# A feebly secure trapdoor function*

Edward A. Hirsch and Sergey I. Nikolenko

Steklov Institute of Mathematics at St.Petersburg, Russia

**Abstract.** In 1992, A. Hiltgen [1] provided the first constructions of provably (slightly) secure cryptographic primitives, namely *feebly one-way functions*. These functions are provably harder to invert than to compute, but the complexity (viewed as circuit complexity over circuits with arbitrary binary gates) is amplified by a constant factor only (with the factor approaching 2).
In traditional cryptography, one-way functions are the basic primitive of private-key and digital signature schemes, while public-key cryptosystems are constructed with trapdoor functions. We continue Hiltgen's work by providing an example of a *feebly trapdoor function* where the adversary is guaranteed to spend more time than every honest participant by a constant factor of $\frac{25}{22}$.

## 1 Introduction

Numerous public-key cryptographic protocols have been devised over the years, and yet there exists *not a single proof* of their security: neither an unconditional proof (that would necessarily imply P $\neq$ NP) nor a proof based on standard (not problem-specific) structural complexity assumptions. While universal primitives for one-way functions [2, 3] and public-key cryptosystems are known [4] (see also [5]), they give no connection to the core assumptions of traditional structural complexity theory; more, the asymptotic nature of polynomial-time reductions leaves no hope for confidence that a particular scheme cannot be broken at a particular key length. In general, it appears like there is still a very long way to go before cryptography can claim any *provably* secure public-key construction.

If we are unable to prove a superpolynomial gap between the complexities of honest parties and adversaries, can we prove at least *some* gap? In 1992, Alain Hiltgen [1] managed to present a function that is *twice* harder to invert than to compute. His example is a linear function over $GF(2)$ with a matrix that has few non-zero entries while the inverse matrix has many non-zero entries; the complexity gap follows by a simple argument of Lamagna and Savage [6, 7]: every bit of the output depends non-idly on many variables and all these bits correspond to different functions, hence a lower bound on the complexity of computing them all together. The model of computation here is the most general one: the number of gates in a Boolean circuit that uses arbitrary binary Boolean gates. Note that little more could be expected for this model at present, since known lower bounds here are linear in the number of inputs [8, 9].

In this work, we construct another feebly secure cryptographic primitive: namely, a feebly trapdoor function. Of course, in order to obtain the result, we have to prove a

* This research was partially supported by the Russian Foundation for Basic Research (RFBR), grants 08-01-00640-a, 08-01-00649, 09-01-00784-a, the Dynasty Foundation, the President of RF grant for leading scientific schools support NSh 4392.2008.1, and the Fundamental research program of the Russian Academy of Sciences.

lower bound on the circuit complexity of a certain function; we use the gate elimination technique from circuit complexity of the eighties. More formally, the complexity of inverting (decryption) without the use of trapdoor information in our construction is at least $\frac{25}{22}$ times greater than the complexities of honest encryption, decryption, and key generation. We also provide hardness amplification results that give exponential guarantees on the success probability for weaker attackers. In Section 2.1, we give basic definitions. Section 2.2 does preparational work, establishing some combinatorial properties of the matrices representing hard function candidates. Section 2.3 reviews the basic method of gate elimination, which we use to prove lower bounds on complexity, and applies it to the feeble security setting. Finally, Sections 3.1 and 3.2 present the construction of a feebly secure trapdoor function, Section 3.3 proves hardness amplification results, and Section 4 lists possible directions for further research.

## 2 Preliminaries

### 2.1 Basic definitions

Let us denote by $\mathbb{B}_{n,m}$ the set of all $2^{m2^n}$ functions $f : \mathbb{B}^n \to \mathbb{B}^m$, where $\mathbb{B} = \{0, 1\}$ is the field of two elements. Our computational model is Boolean circuits with arbitrary binary gates. A circuit is a directed acyclic graph with in-degree zero (these nodes are called *circuit inputs*, or *variables*) and two (these nodes are called *gates*). Every gate is labelled by a binary Boolean function (any of the 16 functions in $\mathbb{B}_{2,1}$). Some gates are marked as *outputs* (to avoid extra negation gates, we may also define an output as the negation of the value obtained in a gate; to avoid identity gates, we also allow to define an output as the value of a variable or the negation of it). A circuit with $n$ inputs and $m$ outputs naturally computes a Boolean function in $\mathbb{B}_{n,m}$. We denote the size of a circuit $c$ by $C(c)$. The *circuit complexity* (or simply *complexity*) of a function $f$, denoted (slightly abusing notation) by $C(f)$, is the smallest number of gates in a circuit computing $f$ (such circuit is called an *optimal* circuit for $f$): $C(f) = \min_{c:\forall x\, c(x)=f(x)} C(c)$. We may safely assume that every gate of this circuit depends on both inputs, i.e., there are no constant functions and no unary functions Id and NOT, because such gates can be easily eliminated from the circuit without increasing the number of the gates.

Following Hiltgen, for every injective $f_n \in \mathbb{B}_{n,m}$ we can define its *measure of one-wayness* $M_F(f_n) = \frac{C(f_n^{-1})}{C(f_n)}$. Hiltgen's work was to find sequences of functions $f = \{f_n\}_{n=1}^{\infty}$ with large asymptotic constant $\liminf_{n\to\infty} M_F(f_n)$, which Hiltgen calls $f$'s *order of one-wayness*. We will discuss his results in more detail in Section 2.3. We now extend this definition in order to capture feebly secure trapdoor functions. Since we are interested in constants here, we must pay attention to all the details.

