1 - 2^{-195}$
F'_{32}	$1 - 2^{-0.66}$	$1 - 2^{-65}$	$1 - 2^{-131}$

Table 1 shows q_B for F'_8, F'_{16}, F'_{24} and F'_{32}. The differential probability of $F'(x)$ is estimated as $B/2^{n/\ell}$. Since q_2 and q_3 are very close to 1 in F'_8 and F'_{16}, and F'_{24} and F'_{32}, respectively, we assume the maximum differential probability of SPACE-8, -16, -24 and -32 to be 2^{-7} ($= 2/2^8$), 2^{-15} ($= 2/2^{16}$), $2^{-22.4}$ ($= 3/2^{24}$) and $2^{-30.4}$ ($= 3/2^{32}$), respectively.

Our search for the minimum number of differentially active F-functions shows that SPACE-8, -16, -24 and -32 have at least 17, 9, 6 and 5 active F-functions after 150, 44, 32 and 14 rounds.

5.4.4 Linear Cryptanalysis

Now we analyze the linear properties of the function $F'_{n_a}(x)$: $\{0,1\}^{n_a} \rightarrow \{0,1\}^{n_b(=n-n_a)}$.

Given an input mask α and an output mask β, $\alpha \in \{0,1\}^{n_a}$ and $\beta \in \{0,1\}^{n_b}$, the correlation of a linear approximation (α, β) for a function f: $\{0,1\}^{n_a} \rightarrow \{0,1\}^{n_b}$ is defined as

$$\begin{aligned} Cor &= 2^{-n_a}[\#\{x \in \{0,1\}^{n_a} | \alpha \cdot x \oplus \beta \cdot f(x) = 0\} - \\ &\quad \#\{x \in \{0,1\}^{n_a} | \alpha \cdot x \oplus \beta \cdot f(x) = 1\}. \end{aligned}$$

The linear probability LP of (α, β) is defined as Cor^2. LP of $F'_{n_a}(x)$ is assumed to be normally distributed [16], using

COROLLARY 1. *[16] The linear probability LP of a non-trivial linear approximation over n_a-bit to n_b-bit functions with $n \geq 5$ has mean $\mu(LP) = 2^{-n_a}$ and variance $\sigma^2(LP) = 2 \times 2^{-2n_a}$.*

Therefore, the linear probability LP of $F'_{n_a}(x)$ with a fixed key is lower than $2^{-n_a} + 10\sigma$ with probability $1 - 2^{-148}$. The value of $2^{-n} + 10\sigma$ for the F-functions of SPACE-8, -16, -24 and -32 is estimated as $2^{-4.5}$, $2^{-12.5}$, $2^{-20.5}$, and $2^{-28.6}$, correspondingly. In our evaluation, we assume the maximum linear probabilities of the F-functions to be 2^{-4}, 2^{-12}, 2^{-20} and 2^{-28}, respectively.

Our search for the minimum number of linearly active F-functions in SPACE-8, -16, -24 and -32 shows that there are at least 32, 11, 7 and 5 active F-functions after 33, 12, 10 and 6 rounds.

5.4.5 Impossible Differential Cryptanalysis

In SPACE-8, -16, -24 and -32, any input bit non-linearly affects all state bits after at least 17, 9, 15 and 5 rounds, respectively. Following the miss-in-the-middle approach, after 34 ($= 17 \times 2$), 18 ($= 9 \times 2$), 30 ($= 15 \times 2$), and 10 ($= 5 \times 2$) rounds, we have not found any useful impossible differentials for the respective variants.

Table 2: Summary of security evaluation for SPACE in the black box: Round numbers needed to resist attacks.

	G	F	D	L	ID	I
SPACE-$(8, R)$	47	17	150	33	34	19
SPACE-$(16, R)$	23	9	44	12	18	12
SPACE-$(24, R)$	-	15	32	10	30	17
SPACE-$(32, R)$	11	5	14	6	10	10
4-SPACE-(R)	-	17	48	16	34	19

G : Generic attack [36, 41], F : Full Diffusion
D : Differential attack, L : Linear attack
ID : Impossible differential attack, I : Integral attack

5.4.6 Other Attacks

From the results by Suzaki et al. [39], it follows that there are no useful integral distinguishers after 19, 12, 17 and 10 rounds of SPACE-8, -16, -24 and -32. We have also considered further attacks including slide, higher order differential, truncated differential, and algebraic attacks. The details of this evaluation are omitted due to the page limitation. Table 2 shows a summary of our security evaluation for SPACE-8, -16, -24 and -32.

