



**HAL**  
open science

# Streebog and Kuznyechik: Inconsistencies in the Claims of their Designers

Léo Perrin

► **To cite this version:**

Léo Perrin. Streebog and Kuznyechik: Inconsistencies in the Claims of their Designers. IETF 105, Jul 2019, Montreal, Canada. hal-02396671

**HAL Id: hal-02396671**

**<https://inria.hal.science/hal-02396671>**

Submitted on 6 Dec 2019

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Streebog and Kuznyechik

Inconsistencies in the Claims of their Designers

**Léo Perrin**

leo.perrin@inria.fr

 @lpp\_crypto

IETF Workshop, Montréal



## Partitions in the S-Box of Streebog and Kuznyechik

Léo Perrin

Iria, France

leo.perrin@ria.fr

**Abstract.** Streebog and Kuznyechik are the latest symmetric cryptographic primitives standardized by the Russian GOST. They share the same S-Box,  $\pi$ , whose design process was not described by its authors. In previous works, Biryukov, Perrin and Udovenko presented two completely different decompositions of this S-Box. We revisit their results and identify a third decomposition of  $\pi$ . It is an instance of a fairly small family of permutations operating on 256 bits which we call TKlog and which is closely related to finite field logarithms. Its simplicity and the small number of components it uses lead us to claim that it has to be the structure intentionally used by the designers of Streebog and Kuznyechik.

The 256-bit permutations of this type have a very strong algebraic structure: they map multiplicative cosets of the subfield  $\text{GF}(2^{16})^*$  to additive cosets of  $\text{GF}(2^{16})$ . Furthermore, the function relating such multiplicative coset to the corresponding additive coset is always essentially the same. To the best of our knowledge, we are the first to expose this very strong algebraic structure.

We also investigate other properties of the TKlog and show in particular that it can always be decomposed in a fashion similar to the first decomposition of Biryukov et al., thus explaining the relation between the two previous decompositions. It also means that it is always possible to implement a TKlog efficiently in hardware and that it always exhibits a visual pattern in its LUT similar to the one present in  $\pi$ . While we could not find attacks based on these new results, we discuss the impact of our work on the security of Streebog and Kuznyechik. To this end, we provide a new simpler representation of the linear layer of Streebog as a matrix multiplication in the exact same field as the one used to define  $\pi$ . We deduce that this matrix interacts in a non-trivial way with the partitions generated by  $\pi$ .

**Keywords:** Boolean function · Kuznyechik · Streebog · Reverse-Engineering · Partitions · Cosets · TKlog

### 1 Introduction

Many symmetric primitives rely on S-Boxes as their unique source of non-linearity, including the AES [AES01]. Such objects are small functions mapping  $\mathbb{F}_2^8$  to  $\mathbb{F}_2^8$  which are often specified via their look-up tables.

Their choice is crucial as both the security and the efficiency of the primitive depends heavily on their properties. For example, a low differential uniformity [N16a] implies a higher resilience against differential attacks [BS91a, BS91b]. On the other hand, the existence of a simple decomposition greatly helps with an efficient bit-level or hardware implementation [LV14, CKL16]. Thus, algorithm designers are expected to provide detailed explanation about their choice of S-Box. Each cipher that was published at a cryptography or security conference has provided such explanations.

There are two prominent S-Boxes for which this information has not been provided. The first is the so-called “F-table” of Skipjack [U596], a lightweight block cipher designed

Licensed under [Creative Commons License CC-BY 4.0](#).

IACR Transactions on Symmetric Cryptology ISSN 2519-171X, Vol. 2019, No. 1, pp. 302–329  
DOI:10.13154/tosc.v2019.1.302-329

## Transactions in Symmetric Cryptology, Volume 2019, No. 1, pp. 302–329. **Best paper award!**

- What is this result?
- Why is it **inconsistent** with the claims of the designers of these algorithms?

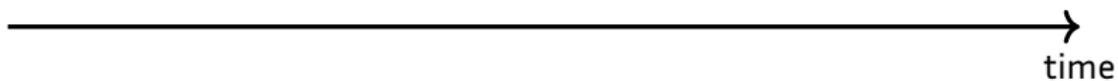
# Outline

- 1 Standards and S-boxes
- 2 On the S-box of RFC 6986 and 7801
- 3 The Core Issue: the S-Box Generation Process
- 4 Conclusion

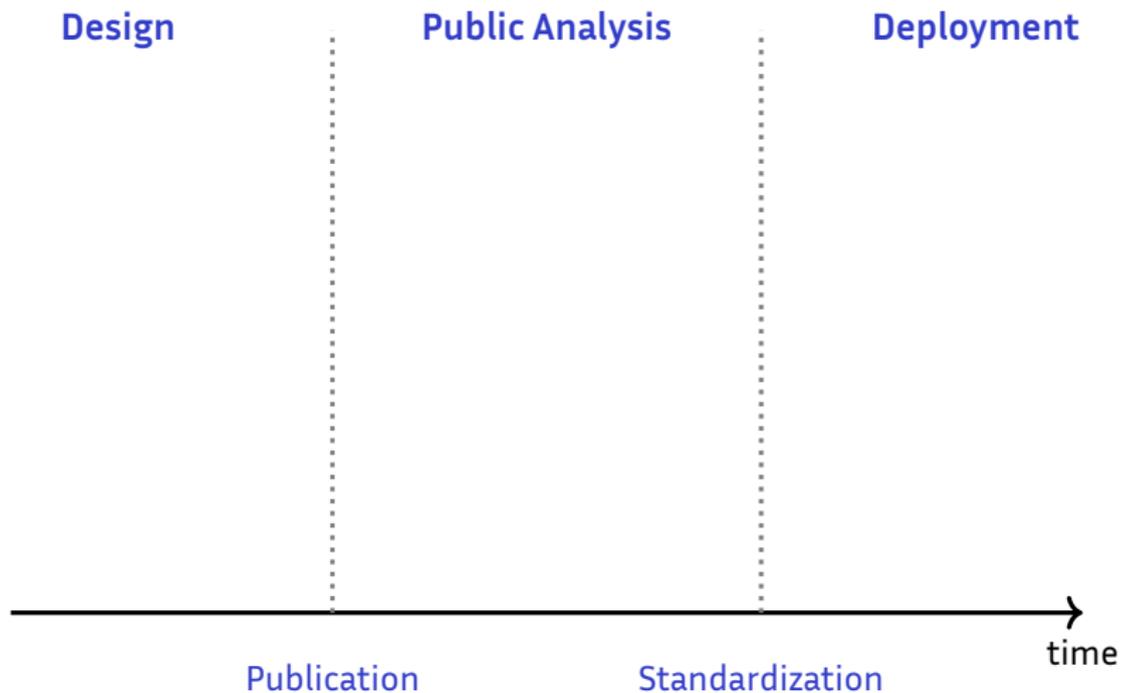
# Outline

- 1** Standards and S-boxes
- 2 On the S-box of RFC 6986 and 7801
- 3 The Core Issue: the S-Box Generation Process
- 4 Conclusion

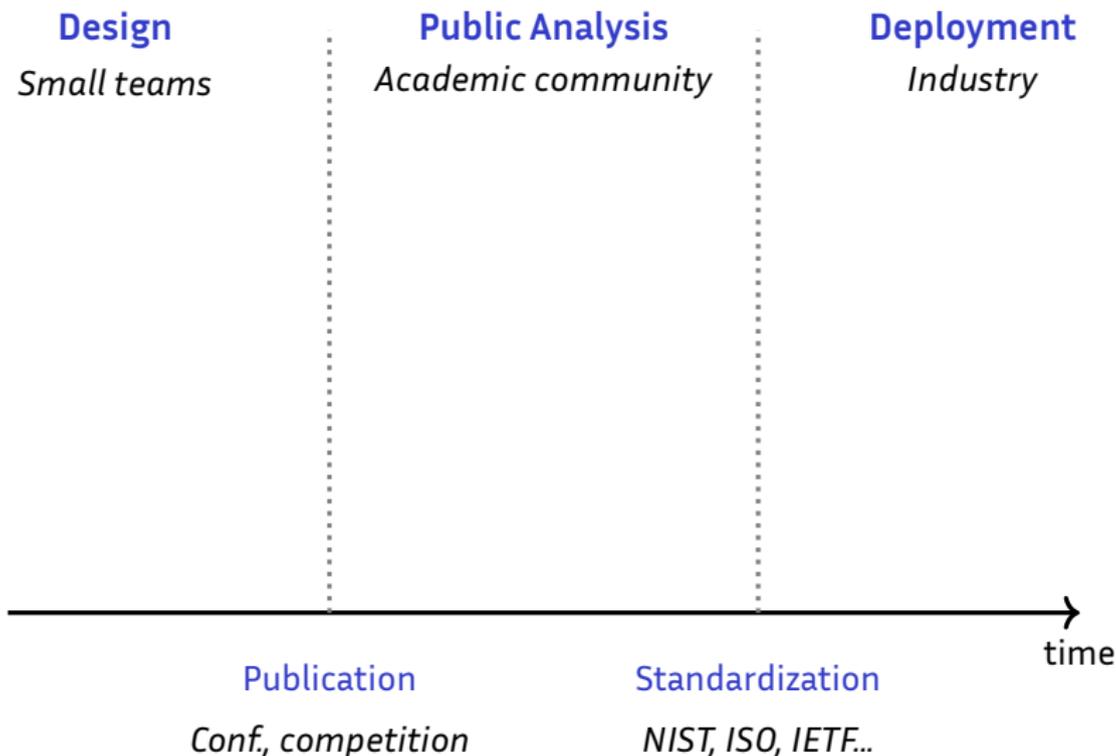
# Life Cycle of a Cryptographic Primitive