**Definition 1.** *A* feebly trapdoor candidate *is a sequence of triples of circuits* $\mathcal{C} = \{(\mathrm{Key}_n, \mathrm{Eval}_n, \mathrm{Inv}_n)\}_{n=1}^{\infty}$ *where:*

- $\{\mathrm{Key}_n\}_{n=1}^{\infty}$ *is a family of sampling circuits* $\mathrm{Key}_n : \mathbb{B}^n \to \mathbb{B}^{\mathrm{pi}(n)} \times \mathbb{B}^{\mathrm{ti}(n)}$,
- $\{\mathrm{Eval}_n\}_{n=1}^{\infty}$ *is a family of evaluation circuits* $\mathrm{Eval}_n : \mathbb{B}^{\mathrm{pi}(n)} \times \mathbb{B}^{m(n)} \to \mathbb{B}^{c(n)}$, *and*
- $\{\mathrm{Inv}_n\}_{n=1}^{\infty}$ *is a family of inversion circuits* $\mathrm{Inv}_n : \mathbb{B}^{\mathrm{ti}(n)} \times \mathbb{B}^{c(n)} \to \mathbb{B}^{m(n)}$

*such that for every security parameter* $n$*, every seed* $s \in \mathbb{B}^n$*, and every input* $\mathrm{m} \in \mathbb{B}^{m(n)}$*,*

$$\mathrm{Inv}_n(\mathrm{Key}_{n,2}(s), \mathrm{Eval}_n(\mathrm{Key}_{n,1}(s), \mathrm{m})) = \mathrm{m},$$

*where* $\mathrm{Key}_{n,1}(s)$ *and* $\mathrm{Key}_{n,2}(s)$ *are the first* $\mathrm{pi}(n)$ *bits ("public information") and the last* $\mathrm{ti}(n)$ *bits ("trapdoor information") of* $\mathrm{Key}_n(s)$*, respectively.*

We call this function a "candidate" because the definition does not imply any security, it merely sets up the dimensions and provides correct inversion. In our constructions, $m(n) = c(n)$ and $\mathrm{pi}(n) = \mathrm{ti}(n)$. To find how secure a function is, we introduce the notion of a break. Informally, an adversary should invert the function without knowing the trapdoor information. We introduce break as inversion with probability greater than $\frac{3}{4}$. (We later investigate security against adversaries having smaller success probabilities, too.) We denote by $C_{3/4}(f)$ the minimal size of a circuit that correctly computes a function $f \in \mathbb{B}_{n,m}$ on more than $\frac{3}{4}$ of its inputs (of length $n$). Obviously, $C_{3/4}(f) \le C(f)$ for all $f$.

**Definition 2.** *A circuit* $N$ breaks *a feebly trapdoor candidate* $\mathcal{C} = \{\mathrm{Key}_n, \mathrm{Eval}_n, \mathrm{Inv}_n\}$ *on seed length* $n$ *if for uniformly chosen seeds* $s \in \mathbb{B}^n$ *and inputs* $m \in \mathbb{B}^{m(n)}$

$$\Pr_{(s,m)\in U}\left[N(\mathrm{Key}_{n,1}(s), \mathrm{Eval}_n(\mathrm{Key}_{n,1}(s), m)) = m\right] > \frac{3}{4}.$$

*Remark 1.* In fact, in what follows we prove a stronger result: we prove that no circuit (of a certain size) can break our candidate *for any random seed* $s$, that is, for every seed $s$, every adversary fails with probability at least $1/4$.

For a trapdoor function to be secure, circuits that break the function should be larger than the circuits computing it.

**Definition 3.** *A feebly trapdoor candidate* $\mathcal{C} = \{\mathrm{Key}_n, \mathrm{Eval}_n, \mathrm{Inv}_n\}$ *has* order of security $k$ *if for every sequence of circuits* $\{N_n\}_{n=1}^{\infty}$ *that break* $f$ *on every input length* $n$*,*

$$\liminf_{n\to\infty} \min\left\{\frac{C(N_n)}{C(\mathrm{Key}_n)}, \frac{C(N_n)}{C(\mathrm{Eval}_n)}, \frac{C(N_n)}{C(\mathrm{Inv}_n)}\right\} \ge k.$$

*Remark 2.* One could consider key generation as a separate process and omit its complexity from the definition of the order of security. However, we prove our results for the definition stated above as it makes them stronger.

*Remark 3.* Let us note explicitly that we are talking about *one-time* security. An adversary can amortize his circuit complexity on inverting a feebly trapdoor candidate for the second time for the same seed, for example, by computing the trapdoor information and successfully reusing it. Thus, in our setting one has to pick a new seed for every input.

In the next sections, we develop our construction of a feebly trapdoor function (that is, a feebly trapdoor candidate with nontrivial order of security).

### 2.2 Hard matrices

All our constructions are based on a linear function $f : \mathbb{B}^n \to \mathbb{B}^n$ shown by A. Hiltgen [1] to be feebly one-way of order 3/2. We restrict ourselves to the case when $n \equiv 0 \pmod 4$. Note that Definition 3 carries through this restriction: for $n \not\equiv 0 \pmod 4$ one

can simply consider circuit with input size equal to the lowest multiple of 4 greater than $n$.

In what follows, all computations are done over $\mathbb{F}_2$. We use standard matrix notation: $\boldsymbol{e}_k$ is the identity $k \times k$ matrix, $\mathbf{0}_k$ is the zero $k \times k$ matrix, $\boldsymbol{e}_{ij}$ is a matrix with a single non-zero element in position $(i, j)$, $\boldsymbol{e}_{i*}$ is a matrix where the $i^{\text{th}}$ row consists of 1's, and all other elements are zero, $\mathbf{1}_k$ is the $k \times k$ matrix filled with 1's, $\boldsymbol{u}_k$ is the upper triangular $k \times k$ matrix ($u_{ij} = 1 \Leftrightarrow i < j$), $\boldsymbol{l}_k$ is the lower triangular $k \times k$ matrix ($l_{ij} = 1 \Leftrightarrow i > j$), and $\boldsymbol{m}_\pi$ is the permutation matrix for $\pi$ ($m_{ij} = 1 \Leftrightarrow j = \pi(i)$). By $\boldsymbol{e}$, $\mathbf{0}$, $\mathbf{1}$, $\boldsymbol{u}$, and $\boldsymbol{l}$ without subscripts we denote the correspondent matrices of dimension $\frac{n}{2} \times \frac{n}{2}$. We also set $\sigma_n$ to be the cyclic permutation $(1\ 2\ \ldots\ n)$.