5.5 Recommended Numbers of Rounds

We conservatively recommend to choose the number R of rounds to be equal to twice the number of rounds resisting the basic white-box compression attack of Section 5.3.2 at the level of $(T/4, 128)$-space hardness or twice the number of rounds covered by the best black-box distinguisher of Section 5.4, whichever is higher.

The recommended variants are SPACE-$(8, 300)$, $(16, 128)$, $(24, 128)$ and $(32, 128)$. In the white box, the security claim is the key extraction security at the level of 128 bits and $(T/4, 128)$-space hardness. In the black box, we claim the classical security of a 128-bit block cipher with a 128-bit key.

Note that lower numbers of rounds can be used for most variants if a more aggressive space hardness level is acceptable to the user.

5.6 Implementation Issues

5.6.1 Implementation in the White Box

In a white-box implementation, $F'_{n_a}(x)$ is implemented by table look-ups. This is by far the most expensive operation and the performance of encryption/decryption can be estimated by the number of table look-ups (TL) along with the table sizes. Such a table consists of 2^{n_a} entries of n_b bits each, and the table size is $T = (2^{n_a} \times n_b)$ bits.

Table 3 shows the performance and the table size for each recommended variant, where L1-TL, L3-TL, RAM-TL and HDD-TL denote table accesses to L1/L2 cache, L3 cache, RAM and HDD, respectively. The sizes of tables in SPACE-$(8, 300)$, -$(16, 128)$, -$(24, 128)$ and -$(32, 128)$ are suited for L1/L2 cache (e.g. 32 KB to 256 KB), L3 cache (e.g. 8 MB), RAM (e.g. a few GB) and HDD (e.g. more than 10 GB), respectively. Assuming that one random access to the table stored in L1/L2 cache, L3 cache, RAM and HDD takes 5, 30, 100 and 1000 cycles, the white-box performance of SPACE-$(8, 300)$, -$(16, 128)$, -$(24, 128)$ and -$(32, 128)$ is roughly estimated as 93, 240, 800 and 8000 cycles per byte, respectively.

Table 3: Performance of SPACE with recommended round numbers

	Performance	T
SPACE-$(8, 300)$	300 L1-TL	3.84 KB
SPACE-$(16, 128)$	128 L3-TL	918 KB
SPACE-$(24, 128)$	128 RAM-TL	218 MB
SPACE-$(32, 128)$	128 HDD-TL	51.5 GB
AES (Chow et al.) [14]	3008 L3-TL	752 KB
AES (Xiao-Lai) [44]	80 RAM-TL	20.5 MB
AES (Black-box) [15]	160 L1-TL	4 KB

For example, SPACE-$(16, 128)$ and the broken white-box AES by Chow et al. [14] has almost the same code size (suitable for L3 cache), but SPACE-$(16, 128)$ is 23 times faster.

A comparison with ASASA constructions will be provided in Section 5.6.3.

5.6.2 Implementation in the Black Box

In the black-box environment, a compact implementation is possible for the key owner by decomposing the table for $F'_{n_a}(x)$. Performance is then estimated by the number of internal block cipher calls. For SPACE-$(8, 300)$, -$(16, 128)$, -$(24, 128)$ and -$(32, 128)$, it is 300, 128, 128 and 128 calls, respectively.

A wide range of implementations is thinkable under the freedom of choice of the underlying block cipher for SPACE. With a lightweight block cipher such as PRIDE [1] and SIMON/SPECK [2] inside, implementations with very low RAM and code size requirements are possible [18].

If AES-128 is chosen, the implementation can be speeded up using the AES-NI instructions. For example, on Intel Haswell, if SPACE is used in a parallel mode such as CTR, one F-function call would require at most 16 clock cycles. This yields performance estimates of at most 300, 128, 128 and 128 cycles per byte for SPACE-$(8, 300)$, -$(16, 128)$, -$(24, 128)$ and -$(32, 128)$, respectively.

While the white-box implementation of SPACE-8 is faster than its black-box implementation, black-box implementations of SPACE-$(16, 128)$, -$(24, 128)$ and -$(32, 128)$ with AES-NI are much faster than those of white-box implementations.