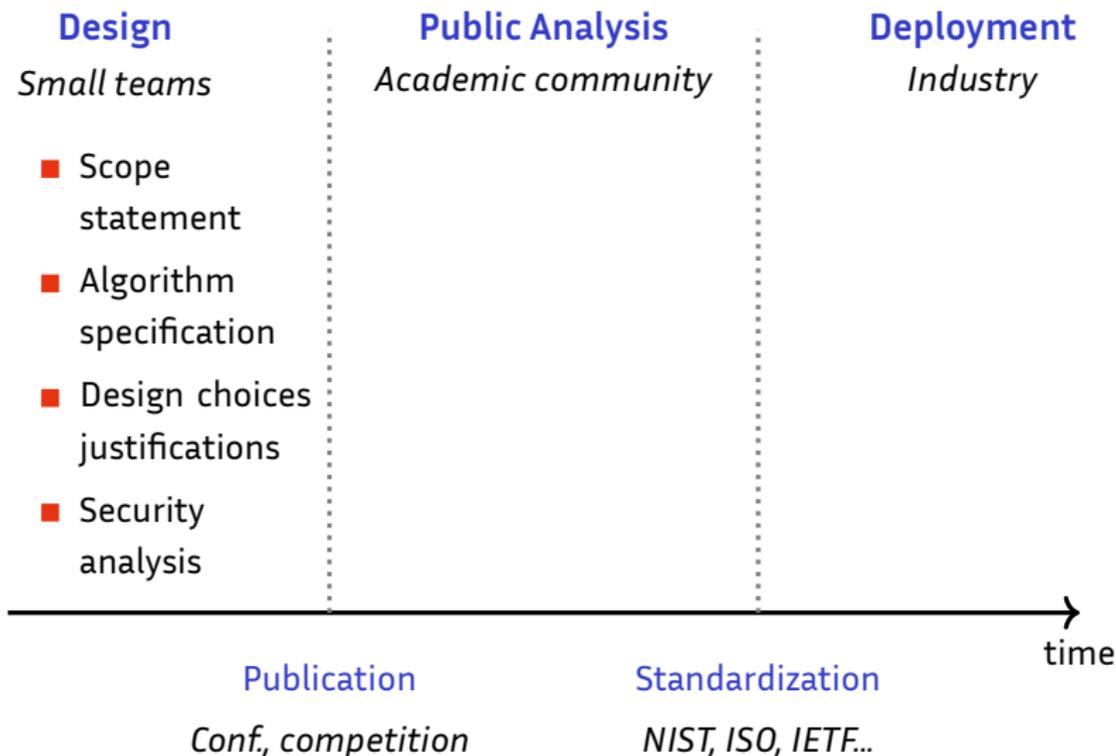
# Life Cycle of a Cryptographic Primitive



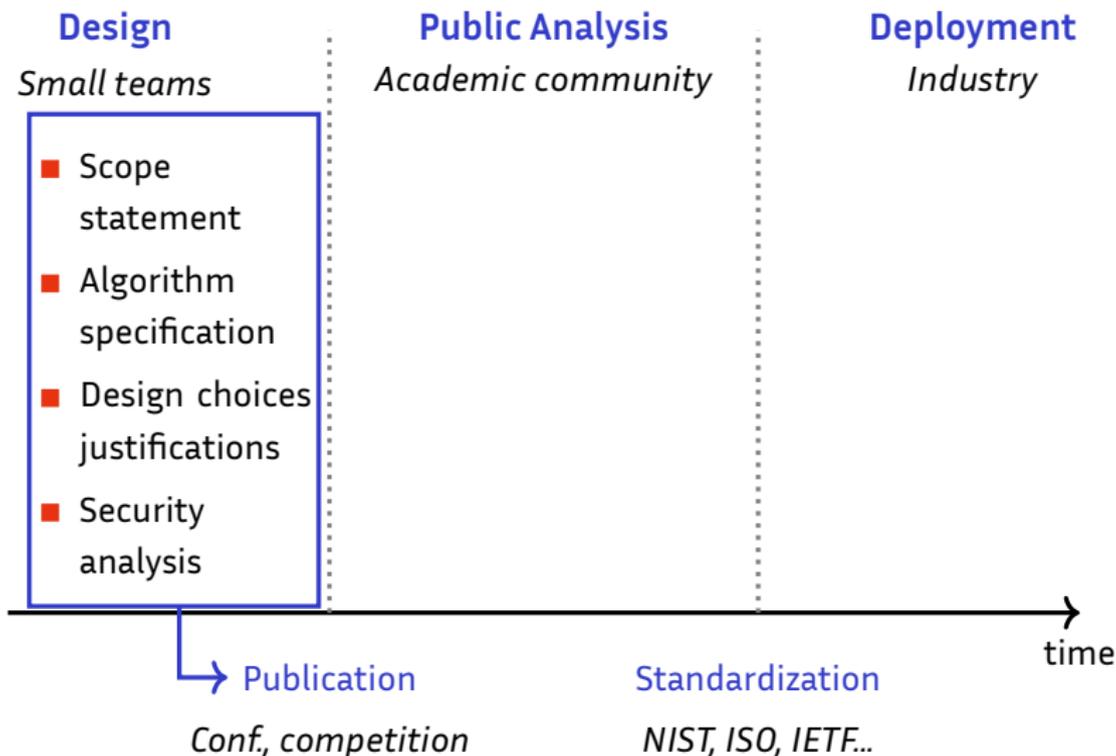
# Life Cycle of a Cryptographic Primitive



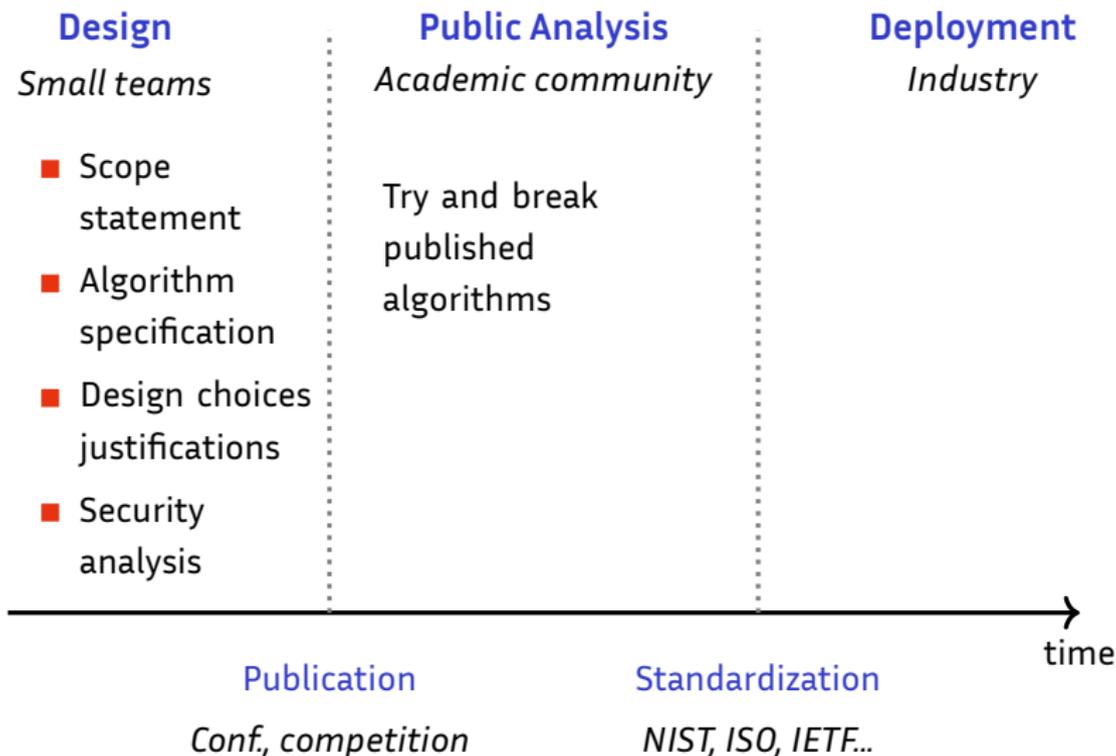
# Life Cycle of a Cryptographic Primitive