In this notation the matrix of the Hiltgen's function $f$ is $A = \boldsymbol{e}_n + \boldsymbol{m}_{\sigma_n} + \boldsymbol{e}_{n,\frac{n}{2}+1}$. We are also interested in the $n \times 2n$ matrix $\mathfrak{A}$ consisting of $A^{-2}$ and $A^{-1}$ stacked together: $\mathfrak{A} = \left(A^{-2}\ A^{-1}\right)$.

The following lemma can be easily verified by direct calculation.

**Lemma 1.** *Let $n = 4k$ for some $k \in \mathbb{N}$. Then*

$$A^{-1} = \mathbf{1}_n + \begin{pmatrix} \boldsymbol{e} + \boldsymbol{u} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{l} \end{pmatrix}, \quad A^{-2} = \mathbf{1}_n\boldsymbol{u}_n + \boldsymbol{u}_n\mathbf{1}_n + \begin{pmatrix} \boldsymbol{e} + \boldsymbol{u}^2 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{l}^2 \end{pmatrix}.$$

**Lemma 2.** *Let $n = 4k$ for some $k \in \mathbb{N}$. The following statements hold.*

1. *All columns of $\mathfrak{A}$ (and, hence, $A^{-1}$) are different.*
2. *Each row of $A^{-1}$ (resp., $\mathfrak{A}$) contains at least $\frac{n}{2}$ (resp., $\frac{5n}{4}$) nonzero entries.*
3. *After eliminating all but two (resp., all but five) columns of $A^{-1}$ (resp., $\mathfrak{A}$) there remains at least one row with two nonzero entries.*

*Proof.* Let us first interpret the results of Lemma 1. Each row of $A$ contains two ones (on the diagonal and to the right) except for the last row that has three ones, in positions $(n, 1)$, $(n, \frac{n}{2} + 1)$, and $(n, n)$. Each row of $A^{-1}$ has at least $\frac{n}{2}$ non-zero elements (ones), and the $(\frac{n}{2} + 1)^{\text{th}}$ row does not contain a single zero.

The $A^{-2}$ matrix also has lots of ones: $(\mathbf{1}_n\boldsymbol{u}_n + \boldsymbol{u}_n\mathbf{1}_n)$ is an $n \times n$ matrix filled with zeroes and ones chequered, since

$$\begin{aligned} (\mathbf{1}_n\boldsymbol{u}_n)_{ij} = 1 \quad &\Leftrightarrow \quad j \equiv 1 \pmod 2, \\ (\boldsymbol{u}_n\mathbf{1}_n)_{ij} = 1 \quad &\Leftrightarrow \quad i \equiv 0 \pmod 2. \end{aligned}$$

Moreover,

$$\begin{aligned} (\boldsymbol{e} + \boldsymbol{u}^2)_{ij} = 1 \quad &\Leftrightarrow \quad j > i \quad \text{and} \quad i + j \equiv 0 \pmod 2, \\ (\boldsymbol{l}^2)_{ij} = 1 \quad &\Leftrightarrow \quad i > j \quad \text{and} \quad i + j \equiv 0 \pmod 2, \end{aligned}$$

and thus $A^{-2}$ has two triangular blocks filled with ones: for $1 \le i \le j \le \frac{n}{2}$ and for $\frac{n}{2} + 1 < j < i \le n$. Thus, each row of $A^{-2}$ contains at least $\frac{n}{2}$ ones; moreover, its triangular blocks consisting of ones coincide with the triangular blocks of $A^{-1}$ filled with zeroes, and the rest is covered with zeroes and ones chequered.

The first statement is obvious.

The $i^{\text{th}}$ row of $A^{-1}$ contains $\frac{n}{2} + i$ non-zero elements for $i \le \frac{n}{2}$ and $\frac{n}{2} + n - i$ non-zero elements for $i \ge \frac{n}{2}$. Thus, the second statement holds for $A^{-1}$. At the same time, the $i^{\text{th}}$ row of $A^{-2}$ contains at least $\frac{3n}{4} - \frac{i}{2}$ non-zero elements for $i \le \frac{n}{2}$ and

at least $\frac{n}{2} + \frac{1}{2}(i - \frac{n}{2} - 1)$ non-zero elements for $i > \frac{n}{2}$. Therefore, the $i^{\text{th}}$ row of $A^{-2}$ contains at least $\frac{n}{2} + i + \frac{3n}{4} - \frac{i}{2} = \frac{5n}{4} + \frac{i}{2}$ nonzero entries for $i \leq \frac{n}{2}$ and at least $\frac{n}{2} + n - i + \frac{n}{2} + \frac{1}{2}(i - \frac{n}{2} - 1) = \frac{7n}{4} - \frac{1}{2}(i - 1) \geq \frac{5n}{4}$ nonzero entries for $i \geq \frac{n}{2}$.

Let us now prove the third claim. Since $A^{-1}$ has a row that contains only nonzero entries, all but one columns of this matrix should be eliminated to leave just one nonzero entry. The same holds for the left part of the matrix $A^{-2}$ (see its first row). The same holds for the right part of the matrix $A^{-2}$ without the last column (see its last row). □

### 2.3 Gate elimination

In this section, we first briefly remind about Hiltgen's methods and then introduce gate elimination as the primary technique for proving our bounds. Hiltgen proved all his bounds with the following very simple argument due to Lamagna and Savage.