5.6.3 Tradeoff between Performance and Security

The performance of our constructions depends on the number of rounds R, which in turn is mostly determined by the desired level of (M, Z)-space hardness. There is an efficient tradeoff between R and (M, Z)-space hardness. Table 4 shows the comparison between SPACE and (broken) ASASA at similar levels of space hardness, where the maximum space hardness stands for the complexity of the basic compression attack of Section 5.3.2.

At the same level of space hardness, the white-box implementation of SPACE offers exactly the same performance as the ASASA constructions. By no means do those numbers $(T/4, 128)$ and $(T/35, 128)$ claim white-box security for the SPACE and ASASA variants in question.

The performance and compressibility of the white-box implementation of $(\mathrm{AS})^i$ do not change even if more layers are added to improve the security. Hence, the figures for ASASA-1, 2 and 3 in Table 4 remain valid for any $(\mathrm{AS})^i$.

Table 4: Comparison of SPACE and ASASA at similar space-hardness levels

	Performance	T	Maximum space hardness
SPACE-$(16, 64)$	64 L3-TL	918 KB	$(T/4, 128)$
SPACE-$(24, 64)$	64 RAM-TL	218 MB	$(T/4, 128)$
SPACE-$(32, 25)$	25 HDD-TL	51.5 GB	$(T/35, 128)$
ASASA-1 [4]	64 L3-TL	8 MB	$(T/4, 128)$
ASASA-2 [4]	64 RAM-TL	384 MB	$(T/4, 128)$
ASASA-3 [4]	25 HDD-TL	20 GB	$(T/35, 128)$

ASASA-1 : S-layer consists of 8 × 16-bit
ASASA-2 : S-layer consists of 24-bit + 6 × 16-bit + 8-bit
ASASA-3 : S-layer consists of 4 × 28-bit + 16-bit

On the other hand, in the black-box implementation of $(\mathrm{AS})^i$, the number of cycles increases linearly with the number of rounds. ASASA-1, -2 and -3 do not benefit from AES-NI unlike SPACE with AES-128. However, since the S-boxes are 8- or 10-bit, optimizations for L1 cache are possible for all variants in the black-box environment.

5.7 Strong Space Hardness

Here we discuss an extension of the notion of space hardness that we call *strong space hardness*:

Definition 4 (Strong (M, Z)-space hardness). *An implementation of a block cipher E_K is (M, Z)-space hard if it is infeasible to obtain a valid plaintext/ciphertext pair with probability higher than 2^{-Z} given the code (table) of size less than M.*

The difference to the notion of space hardness is that the attacker tries to find any valid input/output pair now, not merely a valid output for a given randomly drawn input. Strong space hardness is relevant to message authentication codes in the context of forgeries.

Let us try to come up with some compression attacks against strong space hardness for SPACE. If each entry of ℓ consecutive tables of $F^r, \dots F^{r+\ell}$ is chosen, states $X^r = \{x_0^r, \dots, x_{\ell-1}^r\}$ and $X^{r+\ell+1} = \{x_0^{r+\ell+1}, \dots, x_{\ell-1}^{r+\ell+1}\}$ are determined. The number of start states consisting of ℓ consecutive tables is estimated as $(2^i)^\ell$, where 2^i is the number of known table entries. Thus, the probability of finding a valid pair with $i \leq 2^{n_a}$ table entries is estimated as $(i/(2^{n_a}))^{R-\ell} \times (2^i)^\ell$, and the time complexity is estimated as $(2^i)^\ell$. If $i < 2^{7.35}, 2^{14}$ and 2^{30} for SPACE-$(8, 300)$, SPACE-$(16, 128)$ and -$(32, 128)$, the probability becomes less than 2^{-128}, where the code size is 2.45 KB, 230 KB and 12.9 GB, respectively. For SPACE-$(24, 128)$, six consecutive tables are enough to determine states X^r and X^{r+7}. If $i < 2^{22.95}$, the probability becomes less than 2^{-128}, where the code size is 105 MB.

This evaluation is a straightforward approach to find a valid plaintext/ciphertext. More sophisticated attacks seem possible. Thus, we explicitly do not make any security claims with respect to strong space hardness for SPACE-$(8, 300)$, SPACE-$(16, 128)$, -$(24, 128)$ and -$(32, 128)$, but the above values can be considered as upper bounds on the level of their strong $(M, 128)$-space hardness.

5.8 Other Constructions

There are other possible constructions suitable to attain space hardness. For instance, the following design is thinkable, which combines AES in counter mode and the idea of secret sharing among N_k instances of AES.

Let S_i be

$$S_i = \bigoplus_{j=0}^{N_k} AES_{K_j}(i||IV),$$

where $AES_{Kj}()$ denotes an encryption using AES-128 with the 128-bit key K_j, i is a 64-bit variable and IV is a 64-bit nonce. Given a 128-bit plaintext P_i, the encryption is performed as $C_i = P_i \oplus S_i$, in the stream cipher fashion. This simple construction achieves $(16 \times N_k$ bytes, 128)-space hardness due to fact that it is infeasible to compute S_i without the knowledge of all but few key bits. However, it requires N_k AES calls for encrypting a 128-bit plaintext, which makes its use impractical: For example, in order to achieve (218 MB, 128)-space hardness, $N_k = 13,625,000$ AES calls are required.

6. N-SPACE: VARIABLE SPACE

This section presents our second block cipher. It is called N-SPACE and allows implementations with multiple variable code sizes while keeping the cipher itself unchanged.

6.1 The Design

N-SPACE is an ℓ-line target-heavy generalized Feistel network with N *different* sizes of F-functions. It encrypts an n-bit plaintext under N k-bit secret keys to an n-bit ciphertext as shown in Fig. 7.

Let the n-bit state of round r be $X^r = \{x_0^r, x_1^r, \dots, x_{\ell-1}^r\}$, $x_i^r \in \{0,1\}^{n/\ell}$. Each round updates the state as follows. If $(r \bmod N) = j$:

$$X^{r+1} = X'^r || \big(F^r_{(j+1)n/\ell}(X'^r) \oplus (x_{j+1}^r || \dots || x_{\ell-1}^r)\big) || x_0^r.$$

where $X'^r = (x_0^r || \dots || x_j^r)$.

The instantiation of the cipher with $n = 128$, $\ell = 16, N = 4$, and R rounds is called 4-SPACE-(R) whose round transforms are specified as follows.

If $(r \bmod N) = 0$:

$$X^{r+1} = \big(F_8^r(x_0^r) \oplus (x_1^r || \dots || x_{\ell-1}^r)\big) || x_0^r.$$

If $(r \bmod N) = 1$:

$$X^{r+1} = x_1^r || \big(F_{16}^r(x_0^r || x_1^r) \oplus (x_2^r || \dots || x_{\ell-1}^r)\big) || x_0^r.$$

If $(r \bmod N) = 2$:

$$X^{r+1} = x_1^r || x_2^r || \big(F_{24}^r(x_0^r || x_1^r || x_2^r) \oplus (x_3^r || \dots || x_{\ell-1}^r)\big) || x_0^r.$$

If $(r \bmod N) = 3$:

$$X^{r+1} = x_1^r || x_2^r || x_3^r || \big(F_{32}^r(x_0^r || \dots || x_3^r) \oplus (x_4^r || \dots || x_{\ell-1}^r)\big) || x_0^r.$$

The four F-functions $F_8^r(x)$, $F_{16}^r(x)$, $F_{24}^r(x)$, and $F_{32}^r(x)$ depend on four 128-bit keys, K_1, K_2 and K_3 and K_4, respectively. The F-functions of 4-SPACE are based on AES-128 exactly in the same way as the $F_{n_a}^r$ of SPACE in Section 5.1.

The particularity of 4-SPACE-(R) is that it uses four differently sized F-functions $F_8^r(x)$, $F_{16}^r(x)$, $F_{24}^r(x)$ and $F_{32}^r(x)$. In the white-box implementation, depending on user requirements for the code size, we can choose which of the