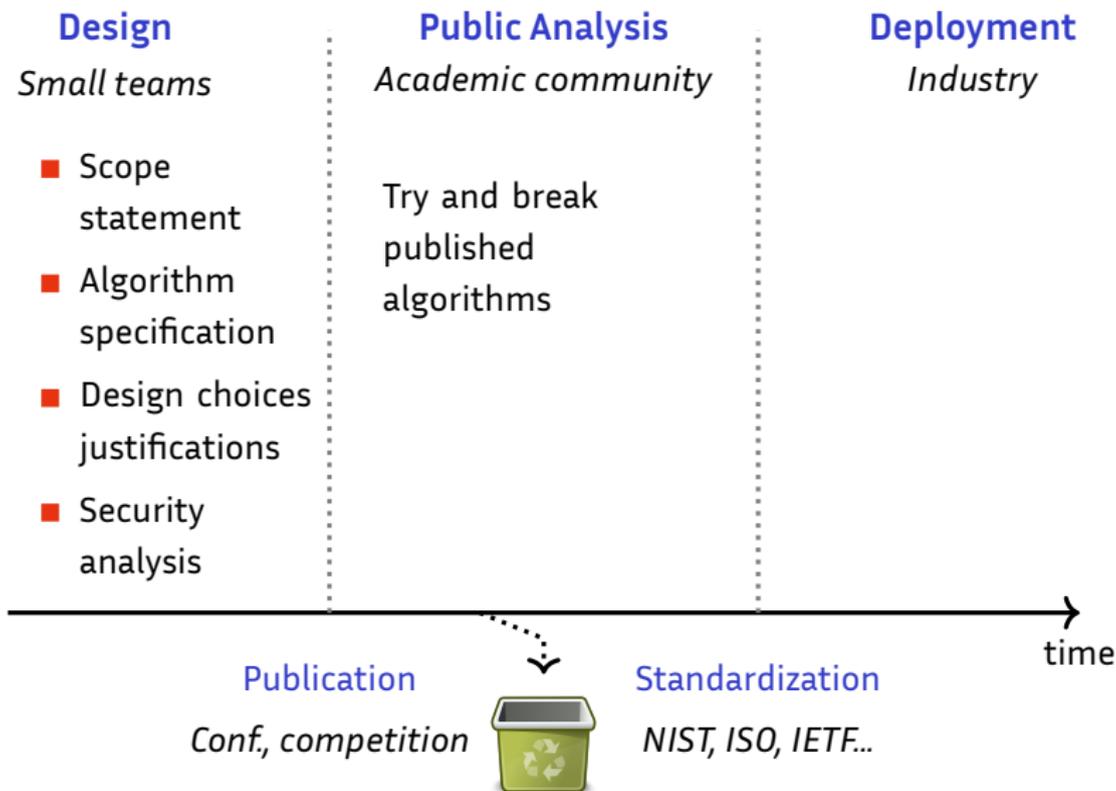
# Life Cycle of a Cryptographic Primitive



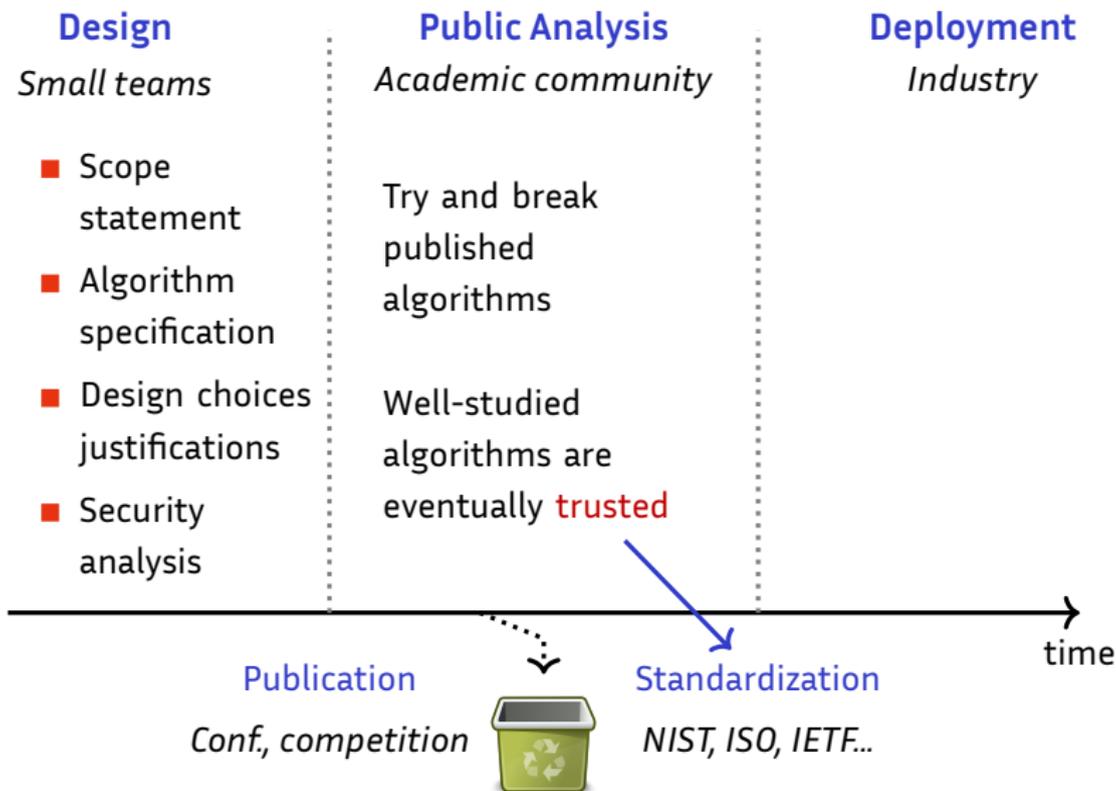
# Life Cycle of a Cryptographic Primitive



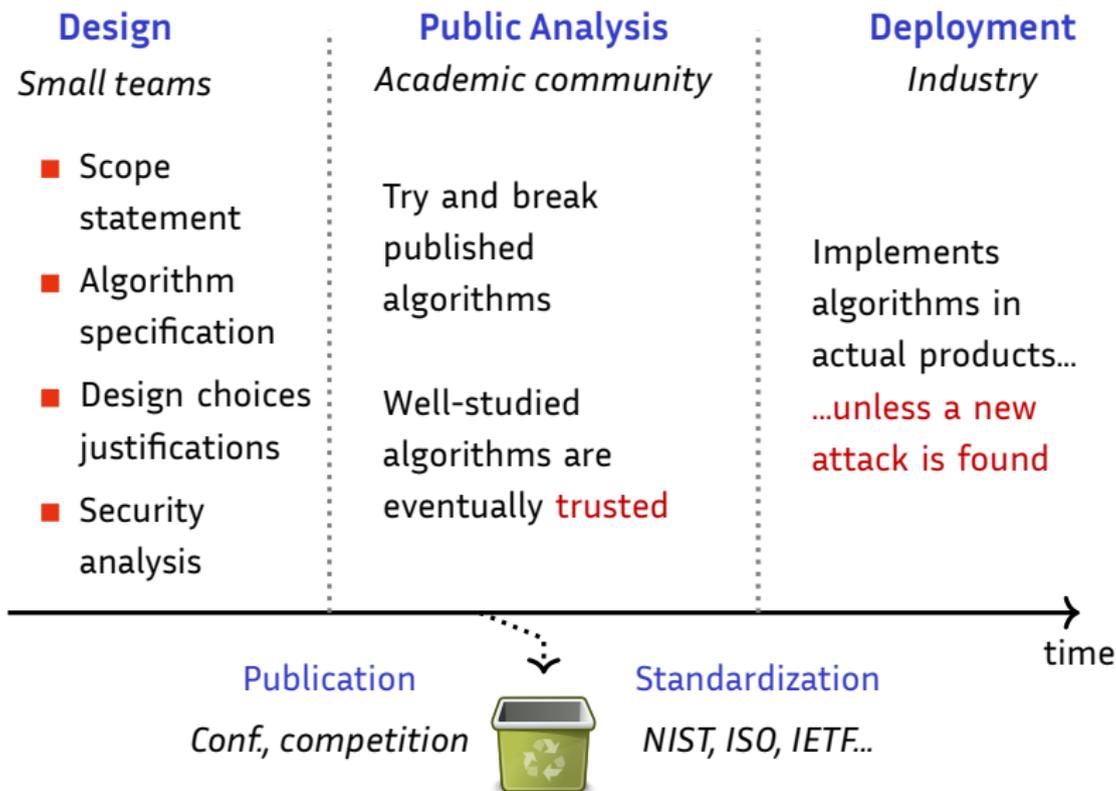
# Life Cycle of a Cryptographic Primitive



# Life Cycle of a Cryptographic Primitive



# Life Cycle of a Cryptographic Primitive



# Breaking the Pipeline

## Design

*Small teams*

- Scope statement
- Algorithm specification
- Design choices justifications
- Security analysis

## Public Analysis

*Academic community*

Try and break published algorithms

Well-studied algorithms are eventually **trusted**

## Deployment

*Industry*

Implements algorithms in actual products...  
...unless a new attack is found

Publication

*Conf., competition*

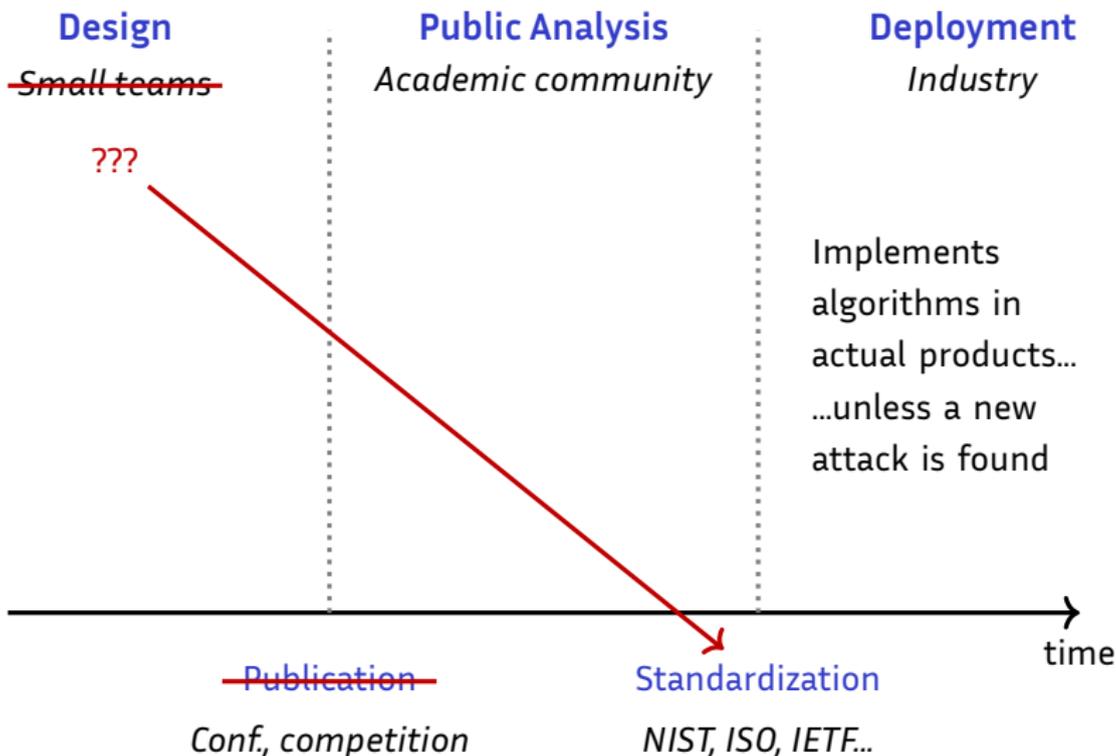


Standardization

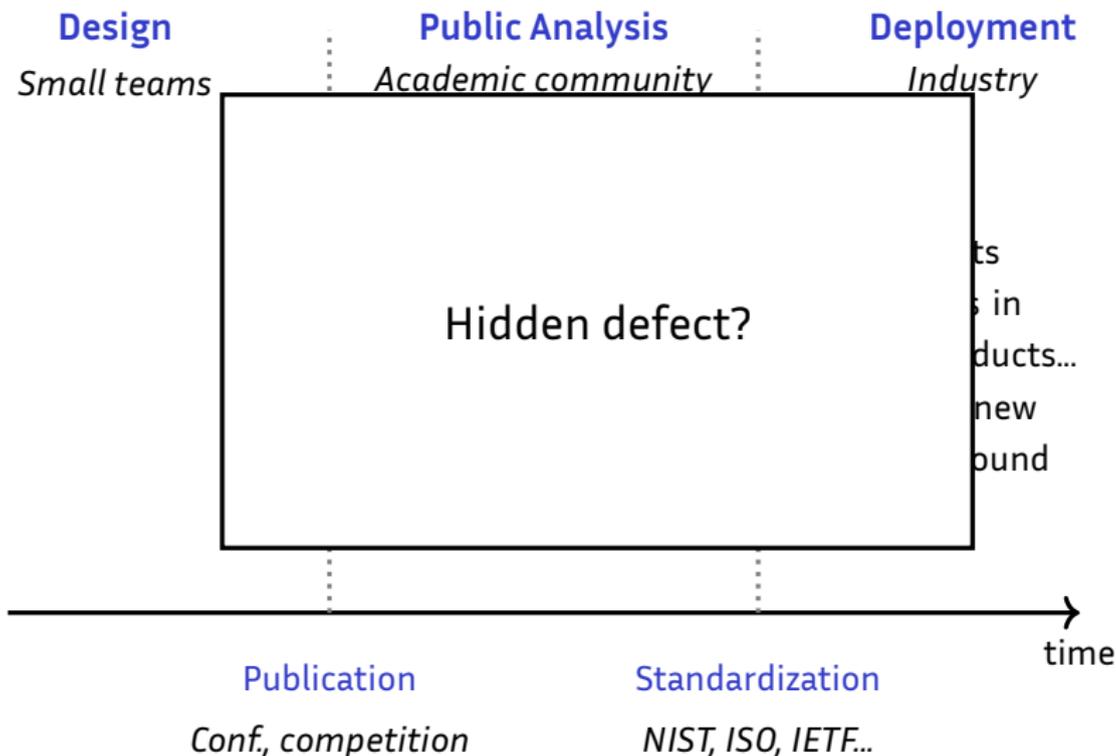
*NIST, ISO, IETF...*

time →

# Breaking the Pipeline



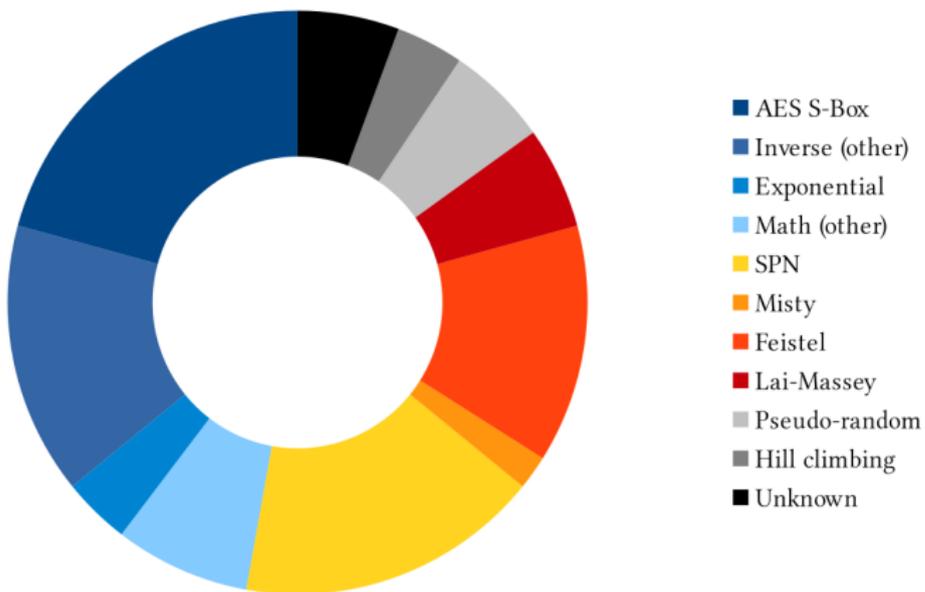
# Breaking the Pipeline



# S-Boxes

## Definition (S(ubstitution)-box)

An S-box  $S : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$  is a small **non-linear** function operating on a small block size (typically  $n \in \{4, 8\}$ ) which **can** be specified via its lookup table.



# Specifying the AES S-box

Authors:

Joan Daemen  
Vincent Rijmen

## The Rijndael Block Cipher AES Proposal

### 7.2 The ByteSub S-box

The design criteria for the S-box are inspired by differential and linear cryptanalysis on the one hand and attacks using algebraic manipulations, such as interpolation attacks, on the other:

1. Invertibility;
2. Minimisation of the largest non-trivial correlation between linear combinations of input bits and linear combination of output bits;
3. Minimisation of the largest non-trivial value in the EXOR table;
4. Complexity of its algebraic expression in  $\text{GF}(2^8)$ ;
5. Simplicity of description.

In [Ny94] several methods are given to construct S-boxes that satisfy the first three criteria. For invertible S-boxes operating on bytes, the maximum input/output correlation can be made as low as  $2^{-3}$  and the maximum value in the EXOR table can be as low as 4 (corresponding to a difference propagation probability of  $2^{-3}$ ).

We have decided to take from the candidate constructions in [Ny94] the S-box defined by the mapping  $x \Rightarrow x^3$  in  $\text{GF}(2^8)$ .

By definition, the selected mapping has a very simple algebraic expression. This enables algebraic manipulations that can be used to mount attacks such as interpolation attacks [Jaf97]. Therefore, the mapping is modified by composing it with an additional invertible affine transformation. This affine transformation does not affect the properties with respect to the first three criteria, but if properly chosen, allows the S-box to satisfy the fourth criterion.

We have chosen an affine mapping that has a very simple description per se, but a complicated algebraic expression if combined with the 'inverse' mapping. It can be seen as modular polynomial multiplication followed by an addition:

$$b(x) = (x^7 + x^6 + x^5 + x) + a(x)(x^7 + x^6 + x^5 + x + 1) \bmod x^8 + 1$$

The modulus has been chosen as the simplest modulus possible. The multiplication polynomial has been chosen from the set of polynomials coprime to the modulus as the one with the simplest description. The constant has been chosen in such a way that the S-box has no fixed points (S-box(a) = a) and no 'opposite fixed points' (S-box(a) =  $\bar{a}$ ).

**Note:** other S-boxes can be found that satisfy the criteria above. In the case of suspicion of a trapdoor being built into the cipher, the current S-box might be replaced by another one. The cipher structure and number of rounds as defined even allow the use of an S-box that does not optimise the differential and linear cryptanalysis properties (criteria 2 and 3). Even an S-box that is "average" in this respect is likely to provide enough resistance against differential and linear cryptanalysis.

<https://csrc.nist.gov/csrc/media/projects/cryptographic-standards-and-guidelines/documents/aes-development/rijndael-amended.pdf>

# Specifying the AES S-box

Authors: **The Rijndael Block Cipher** AES Proposal  
 Joan Daemen  
 Vincent Rijmen

## 7.2 The ByteSub S-box

The design criteria for the S-box are inspired by differential and linear cryptanalysis on the one hand,

1. Invertibility;
2. Minimisation of the largest non-trivial correlation between linear combinations of input bits and linear combination of output bits;
3. Minimisation of the largest non-trivial value in the EXOR table;
4. Complexity of its algebraic expression in  $\text{GF}(2^8)$ ;
5. Simplicity of description.

In [Rij94] we proposed a set of invertible S-boxes operating on bytes, the maximum input/output correlation can be made as low as  $2^{-3}$  and the maximum value in the EXOR table can be as low as 4 (corresponding to a difference propagation probability of  $2^{-3}$ ).

We have decided to take from the candidate constructions in [Ny94] the S-box defined by the mapping  $x \Rightarrow x^3$  in  $\text{GF}(2^8)$ .

By definition, the selected mapping has a very simple algebraic expression. This enables algebraic manipulations that can be used to mount attacks such as interpolation attacks [Jaf97]. Therefore, the mapping is modified by composing it with an additional invertible affine transformation. This affine transformation does not affect the properties with respect to the first three criteria, but if properly chosen, allows the S-box to satisfy the fourth criterion.

We have chosen an affine mapping that has a very simple description per se, but a complicated algebraic expression if combined with the 'inverse' mapping. It can be seen as modular polynomial multiplication followed by an addition:

$$b(x) = (x^7 + x^6 + x^5 + x) + a(x)(x^7 + x^6 + x^5 + 1) \bmod x^8 + 1$$

The modulus has been chosen as the simplest modulus possible. The multiplication polynomial has been chosen from the set of polynomials coprime to the modulus as the one with the simplest description. The constant has been chosen in such a way that the S-box has no fixed points (S-box(a) = a) and no 'opposite fixed points' (S-box(a) =  $\bar{a}$ ).

**Note:** other S-boxes can be found that satisfy the criteria above. In the case of suspicion of a trapdoor being built into the cipher, the current S-box might be replaced by another one. The cipher structure and number of rounds as defined even allow the use of an S-box that does not optimise the differential and linear cryptanalysis properties (criteria 2 and 3). Even an S-box that is "average" in this respect is likely to provide enough resistance against differential and linear cryptanalysis.

<https://csrc.nist.gov/csrc/media/projects/cryptographic-standards-and-guidelines/documents/aes-development/rijndael-ammended.pdf>

## 1 Clear design goals

# Specifying the AES S-box

Authors:

Jean-Philippe Aumasson  
Vincent Rijmen

## The Rijndael Block Cipher AES Proposal

### 7.2 The ByteSub S-box

The design criteria for the S-box are inspired by differential and linear cryptanalysis on the one hand and attacks using algebraic manipulations, such as interpolation attacks, on the other:

1. Invertibility;
2. Minimisation of the largest non-trivial correlation between linear combinations of input bits and linear combination of output bits;
3. Minimisation of the largest non-trivial value in the EXOR table;
4. Complexity of its algebraic expression in  $\text{GF}(2^8)$ ;
5. Simplicity of translation.

In [Ny94] several methods are given to construct S-boxes that satisfy the first three criteria. For invertible S-boxes operating on bytes, the maximum input/output correlation can be made as low as  $2^{-3}$  and the maximum value in the EXOR table can be as low as 4 (corresponding to a difference propagation probability of  $2^{-3}$ ).

We have decided to take from the candidate constructions in [Ny94] the S-box defined by the mapping  $x \mapsto x^3$  in  $\text{GF}(2^8)$ .

By definition, the selected mapping has a very simple algebraic expression. This enables algebraic manipulations that can be used to mount attacks such as interpolation attacks [JafK97]. Therefore, the mapping is modified by composing it with an additional invertible affine transformation. This affine transformation does not affect the properties with respect to the first three criteria, but if properly chosen, allows the S-box to satisfy the fourth criterion.

We have chosen an affine mapping that has a very simple description per se, but a complicated algebraic expression if combined with the 'inverse' mapping. It can be seen as modular polynomial multiplication followed by an addition:

$$b(x) = (x^7 + x^6 + x^5 + x^4 + a)(x^7 + x^6 + x^5 + x^4 + 1) \bmod x^8 + 1$$

The modulus has been chosen as the simplest modulus possible. The multiplication polynomial has been chosen from the set of polynomials coprime to the modulus as the one with the simplest description. The constant has been chosen in such a way that the S-box has no fixed points (S-box(a) = a) and no 'opposite fixed points' (S-box(a) =  $\bar{a}$ ).

**Note:** other S-boxes can be found that satisfy the criteria above. In the case of suspicion of a trapdoor being built into the cipher, the current S-box might be replaced by another one. The cipher structure and number of rounds as defined even allow the use of an S-box that does not optimise the differential and linear cryptanalysis properties (criteria 2 and 3). Even an S-box that is "average" in this respect is likely to provide enough resistance against differential and linear cryptanalysis.

<https://csrc.nist.gov/csrc/media/projects/cryptographic-standards-and-guidelines/documents/aes-development/rijndael-ammended.pdf>

- 1 Clear design goals
- 2 Motivation for the specific solution chosen

# Specifying the AES S-box

Authors:

Joan Daemen  
Vincent Rijmen

## The Rijndael Block Cipher AES Proposal

### 7.2 The ByteSub S-box

The design criteria for the S-box are inspired by differential and linear cryptanalysis on the one hand and attacks using algebraic manipulations, such as interpolation attacks, on the other:

1. Invertibility;
2. Minimisation of the largest non-trivial correlation between linear combinations of input bits and linear combination of output bits;
3. Minimisation of the largest non-trivial value in the EXOR table;
4. Complexity of its algebraic expression in  $\text{GF}(2^8)$ ;
5. Simplicity of description.

In [Ny94] several methods are given to construct S-boxes that satisfy the first three criteria. For invertible S-boxes operating on bytes, the maximum input/output correlation can be made as low as  $2^{-3}$  and the maximum value in the EXOR table can be as low as 4 (corresponding to a difference propagation probability of  $2^{-3}$ ).

We have decided to take from the candidate constructions in [Ny94] the S-box defined by the following expression:

By definition, the selected mapping has a very simple algebraic expression. This enables algebraic manipulations that can be used to mount attacks such as interpolation attacks [Jaf97]. Therefore, the mapping is modified by composing it with an additional invertible affine transformation. This affine transformation does not affect the properties with respect to the first three criteria, but if properly chosen, allows the S-box to satisfy the fourth criterion.

The inverse mapping of the above mapping is a very simple, but somewhat complicated algebraic expression if combined with the 'inverse' mapping. It can be seen as modular polynomial multiplication followed by an addition:

$$h(x) = (x^7 + x^6 + x^5 + x) + a(x)(x^7 + x^6 + x^5 + 1) \bmod x^8 + 1$$

The modulus has been chosen as the simplest modulus possible. The multiplication polynomial has been chosen from the set of polynomials coprime to the modulus as the one with the simplest description. The constant has been chosen in such a way that the S-box has no fixed points (S-box( $a$ ) =  $a$ ) and no 'opposite fixed points' (S-box( $a$ ) =  $\bar{a}$ ).

**Note:** other S-boxes can be found that satisfy the criteria above. In the case of suspicion of a trapdoor being built into the cipher, the current S-box might be replaced by another one. The cipher structure and number of rounds as defined even allow the use of an S-box that does not optimise the differential and linear cryptanalysis properties (criteria 2 and 3). Even an S-box that is "average" in this respect is likely to provide enough resistance against differential and linear cryptanalysis.

<https://csrc.nist.gov/csrc/media/projects/cryptographic-standards-and-guidelines/documents/aes-development/rijndael-ammended.pdf>

- 1 Clear design goals
- 2 Motivation for the specific solution chosen
- 3 A possible pitfall and how it is avoided

# Specifying the AES S-box

Authors:  
Joan Daemen  
Vincent Rijmen

## The Rijndael Block Cipher AES Proposal

### 7.2 The ByteSub S-box

The design criteria for the S-box are inspired by differential and linear cryptanalysis on the one hand and attacks using algebraic manipulations, such as interpolation attacks, on the other:

1. Invertibility;
2. Minimisation of the largest non-trivial correlation between linear combinations of input bits and linear combination of output bits;
3. Minimisation of the largest non-trivial value in the EXOR table;
4. Complexity of its algebraic expression in  $\text{GF}(2^8)$ ;
5. Simplicity of description.

In [Ny94] several methods are given to construct S-boxes that satisfy the first three criteria. For invertible S-boxes operating on bytes, the maximum input/output correlation can be made as low as  $2^{-3}$  and the maximum value in the EXOR table can be as low as 4 (corresponding to a difference propagation probability of  $2^{-3}$ ).

We have decided to take from the candidate constructions in [Ny94] the S-box defined by the mapping  $x \Rightarrow x^3$  in  $\text{GF}(2^8)$ .

By definition, the selected mapping has a very simple algebraic expression. This enables algebraic manipulations that can be used to mount attacks such as interpolation attacks [Juk97]. Therefore, the mapping is modified by composing it with an additional invertible affine transformation. This affine transformation does not affect the properties with respect to the first three criteria, but if properly chosen, allows the S-box to satisfy the fourth criterion.

We have chosen an affine mapping that has a very simple description per se, but a complicated algebraic expression if combined with the 'inverse' mapping. It can be seen as modular polynomial multiplication followed by an addition:

$$b(x) = (x^7 + x^5 + x^2 + x) + a(x)(x^7 + x^6 + x^5 + x^4 + 1) \pmod{x^8 + 1}$$

The modulus has been chosen as the simplest modulus possible. The multiplication polynomial has been chosen from the set of polynomials coprime to the modulus as the one with the simplest description. The constant has been chosen in such a way that the S-box has no fixed points (S-box(a) = a) and no 'opposite fixed points' (S-box(a) =  $\bar{a}$ ).

**Note:** Other S-boxes can be found that satisfy the criteria above. In the case of substitution or a trapdoor being built into the cipher, the current S-box might be replaced by another one. The cipher structure and number of rounds as defined even allow the use of an S-box that does not optimise the differential and linear cryptanalysis properties (criteria 2 and 3). Even an S-box that is "average" in this respect is likely to provide enough resistance against differential and linear cryptanalysis.

<https://csrc.nist.gov/csrc/media/projects/cryptographic-standards-and-guidelines/documents/aes-development/rijndael-ammended.pdf>

- 1 Clear design goals
- 2 Motivation for the specific solution chosen
- 3 A possible pitfall and how it is avoided
- 4 Description of the process for choosing the actual instance

# Outline

- 1 Standards and S-boxes
- 2 On the S-box of RFC 6986 and 7801**
- 3 The Core Issue: the S-Box Generation Process
- 4 Conclusion

# Kuznyechik/Streebog

## Streebog (RFC 6986)

**Type** Hash function

**Publication** 2012 (RFC in Aug. 2013)

## Kuznyechik (RFC 7801)

**Type** Block cipher

**Publication** 2015 (RFC in Mar. 2016)

## Common ground

- Both are standards in Russia.
- They were designed by the TC26 (supervised by the FSB).
- Their RFCs come from the independent stream ( $\neq$  CFRG)
- Both use the same 8-bit S-Box,  $\pi$ .