**Proposition 1 ([6, 7]; [1, Theorems 3 and 4]).**

1. *Suppose that $f : \mathbb{B}^n \to \mathbb{B}$ depends non-idly on each of its $n$ variables, that is, for every $i$ there exist values $a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n \in \mathbb{B}$ such that $f(a_1, \ldots, a_{i-1}, 0, a_{i+1}, \ldots, a_n) \neq f(a_1, \ldots, a_{i-1}, 1, a_{i+1}, \ldots, a_n)$. Then $C(f) \geq n - 1$.*
2. *Let $f = (f^{(1)}, \ldots, f^{(m)}) : \mathbb{B}^n \to \mathbb{B}^m$, where $f^{(k)}$ is the $k^{\text{th}}$ component of $f$. If the $m$ component functions $f^{(i)}$ and their negations[1] are pairwise different and each of them satisfies $C(f^{(i)}) \geq c \geq 1$ then $C(f) \geq c + m - 1$.*

Hiltgen counted the minimal complexity of computing one bit of the input (e.g. since each row of $A^{-1}$ has at least $\frac{n}{2}$ nonzero entries, the minimal complexity of each component of $A^{-1}\boldsymbol{y}$ is $\frac{n}{2}$) and thus produced lower bounds on the complexity of inverting the function (e.g. the complexity of computing $A^{-1}\boldsymbol{y}$ is $\frac{n}{2} + n - 2 = \frac{3n}{2} - 2$).

Besides, in cryptography it is generally desirable to prove not only worst-case bounds, but also that an adversary is unable to invert the function on a substantial fraction of inputs. Indeed, for each of the matrices constructed here, any circuit using less than the minimal necessary number of gates inverts it on less than $\frac{3}{4}$ of the inputs. In Hiltgen's works, this fact followed from a very simple observation (which was not even explicitly stated).

**Lemma 3.** *Consider a function $f = \bigoplus_{i=1}^{n} x_i$. For any $g$ that depends on only $m < n$ of these variables, $\Pr_{x_1,\ldots,x_n}[f(x_1, \ldots, x_n) = g(x_{i_1}, \ldots, x_{i_m})] = \frac{1}{2}$.*

*Proof.* Since $m < n$, there exists an index $j \in 1..n$ such that $g$ does not depend on $x_j$. This means that for every set of values of the other variables, whatever the value of $g$ is, for one of the values of $x_j$ $f$ agrees with $g$, and on the other value $f$ differs from $g$. This means that $f$ differs from $g$ on precisely $\frac{1}{2}$ of the inputs. □

The argument suffices for Hiltgen's feebly one-wayness result for the square matrix $A^{-1}$: first we apply the first part of Proposition 1 and see that every output has complexity at least $\frac{n}{2} - 1$, and then the second part of Proposition 1 yields the necessary bound of $\frac{3n}{2} - 1$. Moreover, if a circuit has less than the necessary number of gates, one

[1] Note that we had to add the requirement $f^{(i)} \neq \neg f^{(j)}$ to the original statement of Lamagna and Savage because we do not count the negation of an output as a separate gate.

of its outputs inevitably depends on less than the necessary number of input variables, which, by Lemma 3, gives the necessary $\frac{1}{2}$ error rate.

However, in this paper we also use non-square matrices, and it turns out that a similar simple argument does not provide sufficient bounds for our matrices. Therefore, we use a different way of proving lower bounds, namely *gate elimination* that has been previously used for every lower bound in "regular" circuit complexity [9]. The basic idea of this method is to use the following inductive argument. Consider function $f$ and substitute some value $c$ for some variable $x$ thus obtaining a circuit for the function $f|_{x=c}$. The circuit can be simplified, because the gates that had this variable as inputs become either unary (recollect that the negation can be embedded into subsequent gates) or constant (in this case we can proceed to eliminating subsequent gates). The important case here is when the gate is non-linear, such as an AND or an OR gate. In this case it is always possible to choose a value for an input of such gate so that this gate becomes a constant. One then proceeds by induction as long as it is possible to find a suitable variable that eliminates many enough gates. Evidently, the number of eliminated gates is a lower bound on the complexity of $f$.

First, we prove a prerequisite to the master lemma.

**Lemma 4.** *Let $t \geq 0$. Assume that $\chi : \mathbb{B}^{v(n)} \to \mathbb{B}^n$ is a linear function with matrix $X$ over $GF(2)$. Assume also that all columns of $X$ are different, there are no zero rows in $X$, and after removing any $t$ columns of $X$, the matrix still has at least one row containing at least two nonzero entries. Then $C(\chi) \geq t+1$ and, moreover, no circuit with less than $t+1$ gates can compute $\chi$ on more than $\frac{1}{2}$ of the inputs.*

*Proof.* We argue by induction on $t$. For $t = 0$ the statement is obvious: the value of a circuit with no gates[2] cannot agree on more than $\frac{1}{2}$ of the input assignments with a linear function essentially depending on two variables.

Let now $t \geq 1$ and consider an optimal circuit of size less than $t+1$ implementing $\chi$ on more than $\frac{1}{2}$ of the inputs. Let us fix a topological order on its nodes. We denote the actual function this circuit implements by $h$ (it does not need to be linear, but does have to agree with $\chi$ on more than $\frac{1}{2}$ of the inputs).

Consider the topmost gate $g$ in this order. Since $g$ is topmost, its incoming edges come from the inputs of the circuit, denote them by $x$ and $y$. To eliminate a gate, we simply substitute a value to $x$; substituting a value for one variable is equivalent to removing a column from the matrix, and it reduces $t$ by at most 1.

To invoke the induction hypothesis, it remains to note that if $h$ agrees with $\chi$ on more than $\frac{1}{2}$ of the inputs, then either $h|_{x=0}$ or $h|_{x=1}$ agrees with the corresponding restriction of $\chi$ on more than $\frac{1}{2}$ of the remaining inputs. Thus, if $h$ did compute $\chi$ on more than $\frac{1}{2}$ of the inputs, substituting this value of $x$ into $h$ would yield a function of $n-1$ inputs that contradicted the induction hypothesis. Thus, substituting this value of $x$ into $\chi$ yields a function of $n-1$ inputs satisfying the conditions of the theorem with parameter $t-1$, and this function can be computed by a circuit of size less than $t-1$, which contradicts the induction hypothesis. □

The following is a "master" lemma that we will apply to our matrices.

**Lemma 5.** *Let $t \geq u \geq 1$. Assume that $\chi : \mathbb{B}^{v(n)} \to \mathbb{B}^n$ is a linear function with matrix $X$ over $GF(2)$. Assume also that all columns of $X$ are different, every row of*

[2] Recall that it can still compute unary functions.

*$X$ has at least $u$ nonzero entries, and after removing any $t$ columns of $X$, the matrix still has at least one row containing at least two nonzero entries. Then $C(\chi) \geq u + t$ and, moreover, $C_{3/4}(\chi) \geq u + t$.*

*Proof.* This time, we argue by induction on $u$.

The induction base ($u = 1$) follows from Lemma 4.

Let now $u \geq 2$ and consider an optimal circuit of size less than $u + t$ implementing $\chi$ on more than $\frac{3}{4}$ of the inputs and fix a topological order on its nodes. We denote the actual function this circuit implements by $h$ (it does not need to be linear, but does have to agree with $\chi$ on more than $\frac{3}{4}$ of the inputs).

Consider the topmost gate $g$ in this order. Since $g$ is topmost, its incoming edges come from the inputs of the circuit, denote them by $x$ and $y$. By Lemma 3, neither of its input variables can be marked as an output, because for $u \geq 2$ each row has at least two variables.

The following possibilities exhaust all possible cases.

1. One of the input variables of $g$, say $x$, enters some other gate. In this case by setting $x$ to any constant we can eliminate at least 2 gates. To invoke the induction hypothesis, it remains to note that if $h$ agrees with $\chi$ on more than $\frac{3}{4}$ of the inputs, then either $h|_{x=0}$ or $h|_{x=1}$ agrees with the corresponding restriction of $\chi$ on more than $\frac{3}{4}$ of the remaining inputs. Thus, substituting this value of $x$ into $\chi$ yields a function of $n - 1$ inputs satisfying the conditions of the theorem with parameters $u - 1$ and $t - 1$, and this function can be computed by a circuit of size less than $u + t - 2$, which contradicts the induction hypothesis.
2. Neither $x$ nor $y$ enters any other gate. In this case, $h$ is a function of neither $x$ nor $y$ but only $g(x, y)$ (if any); we show that this is impossible. Note that $\chi$ depends on $x$ and $y$ separately; in particular, for one of these variables, say $x$, there exists an output gate[3] $\chi_i$ that depends only on $x$: $\chi_i = x \oplus \bigoplus_{z \in Z} z$, where $y \notin Z$. Since every gate of an optimal circuit essentially depends on both its arguments, there exist values $a$ and $b$ such that $g(0, a) = g(1, b)$. Thus, for every assignment of the remaining variables $h_i \neq \chi_i$ either for $x = 0,\ y = a$ or for $x = 1,\ y = b$, which means that $\chi$ and $h$ disagree on at least $\frac{1}{4}$ of all assignments. □

In what follows we will also use block-diagonal matrices. Intuition hints that joint computation of two functions that have different inputs should be as hard as computing them separately (thus, the lower bound should be the sum of respective lower bounds). However, for certain functions it is not the case, as seen in [9, Section 10.2] We show it for our particular case.

**Lemma 6.** *Assume that a linear function $\zeta$ is determined by a block diagonal matrix*

$$\zeta\left(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(m)}\right) = \begin{pmatrix} X_1 & 0 & \ldots & 0 \\ 0 & X_2 & \ldots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \ldots & X_m \end{pmatrix} \begin{pmatrix} \boldsymbol{x}^{(1)} \\ \boldsymbol{x}^{(2)} \\ \vdots \\ \boldsymbol{x}^{(m)} \end{pmatrix},$$

*and the matrices $X_j$ satisfy the requirements of Lemma 5 with $u_j(n)$ and $t_j(n)$, respectively (the matrices may be non-square and of different dimensions). Then $C(\zeta) \geq \sum_{j=1}^{m}(u_j(n) + t_j(n))$ and, moreover, $C_{3/4}(\zeta) \geq \sum_{j=1}^{m}(u_j(n) + t_j(n))$.*

[3] Recall that we distinguish outputs from output gates: an output can be the negation of a function computed in an output gate.

*Proof.* We proceed similarly to Lemma 5. Note that when we substitute a variable from $x^{(1)}$, it does not change anything in $X_2$, and vice versa. Thus we substitute "good" variables (those that eliminate two gates) as long as we have them and then substitute "bad" variables (eliminating one gate per step) when we do not have good ones *separately for each matrix*. If one of the matrices runs out of rows that contain at least two nonzero entries (it may happen after eliminating $u_i(n) - 1$ "good" and then $t_i(n) - u_i(n) + 2$ other variables from it), we substitute the remaining variables corresponding to this matrix and forget about this part of the block-diagonal matrix.

It can happen, however, that one of the inputs (variables) in the topmost gate is from $x^{(1)}$ and the other one is from $x^{(2)}$. Both cases from the proof of Lemma 5 go through smoothly in this situation: in the first case we substitute a value in the good variable, and the second case is impossible for the same reasons.

Thus, eliminating all columns from $X_i$ leads to eliminating at least

$$2(u_i - 1) + (t_i - u_i + 2) = t_i + u_i$$

gates, and we obtain the overall bound of

$$C_{3/4}(\zeta) \geq \sum_{j=1}^{m} (u_j + t_j).$$

□

We now formulate the direct consequences of these lemmas and note upper bounds for our specific matrices.

**Lemma 7.** *Let* $n, n' \equiv 0 \pmod 4$,

$$\alpha(\boldsymbol{x}) = A^{-1}\boldsymbol{x}, \quad \alpha_2(\boldsymbol{x}) = \begin{pmatrix} A^{-1} & A^{-2} \end{pmatrix} \boldsymbol{x}, \quad \alpha_*(\boldsymbol{x}) = \begin{pmatrix} A^{-1} & A^{-2} & 0 \\ 0 & 0 & A_*^{-1} \end{pmatrix} \boldsymbol{x},$$

*where* $A_*^{-1}$ *denotes a matrix with the same structure as* $A^{-1}$*, but with dimension* $n'$ *instead of* $n$*. Then* $C_{3/4}(\alpha) \geq \frac{3n}{2} - 2$*,* $C_{3/4}(\alpha_2) \geq \frac{13n}{4} - 5$*,* $C_{3/4}(\alpha_*) \geq \frac{3n'}{2} + \frac{13n}{4} - 7$*.*

*Proof.* Follows from Lemmas 5 and 6, by substituting the respective bounds $u(n)$ and $t(n)$ from Lemma 2 (in particular, $t(n) = n - 2$ for the matrix $A^{-1}$ and $t(n) = 2n - 5$ for $\mathfrak{A}$). □

**Lemma 8.** *1. There exists a circuit of size* $\frac{3n}{2} - 1$ *that implements the linear function* $\phi : \mathbb{B}^n \to \mathbb{B}^n$ *with matrix* $A^{-1}$*.*

*2. There exists a circuit of size* $\frac{7n}{2}$ *that implements the linear function* $\phi : \mathbb{B}^{2n} \to \mathbb{B}^n$ *with matrix* $\begin{pmatrix} A^{-1} & A \end{pmatrix}$*.*

*3. There exists a circuit of size* $\frac{5n}{2} - 1$ *that implements the linear function* $\phi : \mathbb{B}^{2n} \to \mathbb{B}^n$ *with matrix* $\begin{pmatrix} A^{-1} & A^{-1} \end{pmatrix}$*.*

*Proof.*

1. First construct the sum $\bigoplus_{i=1}^{n/2} x_i$ ($\frac{n}{2} - 1$ gates). Then, adding one by one each of the inputs $x_i$, $i = \frac{n}{2}..n$, compute all outputs $y_i$, $i = \frac{n}{2}..n$ and, by the way, the sum of all inputs $\bigoplus_{i=1}^{n} x_i$ (this takes another $\frac{n}{2}$ gates). Finally, the first $\frac{n}{2}$ outputs will be computed by "subtracting" the first $\frac{n}{2}$ inputs from the sum of all inputs one by one (another $\frac{n}{2}$ gates).

2. To implement the left part of this matrix, we need $\frac{3n}{2}-1$ gates. Afterwards we add to each output the two bits from the right part of the matrix (three bits in case of the last row); we add $2n+1$ gates in this way.
3. Note that in this case $\phi(a,b) = \left(\begin{smallmatrix} A^{-1} & A^{-1} \end{smallmatrix}\right)\binom{a}{b} = A^{-1}(a \oplus b)$ for any $a, b \in \mathbb{B}^n$. Thus, we first add $a \oplus b$ ($n$ gates) and then implement $A^{-1}$ ($\frac{3n}{2}-1$ gates). □

## 3 The feebly secure trapdoor function

### 3.1 Two constructions

We are almost ready to present the construction of our feebly trapdoor function (recall Definition 1). In this section, we consider two different constructions, none of which works alone; however, we will merge them into one in the subsequent section, and the resulting mixture will be feebly secure.

In our first construction, the inversion with trapdoor is faster than inversion without trapdoor, but, unfortunately, evaluating the function is even harder. In terms of Definition 1, we now present a feebly trapdoor candidate with identical lengths of the seed, public information, trapdoor, input, and output $c(n) = m(n) = \mathrm{pi}(n) = \mathrm{ti}(n) = n$. Given a random seed, the sampler produces a pair of public and trapdoor information $(\mathrm{pi}, \mathrm{ti})$, where ti is the random seed itself and $\mathrm{pi} = A(\mathrm{ti})$ (thus, the sampler can be implemented using $n+1$ gates). In this construction, evaluation produces the output $c$ for an input $m$ as follows:

$$\mathrm{Eval}_n(\mathrm{pi}, m) = A^{-1}(\mathrm{pi}) \oplus A(m).$$

An upper bound on evaluation circuit complexity follows from Lemma 8; one can evaluate this function with a circuit of size $\frac{7n}{2}$. Inversion with trapdoor goes as follows:

$$\mathrm{Inv}_n(\mathrm{ti}, c) = A^{-1}(A^{-1}(\mathrm{pi}) \oplus c) = A^{-1}(\mathrm{ti} \oplus c).$$

Due to the nice linearity (note that bounds proven in previous sections do not apply here, because the inversion matrix has a lot of identical columns), this circuit can be implemented in $\frac{5n}{2}-1$ gates: first one computes $\mathrm{ti} \oplus c$ using $n$ gates, then one applies $A^{-1}$ using another $\frac{3n}{2}-1$ gates (see Lemma 8).

Finally, an adversary has to invert Bob's message the hard way:

$$m = A^{-1}(A^{-1}(\mathrm{pi}) \oplus c) = \mathfrak{A}\begin{pmatrix} \mathrm{pi} \\ c \end{pmatrix}.$$

By Lemma 7, the complexity of this function is at least $\frac{13n}{4}-5$ gates, and any adversary with less than $\frac{13n}{4}-5$ gates fails on at least $1/4$ of the inputs.