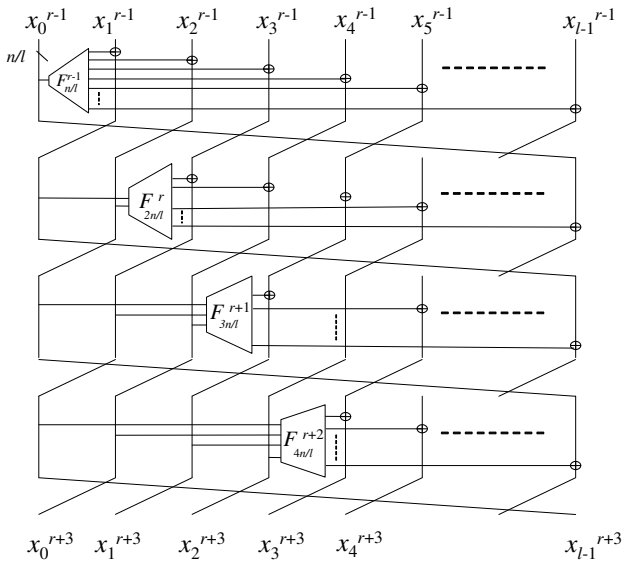

Figure 7: Ciphers with variable space: N-SPACE

functions $F_8^r(x)$, $F_{16}^r(x)$, $F_{24}^r(x)$ and $F_{32}^r(x)$ to implement by table look-ups. We define four implementation variants:

- 4-SPACE-(R)-32 : All four functions are implementation by table look-ups.
- 4-SPACE-(R)-24 : $F_8'(x)$, $F_{16}'(x)$ and $F_{24}'(x)$ are implemented by table look-ups.
- 4-SPACE-(R)-16 : $F_8'(x)$ and $F_{16}'(x)$ are implemented by table look-ups.
- 4-SPACE-(R)-8 : Only $F_8'(x)$ is implemented by table look-ups.

The other respective F-functions are implemented by block cipher calls with the corresponding keys. 4-SPACE-(R) requires four 128-bit keys: K_1, K_2, K_3 and K_4. One can derive those from a 128-bit master key K using a generic key derivation function [24, 29].

6.2 Security in the White Box

6.2.1 Key Extraction

As for SPACE, it is hard to extract the key from the table for an F-function as long as the underlying block cipher E_K is secure against key recovery. In 4-SPACE-(R)-24, -16 and -8, the attacker can directly observe K_4, (K_3, K_4) and (K_2, K_3, K_4) from the white-box implementation. However, (K_3, K_2, K_1), (K_2, K_1) and K_1 are hard to extract, respectively.

6.2.2 Code Lifting

Since the attacker is unable to compute the F-function without knowing of its key K_i, it is infeasible to find a compact representation of any of the variants, as at least one F-function is implemented by table look-ups.

Let us mount a white-box compression attack and consider the case where i entries of the largest table are isolated by the attacker. In each implementation variant, it is the largest table that will dominate the overall table size. All other tables are assumed to be available to the attacker in full. The probability that a random input of the table is among the known entries is $i/(2^m)$, where m is 8, 16, 24 and 32 for 4-SPACE-(R)-8, -16, -24 and -32, respectively. Given any random plaintext or ciphertext, the corresponding output after R rounds can be computed by using i table entries with probability of $(i/(2^m))^{(R/4)}$. This corresponds to $(i \times (n-m)$ bits, $(i/(2^m))^{(R/4)})$-space hardness.

6.3 Security in the Black Box

6.3.1 Differential Cryptanalysis

According to Theorem 1, we assume that the maximum differential probabilities of F_8, F_{16}, F_{24} and F_{32} to be 2^{-7} ($= 2/2^8$), 2^{-15} ($= 2/2^{16}$), $2^{-22.4}$ ($= 3/2^{24}$) and $2^{-30.4}$ ($= 3/2^{32}$), respectively. Our search for the minimum number of differentially active F-functions with the above values of the differential probabilities shows that the probability of any differential characteristic is expected to become less than 2^{-128} after 48 rounds.

6.3.2 Linear Cryptanalysis

Similarly to the evaluation for SPACE, we assume that linear probabilities of F_8, F_{16}, F_{24} and F_{32} are upper-bounded by 2^{-4}, 2^{-12} , 2^{-20} and 2^{-28}, respectively. Our search for the minimum number of linearly active F-functions with the above values of the linear probabilities yields that the probability of the best linear characteristic becomes less than 2^{-128} after 16 rounds.

6.3.3 Impossible Differential Cryptanalysis

Any input bit nonlinearly affects all state bits after 17 rounds. With the miss-in-the-middle approach, we have not found any useful impossible differentials after about 34 rounds.

