Russian Style (Lack of) Randomness

Léo Perrin, Xavier Bonnetain

Russian Style (Lack of) Randomness

Léo Perrin and Xavier Bonnetain
{leo.perrin,xavier.bonnetain}@inria.fr

Inria

Abstract. It is crucial for a cipher to be trusted that its design be well explained. However, some designers do not publish their design method and instead merely put forward a specification. While this information is sufficient for implementers, the lack of explanation hinders third party cryptanalysis.

In a recent string of papers, Biryukov, Perrin and Udovenko identified increasingly strong patterns in a subcomponent shared by the last two Russian standards in symmetric cryptography, namely the hash function Streebog (GOST R 34.11-2012) and the block cipher Kuznyechik (GOST R 34.12-2015). In this paper, we summarize the latest result of Perrin on this topic and argue that, in light of them, these algorithms must be avoided.

1 Introduction

Block ciphers are at the core of symmetric cryptography, as they are used to encrypt messages, but also to build MACs or hash functions. The most famous example is the AES, which is used to encrypt a very large amount of today's communications.

Formally, a block cipher is a permutation E_k operating on blocks of a fixed size (typically 128 bits) which is parametrized by a secret key k, also of a fixed size (typically 128 or 256 bits). In practice, a block cipher is always defined as multiple iterations of a simple *round function* interleaved with the addition of subkeys derived from the master key k. Following the terminology introduced by Shannon, the round function must provide both *diffusion* and *confusion*. Diffusion means that the output bits must depend on many input bits and confusion means that the mathematical relationship between input bits, output bits and key bits must be complex. In particular, it has to be non-linear.

1.1 S-boxes and their Importance

Non-linearity is usually (though not always) provided by small non-linear function called *S-Boxes* while diffusion is provided by the *linear*

layer which operates on much larger parts of the internal state. The small size of the S-Boxes mean that they are usually specified by their *look-up table (LUT)*, i.e. the array $(S[0], S[1], ..., S[2^m - 1])$ where m is the number of input bits of S, and where typically $m = 4$ or $m = 8$. In practice, the output size is usually the same as the input size.

Knowning this table is sufficient to analyze their *generic* cryptographic properties. We can then combine these with other properties of the linear layer to formally prove that some cryptanalysis techniques will fail when applied to the block cipher. For example, the AES designers introduced the *wide-trail argument* to prove that it is safe from single-trail differential and linear cryptanalysis.

The role of these S-Boxes is to mix their input in a non-linear way. This non-linearity can for example be quantified via the *differential uniformity*. For an S-Box S operating on the set $\mathbb{F}_2^n$ of all n-bit strings, the differential uniformity of S is the maximum number of solutionts S of the equation

$$S(x \oplus a) \oplus S(x) = b$$

for all $a, b \in \mathbb{F}_2$ where $a \neq (0, 0, ..., 0)$.

They must also be such that all outputs depend on all their inputs. In fact, S-box that fail to provide this property [1] have been proved in [2] to be precisely those needed to build a specific type of backdoored block cipher. Such a block cipher E_k would be such that every output bit depends on all input bits. However, it would yield secret linear functions ℓ_1 and ℓ_2 such that $\ell_2(E_k(x))$ does not depend on $\ell_1(x)$. In a stealthy way, this block cipher fails to provide diffusion. This allows anyone with the knowledge of the functions ℓ_1 and ℓ_2 to efficiently attack the cipher.

1.2 Ill-Specified S-Boxes

As we have established, S-Boxes play a crucial role in the security level provided by the algorithms using them. Consequently, it is expected of block cipher designers that they carefully choose these components and justify their choice in the paper describing their algorithm. These explanations will help in several ways. Third party cryptanalysts analysing the security of the block cipher will have their task simplified; implementers may be able to leverage the structure of the S-Box to implement it more efficiently [2]; and potential users will have an increased trust in the

1. In fact, it is sufficient that some linear combinations of the output bits do not depend on some linear combinations of the input bits.

2. See for instance the optimizations allowed by the mathematical structure of the AES S-Box [4].

algorithm: it is much harder to hide a trapdoor in an algorithm when all of its components must have a clear justification.

Unfortunately, some cipher designers do *not* provide such justifications. In particular, the American NSA and its Russian counterpart the FSB do not explain the rationale behind the algorithms they design. Yet, the following algorithms have been included in national and international standards

- the American block cipher Skipjack [11], which used to be a NIST [3] standard but is now deprecated,
- the American block cipher CMEA [12], which was standardized by the TIA [4] and used to secure the control channel (e.g. the transmission of phone numbers) of cell phones in North America,
- the Russian hash function Streebog [7] (GOST R 34.11-2012) which has been a standard in Russia since 2012 and is also the IETF RFC 6986 [6], and
- the block cipher Kuznyechik [8] (GOST R 34.12-2015) which is a Russian standard since 2015 and the IETF RFC 7801 [5]. Kuznyechik is also being considered by ISO/IEC for inclusion as one of their standards.