In this construction evaluation is harder than inversion without trapdoor. In order to fix this problem, we use also a different construction, also a candidate trapdoor function now with $c(n) = m(n) = n$ and $\mathrm{pi}(n) = \mathrm{ti}(n) = 0$. Our second construction is just the Hiltgen's feebly one-way function. Thus, the public and trapdoor information is not used at all, and the evaluation–inversion functions are as follows: $\mathrm{Eval}_n(m) = A(m)$, $\mathrm{Inv}_n(c) = A^{-1}(c)$, $\mathrm{Adv}_n(c) = A^{-1}(c)$.

This construction, of course, is not a trapdoor function at all because inversion is implemented with no regard for the trapdoor. For a message $m$ of length $|m| = n$ the evaluation circuit has $n+1$ gates, while inversion, by Lemma 8, can be performed only by circuits of $\frac{3n}{2}-1$ gates each. Thus, in this construction evaluation is easy, while inversion is hard, with or without trapdoor.

### 3.2 The combined construction

We now combine the two functions presented in Section 3.1. The resulting one will make it easier for both inversion with trapdoor and evaluation than for an adversary. We split the input into two parts; the first part $m_1$, of length $n$, will be subject to our first (less trivial) construction, while the second part $m_2$, of length $\alpha n$, will be subject to the second construction. We will choose $\alpha$ later to maximize the relative hardness for an adversary.

Now each participant has a block-diagonal matrix:

$$\begin{aligned}
\mathrm{Eval}_n(\mathrm{pi}, m) &= \begin{pmatrix} A^{-1} & A & 0 \\ 0 & 0 & A_* \end{pmatrix} \begin{pmatrix} \mathrm{pi} \\ m_1 \\ m_2 \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}, \\
\mathrm{Inv}_n(\mathrm{ti}, c) &= \begin{pmatrix} A^{-1} & A^{-1} & 0 \\ 0 & 0 & A_*^{-1} \end{pmatrix} \begin{pmatrix} \mathrm{ti} \\ c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} m_1 \\ m_2 \end{pmatrix}, \\
\mathrm{Adv}_n(\mathrm{pi}, m) &= \begin{pmatrix} A^{-2} & A^{-1} & 0 \\ 0 & 0 & A_*^{-1} \end{pmatrix} \begin{pmatrix} \mathrm{pi} \\ c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} m_1 \\ m_2 \end{pmatrix},
\end{aligned}$$

where $A_*$ denotes the matrix with the same structure as $A$, but with dimension $\alpha n$ instead of $n$. Thus, in terms of Definition 1, we get a feebly trapdoor candidate where inputs and outputs are longer than the seed and the public and trapdoor information: $\mathrm{pi}(n) = \mathrm{ti}(n) = n$, $c(n) = m(n) = (1 + \alpha)n$.

Lemma 8 yields upper bounds for evaluation and inversion with trapdoor, and Lemma 7 yields a lower bound for the adversary: $C(\mathrm{Eval}_n) \leq \frac{7n}{2} + \alpha n + 1$, $C(\mathrm{Inv}_n) \leq \frac{5n}{2} + \frac{3\alpha n}{2} - 2$, $C_{3/4}(\mathrm{Adv}_n) \geq \frac{13n}{4} + \frac{3\alpha n}{2} - 7$. Thus, to get a feebly trapdoor function we simply need to choose $\alpha$ such that $\frac{13}{4} + \frac{3}{2} > \frac{7}{2} + \alpha$ and $\frac{13}{4} + \frac{3}{2} > \frac{5}{2} + \frac{3\alpha}{2}$. The second inequality is trivial, and the first one yields $\alpha > \frac{1}{2}$.

We would like to maximize the order of security of this trapdoor function (Definition 3); since sampling is always strictly faster than evaluation and inversion with trapdoor, we are maximizing

$$\min\left\{ \lim_{n\to\infty} \frac{C_{3/4}(\mathrm{Adv}_n)}{C(\mathrm{Inv}_n)}, \lim_{n\to\infty} \frac{C_{3/4}(\mathrm{Adv}_n)}{C(\mathrm{Eval}_n)} \right\} = \min\left\{ \frac{\frac{13}{4} + \frac{3\alpha}{2}}{\frac{5}{2} + \frac{3\alpha}{2}}, \frac{\frac{13}{4} + \frac{3\alpha}{2}}{\frac{7}{2} + \alpha} \right\}.$$

This expression reaches maximum when $\alpha = 2$, and the order of security in this case reaches $\frac{25}{22}$. We summarize this in the following theorem.

**Theorem 1.** *There exists a feebly trapdoor function with the seed length* $\mathrm{pi}(n) = \mathrm{ti}(n) = n$*, the length of inputs and outputs* $c(n) = m(n) = 3n$*, and the order of security* $\frac{25}{22}$*.*

### 3.3 Hardness amplification

In the previous sections, we saw a construction of a linear feebly trapdoor function that guarantees that any circuit with less than precisely the necessary number of gates fails to invert this function on more that $\frac{3}{4}$ of its inputs. In this section, we use this construction to design a function with a superpolynomial bound on the probability of

success (namely, $2^{-c\sqrt{n}+o(\sqrt{n})}$). Let us denote by $h$ the function an adversary had to compute in the previous section, and by $X$ its matrix. Consider the linear function $H$ defined by the block diagonal matrix

$$H\left(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(m)}\right) = \begin{pmatrix} X & 0 & \ldots & 0 \\ 0 & X & \ldots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \ldots & X \end{pmatrix} \begin{pmatrix} \boldsymbol{x}^{(1)} \\ \boldsymbol{x}^{(2)} \\ \vdots \\ \boldsymbol{x}^{(m)} \end{pmatrix}.$$

By Lemma 6, $H$ has complexity at least $mC(h)$. The dimensions of $X$ are $(1+\alpha)n \times (2+\alpha)n$; we denote $n' = (1+\alpha)n$.