6.3.4 Other Attacks

According to the results by Suzaki et al. in [39], after 19 rounds, there is no useful integral distinguisher either. We also considered other attacks, and we expect that none of them works significantly better than the previously mentioned attacks.

6.4 Recommended Number of Rounds

As for SPACE, we recommend to choose the number of rounds such that the basic white-box compression attack of Section 6.2.2 covers at most half of the rounds at the level of $(T/4, 128)$-space hardness and the best black-box property in Section 6.3 covers at most half of the rounds. For the instantiation of 4-SPACE at hand, following this guideline gives 512 rounds. A less conservative claim for the space hardness may be acceptable for many users resulting in a significantly lower number of rounds.

For the recommended number $R = 512$ of rounds, the claimed white-box security for 4-SPACE is $(T/4, 128)$-space hardness, see also Table 5. The claimed black-box security is that of a 128-bit block cipher with a 128-bit key. We do not claim any security against combined black- and white-box attackers.

Table 5: Performance and table sizes for the four 4-SPACE implementation variants and recommended round number: All of them offer exactly the same functionality

	Performance	T
4-SPACE-(512)-8	128 L1-TL+384 BC	3.84 KB
4-SPACE-(512)-16	128(L1-TL+L3-TL)+ 256 BC	918 KB
4-SPACE-(512)-24	128(L1-TL+L3-TL)+ 128 RAM-TL+128 BC	218 MB
4-SPACE-(512)-32	128(L1-TL+L3-TL)+ 128(RAM-TL+HDD-TL)	51.5 GB

6.5 Implementation Issues

Table 5 demonstrates the performance and table sizes for each of the four implementation variants for 4-SPACE. The sizes of 4-SPACE-(R)-8, -16, -24 and -32 are suited for L1/L2 cache, L3 cache, RAM and HDD, respectively. All implementation variants offer exactly the same functionality.

Under the rough assumption that a table access to L1/L2 cache, L3 cache, RAM and HDD costs 5, 30, 100, and 1000 cycles, respectively, and that an AES-128 encryption with AES-NI takes 16 cycles (in a parallel mode of operation), the performance of 4-SPACE-(512)-8, -16, -24 and -32 is approximately evaluated as 424, 536, 1208 and 9080 cycles per byte.

In the black box, implementations without tables are possible by decomposing the tables. The performance is then estimated by the number of block function calls. A combination of F-functions implemented with table look-ups and AES-NI is beneficial.

7. CONCLUSIONS

We have opened up a new direction for white-box cryptography, by introducing the idea that white-box security can rely on key recovery problems for well-analyzed block ciphers in the standard black-box setting and by proposing the new security notion of (M, Z)-space hardness. This enables us to demonstrate security against key extraction, table decomposition and code lifting attacks in the white-box environment, which have been the crucial limitation of the published techniques.

As an example, we design the family of block ciphers SPACE. It includes four variants with different but fixed code sizes, and a variant N-SPACE with variable code sizes while keeping the cipher itself unchanged.

Acknowledgments

We would like to thank Bart Preneel for his highly valuable and constructive feedback, which helped us to improve the technical and editorial quality of the paper. Moreover, we would like to thank the anonymous referees for their insightful comments. Among others, we are grateful to the referee who proposed the secret-sharing based construction described in Section 5.8.

8. REFERENCES

[1] Martin R. Albrecht, Benedikt Driessen, Elif Bilge Kavun, Gregor Leander, Christof Paar, and Tolga Yalçin. Block Ciphers - Focus on the Linear Layer (feat. PRIDE). In *Advances in Cryptology - CRYPTO 2014*, LNCS, Vol. 8616, pages 57–76, 2014.

[2] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. Simon and Speck: Block Ciphers for Internet of Things. NIST Lightweight Cryptography Workshop 2015, 2015.

[3] Olivier Billet, Henri Gilbert, and Charaf Ech-Chatbi. Cryptanalysis of a White Box AES Implementation. In *Selected Areas in Cryptography - SAC 2004*, LNCS, Vol. 3357, pages 227–240, 2004.

[4] Alex Biryukov, Charles Bouillaguet, and Dmitry Khovratovich. Cryptographic Schemes Based on the ASASA Structure: Black-Box, White-Box, and Public-Key (Extended Abstract). In *Advances in Cryptology - ASIACRYPT 2014*, LNCS, Vol. 8873, pages 63–84, 2014.