## Timeline

July 2012	GOST standardization of Streebog	GOST
Aug. 2013	RFC for Streebog (RFC 6986)	IETF
June 2015	GOST standardization of Kuznyechik	GOST
Mar. 2016	RFC for Kuznyechik (RFC 7801)	IETF

# Timeline

- July 2012** GOST standardization of Streebog GOST
- Aug. 2013** RFC for Streebog (RFC 6986) IETF
- June 2015** GOST standardization of Kuznyechik GOST
- Mar. 2016** RFC for Kuznyechik (RFC 7801) IETF
- May 2016** Publication of the first decomposition IACR  
*Biryukov, Perrin, Udovenko. Reverse-engineering the S-box of Streebog, Kuznyechik and STRIBOBr1. EUROCRYPT'16*
- Mar. 2017** Publication of the second decomposition IACR  
*Perrin, Udovenko. Exponential S-Boxes: a Link Between the S-Boxes of BelT and Kuznyechik/Streebog. IACR ToSC 2016*

# Timeline

**July 2012** GOST standardization of Streebog GOST

**Aug. 2013** RFC for Streebog (RFC 6986) IETF

**June 2015** GOST standardization of Kuznyechik GOST

**Mar. 2016** RFC for Kuznyechik (RFC 7801) IETF

**May 2016** Publication of the first decomposition IACR

Biryukov, Perrin, Udovenko. *Reverse-engineering the S-box of Streebog, Kuznyechik and STRIBOBr1*. EUROCRYPT'16

**Mar. 2017** Publication of the second decomposition IACR

Perrin, Udovenko. *Exponential S-Boxes: a Link Between the S-Boxes of BelT and Kuznyechik/Streebog*. IACR ToSC 2016

**Oct. 2018** ISO standardization of Streebog (ISO 10118-3) ISO

# Timeline

**July 2012** GOST standardization of Streebog GOST

**Aug. 2013** RFC for Streebog (RFC 6986) IETF

**June 2015** GOST standardization of Kuznyechik GOST

**Mar. 2016** RFC for Kuznyechik (RFC 7801) IETF

**May 2016** Publication of the first decomposition IACR

*Biryukov, Perrin, Udovenko. Reverse-engineering the S-box of Streebog, Kuznyechik and STRIBOBr1. EUROCRYPT'16*

**Mar. 2017** Publication of the second decomposition IACR

*Perrin, Udovenko. Exponential S-Boxes: a Link Between the S-Boxes of BelT and Kuznyechik/Streebog. IACR ToSC 2016*

**Oct. 2018** ISO standardization of Streebog (ISO 10118-3) ISO

**Jan. 2019** Publication of the final decomposition IACR

*Perrin. Partitions in the S-box of Streebog and Kuznyechik. IACR ToSC 2019.*

# Timeline

- |           |  |      |
|-----------|--|------|
| July 2012 | GOST standardization of Streebog   | GOST |
| Aug. 2013 | RFC for Streebog (RFC 6986)  | IETF |
| June 2015 | GOST standardization of Kuznyechik   | GOST |
| Mar. 2016 | RFC for Kuznyechik (RFC 7801)  | IETF |
| May 2016  | Publication of the first decomposition   | IACR |
|           | <i>Biryukov, Perrin, Udovenko. Reverse-engineering the S-box of Streebog, Kuznyechik and STRIBOBr1. EUROCRYPT'16</i>     |      |
| Mar. 2017 | Publication of the second decomposition  | IACR |
|           | <i>Perrin, Udovenko. Exponential S-Boxes: a Link Between the S-Boxes of BelT and Kuznyechik/Streebog. IACR ToSC 2016</i> |      |
| Oct. 2018 | ISO standardization of Streebog (ISO 10118-3)  | ISO  |
| Jan. 2019 | Publication of the final decomposition   | IACR |
|           | <i>Perrin. Partitions in the S-box of Streebog and Kuznyechik. IACR ToSC 2019.</i>                                       |      |
| Feb. 2019 | Kuznyechik at ISO: <b>decision post-poned</b>  | ISO  |

# Timeline

- |           |  |      |
|-----------|--|------|
| July 2012 | GOST standardization of Streebog   | GOST |
| Aug. 2013 | RFC for Streebog (RFC 6986)  | IETF |
| June 2015 | GOST standardization of Kuznyechik   | GOST |
| Mar. 2016 | RFC for Kuznyechik (RFC 7801)  | IETF |
| May 2016  | Publication of the first decomposition   | IACR |
|           | <i>Biryukov, Perrin, Udovenko. Reverse-engineering the S-box of Streebog, Kuznyechik and STRIBOBr1. EUROCRYPT'16</i>     |      |
| Mar. 2017 | Publication of the second decomposition  | IACR |
|           | <i>Perrin, Udovenko. Exponential S-Boxes: a Link Between the S-Boxes of BelT and Kuznyechik/Streebog. IACR ToSC 2016</i> |      |
| Oct. 2018 | ISO standardization of Streebog (ISO 10118-3)  | ISO  |
| Jan. 2019 | Publication of the final decomposition   | IACR |
|           | <i>Perrin. Partitions in the S-box of Streebog and Kuznyechik. IACR ToSC 2019.</i>                                       |      |
| Feb. 2019 | Kuznyechik at ISO: <b>decision post-poned</b>  | ISO  |
| Sep. 2019 | Kuznyechik at ISO: <b>decision must be taken!</b>  | ISO  |

# Outline

- 1 Standards and S-boxes
- 2 On the S-box of RFC 6986 and 7801
- 3 The Core Issue: the S-Box Generation Process**
- 4 Conclusion

## The Russian S-box

$\pi' = (252, 238, 221, 17, 207, 110, 49, 22, 251, 196, 250, 218, 35, 197, 4, 77, 233, 119, 240, 219, 147, 46, 153, 186, 23, 54, 241, 187, 20, 205, 95, 193, 249, 24, 101, 90, 226, 92, 239, 33, 129, 28, 60, 66, 139, 1, 142, 79, 5, 132, 2, 174, 227, 106, 143, 160, 6, 11, 237, 152, 127, 212, 211, 31, 235, 52, 44, 81, 234, 200, 72, 171, 242, 42, 104, 162, 253, 58, 206, 204, 181, 112, 14, 86, 8, 12, 118, 18, 191, 114, 19, 71, 156, 183, 93, 135, 21, 161, 150, 41, 16, 123, 154, 199, 243, 145, 120, 111, 157, 158, 178, 177, 50, 117, 25, 61, 255, 53, 138, 126, 109, 84, 198, 128, 195, 189, 13, 87, 223, 245, 36, 169, 62, 168, 67, 201, 215, 121, 214, 246, 124, 34, 185, 3, 224, 15, 236, 222, 122, 148, 176, 188, 220, 232, 40, 80, 78, 51, 10, 74, 167, 151, 96, 115, 30, 0, 98, 68, 26, 184, 56, 130, 100, 159, 38, 65, 173, 69, 70, 146, 39, 94, 85, 47, 140, 163, 165, 125, 105, 213, 149, 59, 7, 88, 179, 64, 134, 172, 29, 247, 48, 55, 107, 228, 136, 217, 231, 137, 225, 27, 131, 73, 76, 63, 248, 254, 141, 83, 170, 144, 202, 216, 133, 97, 32, 113, 103, 164, 45, 43, 9, 91, 203, 155, 37, 208, 190, 229, 108, 82, 89, 166, 116, 210, 230, 244, 180, 192, 209, 102, 175, 194, 57, 75, 99, 182).$

*Screen capture of the specification of **Kuznyechik** (2015).*

# How Was it Generated?

## According to the designers (April 2018)

questioned is the S-box  $\pi$ . This S-box was chosen from Streebog hash-function and it was synthesized in 2007. Note that through many years of cryptanalysis no weakness of this S-box was found. The S-box  $\pi$  was obtained by pseudo-random search and the following properties were taken into account.

[..]

No secret structure was enforced during construction of the S-box. At the same time, it is obvious that for any transformation a lot of representations are possible (see, for example, a lot of AES S-box representations).

- Source: <https://cdn.virgilsecurity.com/assets/docs/memo-on-kuznyechik-s-box.pdf>
- See also the discussion summary: <https://cdn.virgilsecurity.com/assets/docs/meeting-report-for-the-discussion-on-kuznyechik-and-streebog.pdf>

# How Was it Generated?

## According to the designers (April 2018)

questioned is the S-box  $\pi$ . This S-box was chosen from Streebog hash-function and it was synthesized in 2007. Note that through many years of cryptanalysis no weakness of this S-box was found. The S-box  $\pi$  was obtained by pseudo-random search and the following properties were taken into account.

[..]

No secret structure was enforced during construction of the S-box. At the same time, it is obvious that for any transformation a lot of representations are possible (see, for example, a lot of AES S-box representations).