All these algorithms have been standardized and used despite the fact that their designers did not provide any information about their design. In particular, they all use unexplained S-Boxes. In this context, it is crucial to be able to recover the design criteria and/or the structure used to build an S-Box S given only its LUT.

Both Russian algorithms (Streebog and Kuznyechik) use the same S-box, π. It is an 8-bit permutation which was only specified via its look-up table (see Figure 1). In the next section, we summarize the results that were obtained by cryptanalysts about this S-box and, in particular, how it has a hidden structure which could have negative consequences regarding the security of the algorithms using it.

2 Reverse-Engineering the Russian S-box

Reverse-engineering an S-box is a challenging task as it is a priori very difficult to know if a decomposition is the one that was intended by its designers. Biryukov et al. first identified a new generic reverse-engineering technique, the TU-decomposition, which they could successfully apply to π [3]. Then, Perrin and Udovenko found a relation between π and

3. National Institute for Standards and Technology.
4. Telecommunications Industry Association.

π' = (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).

Fig. 1. A screen capture of the specification of the block cipher Streebog [8, page 4].

a discrete logarithm. Though it showed that there was more to π than previously thought, their decomposition left them unconvinced as some of its components were somewhat inelegant [10]. Finally, building upon these results, Perrin identified a structure which is likely to be the one intended by the designers of π [9].

Let us describe this last decomposition. Perrin showed that π can be evaluated by the following equations:

$$\begin{cases} \pi(0) & = \kappa(0) \\ \pi\left(\alpha^{17j}\right) & = \kappa(16-j), \quad \text{where} \quad 0 < j \leq 15 \\ \pi\left(\alpha^{i+17j}\right) & = \kappa(16-i) \oplus \left(\alpha^{17}\right)^{s(j)}, \quad \text{where} \quad 0 \leq j < 15, 0 < i \leq 16 \ , \end{cases}$$

where

- the finite field $\mathrm{GF}(2^8)$ is defined as $\mathbb{F}_2[X]/p(X)$ with $p(X) = X^8 + X^4 + X^3 + X^2 + 1$,
- α is a root of p and thus [5] a multiplicative generator of $\mathrm{GF}(2^8)^*$,
- s is a permutation of $\{0, 1, 2, ..., 14\}$ given in Table 1, and
- $\kappa : \{0,1\}^4 \to \mathrm{GF}(2^8)$ is a linear permutation such that $\kappa(x) = \Lambda(x) \oplus \texttt{0xfc}$ and such that $\Lambda : \{0,1\}^4 \to \mathrm{GF}(2^8)$ is the linear function defined by

$$\Lambda(\texttt{0x1}) = \texttt{0x12},\ \Lambda(\texttt{0x2}) = \texttt{0x26},\ \Lambda(\texttt{0x4}) = \texttt{0x24},\ \Lambda(\texttt{0x8}) = \texttt{0x30}\ .$$

This highly structured decomposition is unlike anything else in the literature. It was also kept secret by the designers of π. Still, it is likely to be the one they used: it is very simple [6] and the number of permutations

5. It also holds that α^{17} is a multiplicative generator $\mathrm{GF}(2^4)^*$.
6. Especially when compared to the previous two decompositions of [3] and [10].

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$s(x)$	0	12	9	8	7	4	14	6	5	10	2	11	1	3	13

Table 1. The subfunction s.

with such a structure is negligible. In fact, Perrin established that the probability for a random permutation to have such a structure is equal to about $2^{82.6-1684} \approx 2^{-1601}$. For comparison, the probability that an 8-bit permutation is affine is equal to $2^{70.2-1684} \approx 2^{-1618}$ and the probability to win the Loto [7] is about $2^{-24.2}$. Thus, the probability for a permutation picked uniformly at random to have a structure similar to that of π is comparable to the probability of gaining the Loto 66 times in row! Hence, it is likely to be the structure originally intended by the designers of this S-box.