[5] Alex Biryukov, Daniel Dinu, and Dmitry Khovratovich. Fast and Tradeoff-Resilient Memory-Hard Functions for Cryptocurrencies and Password Hashing. Cryptology ePrint Archive, Report 2015/430, 2015.

[6] Alex Biryukov and Dmitry Khovratovich. Decomposition attack on SASASASAS. Cryptology ePrint Archive, Report 2015/646, 2015.

[7] Alex Biryukov and Adi Shamir. Structural Cryptanalysis of SASAS. *J. Cryptology*, Vol. 23(4), pages 505–518, 2010.

[8] Céline Blondeau, Andrey Bogdanov, and Gregor Leander. Bounds in Shallows and in Miseries. In *Advances in Cryptology - CRYPTO 2013*, LNCS, Vol. 8042, pages 204–221, 2013.

[9] Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. Biclique Cryptanalysis of the Full AES. In *Advances in Cryptology - ASIACRYPT 2011*, LNCS, Vol. 7073, pages 344–371, 2011.

[10] Andrey Bogdanov, Florian Mendel, Francesco Regazzoni, Vincent Rijmen, and Elmar Tischhauser. ALE: AES-Based Lightweight Authenticated Encryption. In *Fast Software Encryption - FSE 2013*, LNCS, Vol. 8424, pages 447–466, 2013.

[11] Julia Borghoff, Lars R. Knudsen, Gregor Leander, and Søren S. Thomsen. Slender-Set Differential Cryptanalysis. *J. Cryptology*, Vol. 26(1), pages 11–38, 2013.

[12] Julien Bringer, Hervé Chabanne, and Emmanuelle Dottax. White Box Cryptography: Another Attempt. *IACR Cryptology ePrint Archive*, 2006:468, 2006.

[13] Stanley Chow, Philip A. Eisen, Harold Johnson, and Paul C. van Oorschot. A White-Box DES Implementation for DRM Applications. In *Security and Privacy in Digital Rights Management, ACM CCS-9 Workshop*, pages 1–15, 2002.

[14] Stanley Chow, Philip A. Eisen, Harold Johnson, and Paul C. van Oorschot. White-Box Cryptography and an AES Implementation. In *Selected Areas in Cryptography - SAC 2002*, LNCS, Vol. 2595, pages 250–270, 2002.

[15] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002.

[16] Joan Daemen and Vincent Rijmen. Probability distributions of correlation and differentials in block ciphers. *J. Mathematical Cryptology*, Vol. 1(3), pages 221–242, 2007.

[17] Patrick Derbez, Pierre-Alain Fouque, and Jérémy Jean. Improved Key Recovery Attacks on Reduced-Round AES in the Single-Key Setting. In *Advances in Cryptology - EUROCRYPT 2013*, LNCS, Vol. 7881, pages 371–387, 2013.

[18] Daniel Dinu, Yann Le Corre, Dmitry Khovratovich, Leo Perrin, Johann Grosschadl, and Alex Biryukov. Triathlon of Lightweight Block Ciphers for the Internet of Things. NIST Lightweight Cryptography Workshop 2015, 2015.

[19] Itai Dinur, Orr Dunkelman, Thorsten Kranz, and Gregor Leander. Decomposing the ASASA Block Cipher Construction. Cryptology ePrint Archive, Report 2015/507, 2015.

[20] Orr Dunkelman, Nathan Keller, and Adi Shamir. Improved Single-Key Attacks on 8-Round AES-192 and AES-256. J. Cryptology, Vol. 28(3), pages 397–422, 2015.

[21] Christian Forler, Stefan Lucks, and Jakob Wenzel. Memory-Demanding Password Scrambling. In *Advances in Cryptology - ASIACRYPT 2014*, LNCS, Vol. 8874, pages 289–305, 2014.

[22] Henri Gilbert, Jérôme Plût, and Joana Treger. Key-Recovery Attack on the ASASA Cryptosystem with Expanding S-Boxes. In *Advances in Cryptology - CRYPTO 2015*, LNCS, Vol. 9215, pages 475–490, 2015.

[23] Mohamed Karroumi. Protecting White-Box AES with Dual Ciphers. In *Information Security and Cryptology - ICISC 2010*, LNCS, Vol. 6829, pages 278–291, 2010.