- Source: <https://cdn.virgilsecurity.com/assets/docs/memo-on-kuznyechik-s-box.pdf>
- See also the discussion summary: <https://cdn.virgilsecurity.com/assets/docs/meeting-report-for-the-discussion-on-kuznyechik-and-streebog.pdf>

## What I proved (IACR ToSC 2019)

$$\pi \begin{cases} \mathbb{F}_{2^8} & \rightarrow \mathbb{F}_{2^8} \\ 0 & \mapsto \kappa(0), \\ (\alpha^{2^m+1})^j & \mapsto \kappa(2^m - j), \text{ for } 1 \leq j \leq 2^m - 1, \\ \alpha^{i+(2^m+1)j} & \mapsto \kappa(2^m - i) \oplus (\alpha^{2^m+1})^{s(j)}, \text{ for } 0 < i, 0 \leq j < 2^m - 1. \end{cases}$$

## Such a Structure is Beyond Unlikely

### Lemma (more details available online<sup>1</sup>)

*There are  $256! \approx 2^{1684}$  different 8-bit permutations, meaning you need at least **1684 bits** to represent all of them in **any** language.*

---

<sup>1</sup> Bonnetain, Perrin, Tian. **Anomalies and Vector Space Search: Tools for S-Box Reverse-Engineering**. <https://ia.cr/2019/528>

<sup>2</sup> Credit to @odzhan on stackexchange.

<sup>3</sup> <https://codegolf.stackexchange.com/questions/186498/proving-that-a-russian-cryptographic-standard-is-too-structured>

## Such a Structure is Beyond Unlikely

### Lemma (more details available online<sup>1</sup>)

*There are  $256! \approx 2^{1684}$  different 8-bit permutations, meaning you need at least **1684** bits to represent all of them in **any** language.*

```
p(x){unsigned char*k="Q`rFTDVbpPB  
vdtfRQ\xacp?\xe2>4\xa6\xe9{z\xe3q  
5\xa7\xe8",a=2,l=0,b=17;while(x&&  
(l++,a^x))a=2*a^a/128*29;return l  
%b?k[l%b]^k[b+l/b]^b:k[l/b]^188;}
```

- 165 ASCII characters that fit on 7 bits: this program is **1155**-bit long

---

<sup>1</sup> Bonnetain, Perrin, Tian. *Anomalies and Vector Space Search: Tools for S-Box Reverse-Engineering*. <https://ia.cr/2019/528>

<sup>2</sup> Credit to @odzhan on stackexchange.

<sup>3</sup> <https://codegolf.stackexchange.com/questions/186498/>

## Such a Structure is Beyond Unlikely

### Lemma (more details available online<sup>1</sup>)

*There are  $256! \approx 2^{1684}$  different 8-bit permutations, meaning you need at least **1684** bits to represent all of them in **any** language.*

```
p(x){unsigned char*k="Q`rFTDVbpPB  
vdtfRQ\xacp?\xe2>4\xa6\xe9{z\xe3q  
5\xa7\xe8",a=2,l=0,b=17;while(x&&  
(l++,a^x))a=2*a^a/128*29;return l  
%b?k[l%b]^k[b+l/b]^b:k[l/b]^188;}
```

- 165 ASCII characters that fit on 7 bits: this program is **1155**-bit long
- An AMD64 binary implementation fits<sup>2</sup> on 78 bytes, i.e. **624** bits.
- Many more short implementations have been found by code golfers!<sup>3</sup>

<sup>1</sup> Bonnetain, Perrin, Tian. *Anomalies and Vector Space Search: Tools for S-Box Reverse-Engineering*. <https://ia.cr/2019/528>

<sup>2</sup> Credit to @odzhan on stackexchange.

<sup>3</sup> <https://codegolf.stackexchange.com/questions/186498/>

## Such a Structure is Beyond Unlikely

### Lemma (more details available online<sup>1</sup>)

There are  $256! \approx 2^{1684}$  different 8-bit permutations, meaning you need at least **1684 bits** to represent all of them in **any** language.

```
p(x){unsigned char*k="Q`rFTDVbpPB  
vdtfRQ\xacp?\xe2>4\xa6\xe9{z\xe3q  
5\xa7\xe8",a=2,l=0,b=17;while(x&&  
(l++,a^x))a=2*a^a/128*29;return l  
%b?k[l%b]^k[b+l/b]^b:k[l/b]^188;}
```

- 165 ASCII characters that fit on 7 bits: this program is **1155-bit** long
- An AMD64 binary implementation fits<sup>2</sup> on 78 bytes, i.e. **624 bits**.
- Many more short implementations have been found by code golfers!<sup>3</sup>

The probability that a random S-box is that **simple**  
is **completely negligible** ( $\leq 2^{-1059}$ ).

<sup>1</sup> Bonnetain, Perrin, Tian. Anomalies and Vector Space Search: Tools for S-Box Reverse-Engineering. <https://ia.cr/2019/528>

<sup>2</sup> Credit to @odzhan on stackexchange.

<sup>3</sup> <https://codegolf.stackexchange.com/questions/186498/>

# Outline

- 1 Standards and S-boxes
- 2 On the S-box of RFC 6986 and 7801
- 3 The Core Issue: the S-Box Generation Process
- 4 Conclusion**

## Conclusion

No secret structure was enforced during construction of the S-box. At the same time, it is obvious that for any transformation a lot of representations are possible (see, for example, a lot of AES S-box representations).

**vs.**

```
p(x){unsigned char*k="0~rFTDVbpPB
vdtfRQ\xacp?\xe2>4\xa6\xe9{z\xe3q
5\xa7\xe8",a=2,l=0,b=17;while(x&&
(l++,a^x))a=2*a^128*29;return l
%b?k[l%b]^k[b+l/b]^b:k[l/b]^188;}
```

- This claim and this fact **cannot** be reconciled.

## Conclusion

No secret structure was enforced during construction of the S-box. At the same time, it is obvious that for any transformation a lot of representations are possible (see, for example, a lot of AES S-box representations).

vs.

```
p(x){unsigned char*k="0~rFTDVbpPB  
vdtfRQ\xacp?\xe2>4\xa6\xe9{z\xe3q  
5\xa7\xe8", a=2, l=0, b=17;while(x&&  
(l++, a^x))a=2*a^a/128*29;return l  
%b?k[l%b]^k[b+l/b]^b:k[l/b]^188;}
```

- This claim and this fact **cannot** be reconciled.
- In my opinion, the designers of these algorithms **have provided misleading information** for the external analysis of their design.

## Conclusion

No secret structure was enforced during construction of the S-box. At the same time, it is obvious that for any transformation a lot of representations are possible (see, for example, a lot of AES S-box representations).

vs.

```
p(x){unsigned char*k="Q~rFTDVbpPB  
vdtfRQ\xacp?\xe2>4\xa6\xe9{z\xe3q  
5\xa7\xe8",a=2,l=0,b=17;while(x&&  
(l++,a^x))a=2*a^a/128*29;return l  
%b?k[l%b]^k[b+l/b]^b:k[l/b]^188;}
```

- This claim and this fact **cannot** be reconciled.
- In my opinion, the designers of these algorithms **have provided misleading information** for the external analysis of their design.
- Security analysis is hard enough with proper information: **there is no good reason** to complicate it further with wrong data!

## Conclusion

No secret structure was enforced during construction of the S-box. At the same time, it is obvious that for any transformation a lot of representations are possible (see, for example, a lot of AES S-box representations).

vs.

```
p(x){unsigned char*k="Q~rFTDVbpPB  
vdtfRQ\xacp?\xe2>4\xa6\xe9{z\xe3q  
5\xa7\xe8",a=2,l=0,b=17;while(x&&  
(l++,a^x))a=2*a^a/128*29;return l  
:b?k[l:b]^k[b+l/b]^b:k[l/b]^188;}
```

- This claim and this fact **cannot** be reconciled.
- In my opinion, the designers of these algorithms **have provided misleading information** for the external analysis of their design.
- Security analysis is hard enough with proper information: **there is no good reason** to complicate it further with wrong data!

⇒ **These algorithms cannot be trusted and I believe they should be deprecated.**

## Conclusion

No secret structure was enforced during construction of the S-box. At the same time, it is obvious that for any transformation a lot of representations are possible (see, for example, a lot of AES S-box representations).

vs.

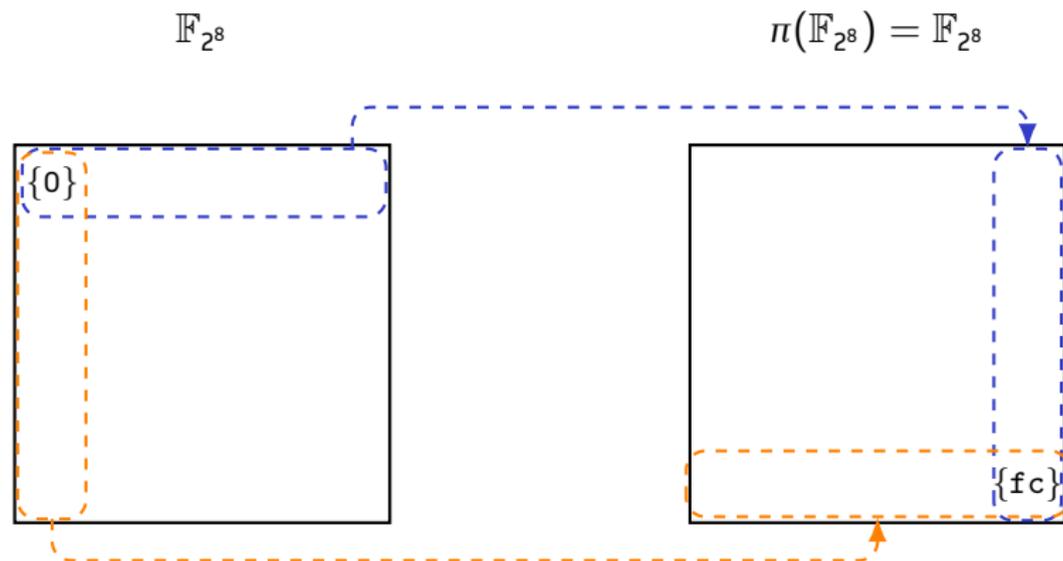
```
p(x){unsigned char*k="Q~rFTDVbpPB  
vdtfRQ\xacp?\xe2>4\xa6\xe9{z\xe3q  
5\xa7\xe8",a=2,l=0,b=17;while(x&&  
(l++,a^x))a=2*a^a/128*29;return l  
?b?k[l?b]^k[b+l/b]^b:k[l/b]^188;}
```

- This claim and this fact **cannot** be reconciled.
- In my opinion, the designers of these algorithms **have provided misleading information** for the external analysis of their design.
- Security analysis is hard enough with proper information: **there is no good reason** to complicate it further with wrong data!

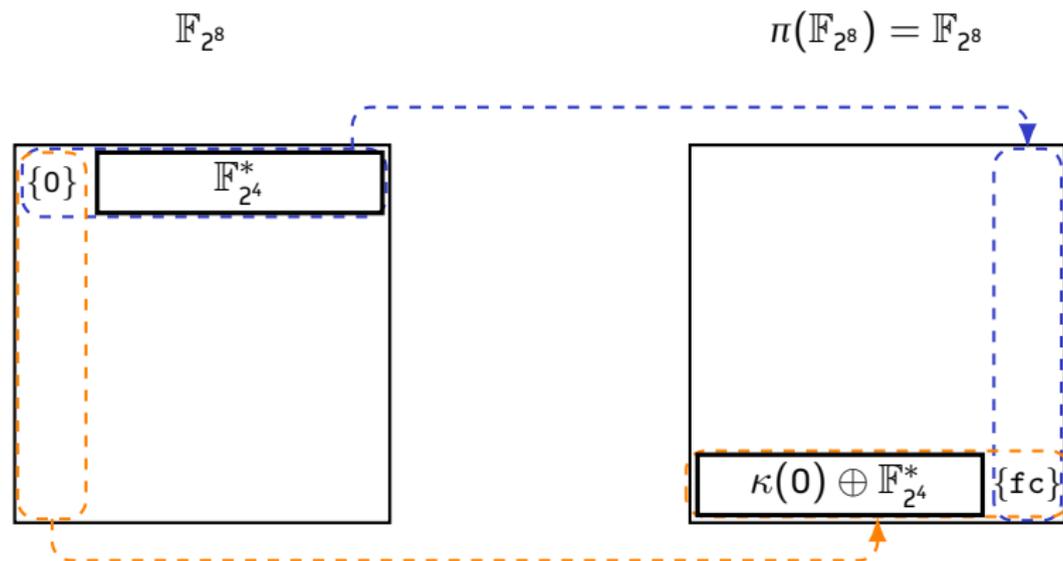
⇒ **These algorithms cannot be trusted and I believe they should be deprecated.**

Thank you!

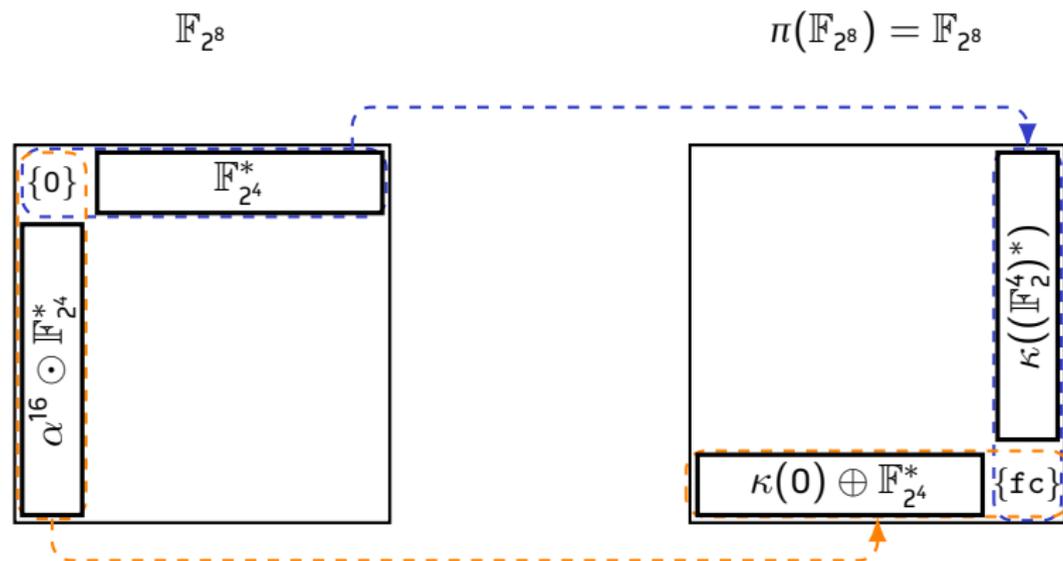
# Cosets to Cosets



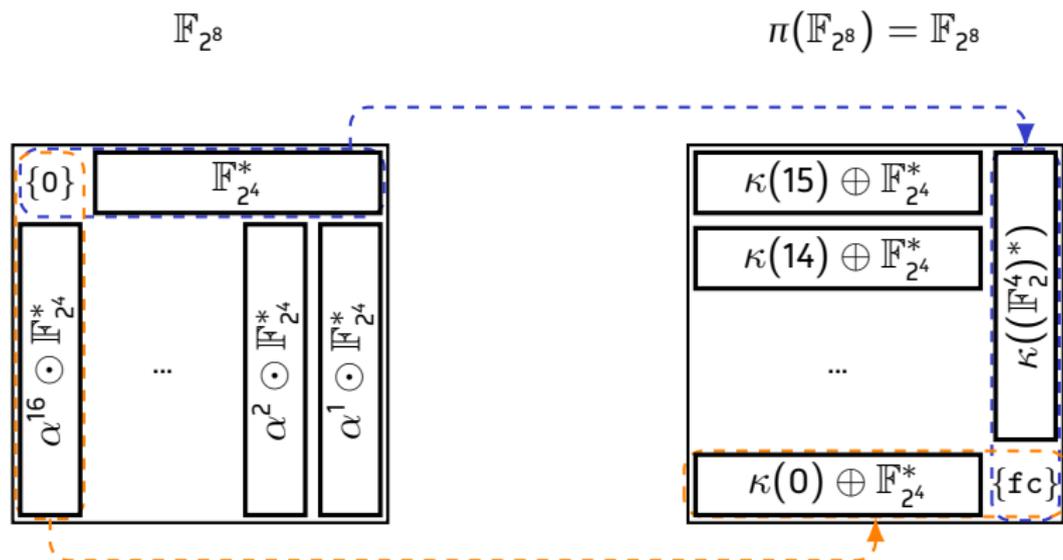
# Cosets to Cosets



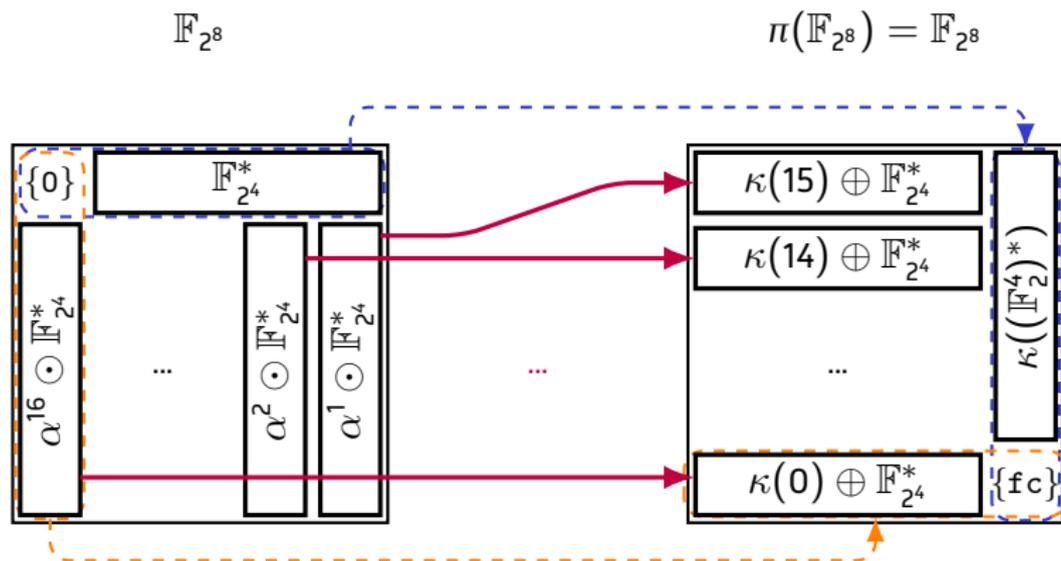
# Cosets to Cosets



# Cosets to Cosets

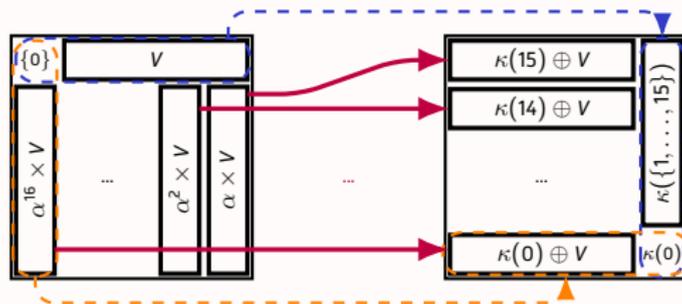


# Cosets to Cosets



# Why it is Worrying

## Russian S-box



## Backdoored S-box

(<https://ia.cr/2016/493>)