**Lemma 9.** *If a circuit computes $H$ on more that $\frac{1}{p(m)}$ fraction of its inputs, and for each block in $H$:*

- *all columns of $X$ are different;*
- *every row of $X$ has at least $u(n)$ nonzero entries;*
- *after removing any $t(n)$ columns of $X$, this matrix still has at least one row containing at least two nonzero entries,*

*then the complexity of this circuit is not less than $(u(n)+t(n))(m-\log_{4/3} p(m))$.*

*Proof.* First, recall that $H$ consists of $m$ separate blocks with disjoint sets of variables $X_i$; let us denote $h_i = H|_{X_i}$. Since $X_i$ are disjoint, mistakes in computing $h_i$ are independent: if a circuit $C$ computes $h_i$ on $\beta_i$ fraction of its inputs and $h_j$ on $\beta_j$ fraction of its inputs, it cannot compute $H$ on more that $\beta_i\beta_j$ fraction of its inputs. Thus, there are at most $\log_{4/3} p(m)$ blocks where $C$ violating our claim can afford to make mistakes on more than $\frac{1}{4}$ of the inputs. During this proof we will call them "terrible" blocks.

We proceed by the same gate elimination induction we had been using several times already. Consider the topmost gate and the two variables that enter it. In the proof of Lemma 6, we marked variables as "good" or "bad" depending on whether they fall into a block where all "good" variables have been eliminated. This time, we do the same thing, but mark all variables in terrible blocks as "bad" in advance. As in the previous proofs, whenever the topmost gate has at least one "bad" variable as input, we set this variable, thus eliminating only one gate from the circuit. Whenever the topmost gate has two "good" variables as inputs, we should always be able to eliminate two gates from the circuit. There are still the same basic possibilities as in Lemma 5, and we also have to always choose the part of the input assignments space where the circuit errs on less than $\frac{1}{4}$ of the remaining inputs (since the initial circuit errs on less than $\frac{1}{4}$ of these inputs, such a subcurcuit must exist).

The rest of the proof is the same as Lemma 5. We proceed by induction, eliminating two gates whenever two "good" variables enter a topmost gate, and eliminating one "bad" variable whenever it enters a topmost gate. Thus, the overall complexity is at least twice the number of "good" variables plus the number of remaining "bad" variables. We have to discard "terrible" blocks completely — after all, their complexity may actually be equal to zero. Thus, we get a bound of $(t+u)(m-\log_{4/3} p(m))$. □

Note also that stacking the matrices up in a large block diagonal matrix does not change the parameters of a feebly trapdoor function. Thus, we have obtained the following theorem.

**Theorem 2.** *There exists a feebly trapdoor function candidate* $\mathcal{C} = \{\mathrm{Key}_n, \mathrm{Eval}_n, \mathrm{Inv}_n\}$ *with the seed length* $\mathrm{pi}(n) = \mathrm{ti}(n) = n$*, the length of inputs and outputs* $c(n) = m(n) = 3n$ *with complexities* $C(\mathrm{Inv}_n) \le \frac{11n}{2} + O(1)$*,* $C(\mathrm{Eval}_n) \le \frac{11n}{2} + O(1)$*,* $C(\mathrm{Key}_n) = n+1$*, and the order of security* $\frac{25}{22}$*. Moreover, no adversary with less than* $\frac{25}{4}n - \frac{25}{4}\delta n^{(a+1)/2}$ *gates is able to invert this feebly trapdoor function on more than* $(4/3)^{-\delta n^{a/2}+o(\sqrt{n})}$ *fraction of the inputs for any constant* $\delta > 0$*,* $1 > a > 0$*.*

## 4 Conclusion and further work

In this work, we have presented the first known construction of a provably secure trapdoor function, although "security" should be understood in a very restricted sense of Definition 3.

Here are natural directions for further research. First, it would be interesting to devise a more natural construction. Both the second (keyless) construction and the merge of two matrices are counter-intuitive. Second, it would be great to substantially improve the order of security. While a certain improvement to the constant $\frac{25}{22}$ may be straightforward, further development will soon hit the general obstacle of our present inability to prove lower bounds greater than $4n - o(1)$ for $\mathbb{B}^n \to \mathbb{B}^n$ functions. Moreover, the constructions based on linear functions will never overcome a bound of $n-1$ gates per one bit of output; thus, nonlinear constructions are necessary. Finally, a natural extension of this work would be to devise other feebly secure cryptographic primitives.

### Acknowledgements

The authors are grateful to Olga Melanich who noticed a flaw in the initial version of the proof at the very last moment and to the anonymous referees for useful comments.

## References

1. Hiltgen, A.P.: Constructions of feebly-one-way families of permutations. In: Proc. of AsiaCrypt '92. (1992) 422–434
2. Levin, L.A.: One-way functions and pseudorandom generators. Combinatorica **7**(4) (1987) 357–363
3. Goldreich, O.: Foundations of Cryptography. Cambridge University Press (2001)
4. Harnik, D., Kilian, J., Naor, M., Reingold, O., Rosen, A.: On robust combiners for oblivious transfers and other primitives. In: Proc. of EuroCrypt'05, LNCS. Volume 3494. (2005) 96–113
5. Grigoriev, D., Hirsch, E.A., Pervyshev, K.: A complete public-key cryptosystem. Groups, Complexity, Cryptology **1**(1) (2008) 1–12
6. Lamagna, E.A., Savage, J.E.: On the logical complexity of symmetric switching functions in monotone and complete bases. Technical report, Brown University, Rhode Island (jul 1973)
7. Savage, J.E.: The Complexity of Computing. Wiley, New York (1976)
8. Blum, N.: A boolean function requiring $3n$ network size. Theoretical Computer Science **28** (1984) 337–345
9. Wegener, I.: The Complexity of Boolean Functions. B. G. Teubner, and John Wiley & Sons (1987)