[24] Hugo Krawczyk. Cryptographic extraction and key derivation: The HKDF scheme. In *Advances in Cryptology - CRYPTO 2010*, LNCS, Vol. 6223, pages 631–648, 2010.

[25] Tancrède Lepoint, Matthieu Rivain, Yoni De Mulder, Peter Roelse, and Bart Preneel. Two Attacks on a White-Box AES Implementation. In *Selected Areas in Cryptography - SAC 2013*, LNCS, Vol. 8282, pages 265–285, 2013.

[26] Hamilton E. Link and William D. Neumann. Clarifying Obfuscation: Improving the Security of White-Box DES. In *International Symposium on Information Technology: Coding and Computing - ITCC 2005*, IEEE Computer Society, Vol. 1, pages 679–684, 2005.

[27] Marlin Developer Community. Marlin architecture overview. http://www.marlin-community.com, 2011.

[28] NIST Special Publication 800-38B. Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication. 2005.

[29] NIST Special Publication (SP) 800-108. Recommendation for Key Derivation Using Pseudorandom Functions. 2009.

[30] Wil Michiels. Opportunities in White-Box Cryptography. *IEEE Security & Privacy*, vol. 8(1), pages 64–67, 2010.

[31] Wil Michiels, Paul Gorissen, and Henk D. L. Hollmann. Cryptanalysis of a Generic Class of White-Box Implementations. In *Selected Areas in Cryptography - SAC 2008*, LNCS, Vol. 7707, pages 414–428, 2008.

[32] Brice Minaud, Patrick Derbez, Pierre-Alain Fouque, and Pierre Karpman. Key-Recovery Attacks on ASASA. Cryptology ePrint Archive, Report 2015/516, 2015.

[33] Yoni De Mulder. White-Box Cryptography: Analysis of White-Box AES Implementations. PhD thesis, KU Leuven, 2014.

[34] Yoni De Mulder, Peter Roelse, and Bart Preneel. Cryptanalysis of the Xiao - Lai White-Box AES Implementation. In *Selected Areas in Cryptography - SAC 2012*, LNCS, Vol. 7707, pages 34–49, 2012.

[35] Yoni De Mulder, Brecht Wyseur, and Bart Preneel. Cryptanalysis of a Perturbated White-Box AES Implementation. In *Progress in Cryptology - INDOCRYPT 2010*, LNCS, Vol. 6498, pages 292–310, 2010.

[36] Jacques Patarin, Valérie Nachef, and Côme Berbain. Generic Attacks on Unbalanced Feistel Schemes with Expanding Functions. In *Advances in Cryptology - ASIACRYPT 2007*, LNCS, Vol. 4833, pages 325–341, 2007.

[37] Colin Percival. Stronger Key Derivation via Sequential Memory-Hard Functions. presented at BSDCan'09, 2009.

[38] Bruce Schneier and John Kelsey. Unbalanced Feistel Networks and Block Cipher Design. In *Fast Software Encryption - FSE 1996*, LNCS, Vol. 1039, pages 121–144, 1996.

[39] Tomoyasu Suzaki and Kazuhiko Minematsu. Improving the Generalized Feistel. In *Fast Software Encryption - FSE 2010*, LNCS, Vol. 6147, pages 19–39, 2010.

[40] Tyge Tiessen, Lars R. Knudsen, Stefan Kolbl, and Martin M. Lauridsen. Security of AES with a Secret S-box . In the preproceedings of FSE 2015, 2015.

[41] Emmanuel Volte, Valérie Nachef, and Jacques Patarin. Improved Generic Attacks on Unbalanced Feistel Schemes with Expanding Functions. In *Advances in Cryptology - ASIACRYPT 2010*, LNCS, Vol. 6477, pages 94–111, 2010.

[42] Brecht Wyseur. White-Box Cryptography. PhD thesis, KU Leuven, 2009.

[43] Brecht Wyseur, Wil Michiels, Paul Gorissen, and Bart Preneel. Cryptanalysis of White-Box DES Implementations with Arbitrary External Encodings. In *Selected Areas in Cryptography - SAC 2007*, LNCS, Vol. 4876, pages 264–277, 2007.

[44] Yaying Xiao and Xuejia Lai. A Secure Implementation of White-box AES. In *2nd International Conference on Computer Science and its Applications - CSA 2009*, pages 1–6, 2009.