Besides, this structure also has some strange cryptographic properties. The field $\mathrm{GF}(2^4)$ is contained in $\mathrm{GF}(2^8)$ and π has a specific interaction with the cosets of this subfield. If $i > 0$, then

$$\{\pi(\alpha^i \times x), x \in \mathrm{GF}(2^4), x \neq 0\} \;=\; \{\kappa(16-i) \oplus x, x \in \mathrm{GF}(2^4), x \neq 0\} \quad (1)$$

where the multiplication is done in $\mathrm{GF}(2^8)$. In other words, π maps the partition of $\mathrm{GF}(2^8)$ into multiplicative cosets of $\mathrm{GF}(2^4)^*$ to its partition into additive cosets of $\mathrm{GF}(2^4)^*$.

The other main component of the hash function Streebog is a 64-bit linear permutation originally specified as a 64×64 binary matrix. It is in fact a 16×16 matrix with coefficients in $\mathrm{GF}(2^8)$ where this field is defined by the same polynomial as in π. Hence, this component interacts in a way which is yet to be fully understood with the partitions in Equation (1). Much like the structure of π, the structure of this component was kept secret and had to be reverse-engineered—though it was much simpler.

In light of these results, new security analyses of Kuznyechik and even more so of Streebog are necessary. We have yet to perfectly understand the consequences of these partition-preserving properties. Had the designers of the Russian ciphers disclosed the structures they used, cryptographers could have focused on this analysis much sooner. Of course, their aim when hiding those may have been to try and prevent such an analysis.

7. The French lotery is won if 5 numbers in $\{1, 2, ..., 49\}$ and one in $\{1, ..., 10\}$ are chosen correctly, an event with probability $(49 \times 48 \times 47 \times 46 \times 45 \times 10/5!)^{-1} \approx (19 \times 10^6)^{-1} \approx 2^{-24.2}$.

3 Conclusion

The last symmetric cryptographic algorithms standardized in Russia (and later included in some IETF RFC as well as an ISO/IEC standard, possibly followed by a second one soon) only had their specifications published. Their authors did not disclose their design rationale which, in and on itself, should warrant caution. Through a series of papers, Biryukov, Perrin and Udovenko have progressively unlocked the secrets of one of the main components of these algorithms. In particular, the latest results of Perrin show that the designers of Streebog and Kuznyechik purposefully hid a structure in this component. This structure is very strong, very uncommon and interacts in a non-trivial way with the other main component of Streebog.

In light of these results, we urge security professionals to avoid these algorithms. More generally, we invite them to keep in mind that, ultimately, both national and international standards in cryptography are seldom chosen transparently and that the final decision is rarely made by cryptographers (see also [1]). Hence, we ugre practitioners to carefully check where the algorithms they plan to use come from, *even if they are standards*.

Acknowledgements This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement no. 714294 - acronym QUASYModo).

References

1. Tomer Ashur and Atul Luykx. An account on the ISO/IEC standardization of Simon and Speck. Presentation at *Real World Crypto*, 2019.
2. Arnaud Bannier. *Combinatorial Analysis of Block Ciphers With Trapdoors*. PhD thesis, 2017.
3. Alex Biryukov, Léo Perrin, and Aleksei Udovenko. Reverse-engineering the s-box of streebog, kuznyechik and stribobr1. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016*, pages 372–402, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.
4. D. Canright. A very compact s-box for aes. In Josyula R. Rao and Berk Sunar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2005*, pages 441–455, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.
5. Vasily Dolmatov. GOST R 34.12-2015: Block cipher "kuznyechik". *RFC*, 7801:1–14, 2016.

6. Vasily Dolmatov and Alexey Degtyarev. GOST R 34.11-2012: Hash function. *RFC*, 6986:1–40, 2013.
7. Federal Agency on Technical Regulation and Metrology. Information technology – data security: Hash function. English description available at `https://www.streebog.net/`, 2012.
8. Federal Agency on Technical Regulation and Metrology. Information technology – data security: Block ciphers. English version available at `https://tc26.ru/upload/iblock/fc9/GOST_R_34_12_2015_ENG.pdf`, 2015.
9. Léo Perrin. Partitions in the S-box of Streebog and Kuznyechik. To appear (IACR ToSC), 2018.
10. Léo Perrin and Aleksei Udovenko. Exponential s-boxes: a link between the s-boxes of belt and kuznyechik/streebog. *IACR Transactions on Symmetric Cryptology*, 2016(2):99–124, Feb. 2017.
11. U.S. Department Of Commerce/National Institute of Standards and Technology. Skipjack and KEA algorithms specifications, v2.0, 1998. `http://csrc.nist.gov/groups/ST/toolkit/documents/skipjack/skipjack.pdf`.
12. David Wagner, Bruce Schneier, and John Kelsey. Cryptanalysis of the cellular message encryption algorithm. In Burton S. Kaliski, editor, *Advances in Cryptology — CRYPTO '97*, pages 526–537, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg.