Секция 4

МАТЕМАТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

УДК 004.056.53, 004.032.26 DOI 10.17223/2226308X/13/25

НЕЙРОСЕТЕВАЯ ОБФУСКАЦИЯ ВЫЧИСЛЕНИЙ НАД ЗАШИФРОВАННЫМИ ДАННЫМИ

В. Л. Елисеев

Предложен подход по нейросетевой криптографической обфускации вычислений. Опираясь на ранее полученные результаты о свойстве строгой обфускации неразличимости для нейросетевого аппроксиматора, мы предлагаем использовать нейросети для выполнения арифметических и других операций над зашифрованными данными, реализуя таким образом идею применения гомоморфного шифрования для выполнения доверенных вычислений в недоверенной среде. Проводится оценка криптографических свойств предложенного механизма и сопоставление с традиционными подходами к шифрованию на основе секретного ключа. Обсуждаются достоинства и недостатки нейронных сетей применительно к задачам обфускации и обработки зашифрованных данных.

Ключевые слова: *искусственная нейронная сеть, обфускация, гомоморфное шифрование, оценка стойкости.*

Введение

Тенденцией настоящего времени в криптологии является активное исследование новых криптографических систем, обладающих новыми свойствами. Множество работ посвящено изучению так называемых постквантовых алгоритмов, обеспечивающих стойкость к современным и перспективным угрозам. Большое внимание уделяется также функциональному и гомоморфному шифрованию, основной целью которых является выполнение некоторых типов операций над зашифрованными данными без расшифрования. Для обеспечения требуемых свойств задействованы многие математические формализмы, прежде не использовавшиеся в криптографии. Рассмотрим механизм искусственных нейронных сетей для реализации цели, преследуемой гомоморфной криптографией, — выполнению вычислений над зашифрованными данными.

1. Обзор

1.1. Искусственные нейронные сети

Исследования биологических нейронных сетей в 30-е годы XX века дали многие из идей, лёгших в основу кибернетики. Наиболее прямолинейной была попытка получить свойства биологических нейронных сетей с помощью формальных математических моделей, получивших обобщённое название «*искусственные нейронные сети*». Одной из успешных формальных моделей биологических нейронных сетей стал *многослойный перцептрон* (*MultiLayer Perceptron* (*MLP*)). Многослойный перцептрон (далее — нейросеть) вычисляет выходной вектор $x^{(N)}$ по входному $x^{(0)}$:

$$x^{(N)} = \mathcal{N}(x^{(0)}),$$

при этом реализует алгебраическое преобразование вида

$$x_i^{(k)} = s\left(\sum_{j=1}^{m_{k-1}} w_{ij}^{(k)} x_j^{(k-1)}\right),\ k = 1, \ldots, N,\ i = 1, \ldots, m_k,$$

где N — количество слоёв нейросети; $x^{(k)} \in \mathbb{R}^{m_k}$ — вектор выходов k-го слоя; $x^{(0)} \in \in \mathbb{R}^{m_0}$ — вектор входов нейросети; m_k — количество нейронов слоя k; m_0 — количество входов первого слоя нейросети; $w_{ij}^{(k)} \in \mathbb{R}$ — весовой коэффициент j-го входа i-го нейрона слоя k; $s(.)$ — дифференцируемая функция, называемая также функцией активации. В зависимости от вида функции активации различают линейный ($s(t) = t$), ступенчатый ($s(t) = \text{sign}(t)$), сигмоидальный $\left(s(t) = \dfrac{1}{1+e^{-t}}\right)$ нейроны. В глубоких нейросетях используются и другие виды функций активации [1].

Для многослойного персептрона с двумя слоями и сигмоидальной функцией активации в 1987 г. Р. Хехт-Нильсеном доказана представимость любой непрерывной функции $f(\cdot)$ с любой наперёд заданной точностью [2]. Таким образом, можно утверждать, что многослойный перспептрон с сигмоидальной функцией активации подходит для реализации любых сколь угодно сложных функций, в том числе булевых, широко используемых в криптографии.

Для аппроксимации некоторой целевой функции $f(\cdot)$ нейросетью $\mathcal{N}(\cdot)$ применяется процедура, называемая обучением, результатом которой являются такие значения весовых коэффициентов нейронной сети $w_{ij}^{(k)}$, что ошибка аппроксимации $\varepsilon = \| \mathcal{N}(x) - f(x) \|$ оказывается в допустимых пределах. Процедура обучения, как правило, реализуется путём многомерной оптимизации заданной функции ошибки в пространстве весовых коэффициентов на некотором множестве обучающих данных, образованных парами $(x, f(x))$. Оптимизируемой функцией часто выступает среднеквадратическая ошибка, вычисляемая по формуле

$$\overline{\varepsilon^2} = \sqrt{\frac{1}{M} \sum_{i=1}^{M} \| \mathcal{N}(x_i) - f(x_i) \|^2}.$$

Следует отметить, что в большинстве случаев нейросети используются для аппроксимации *неизвестной* функции $f(\cdot)$, про которую известно только то, что она для некоторых аргументов $x \in \mathbb{X}$ принимает значения $y \in \mathbb{Y}$, однако вид функции и её поведение (за исключением непрерывности) между точками (x, y) неизвестны. Аппроксимируя неизвестную функцию по таблично заданному подмножеству её значений, нейросеть позволяет решать многие задачи, связанные с классификацией, распознаванием образов, фильтрацией помех, предсказанием временных рядов и даже моделированием квантовых процессов.

При обучении нейросети на ограниченном множестве данных и при неизвестном виде аппроксимируемой функции приходится решать ряд проблем, связанных с выбором архитектуры нейросети (число слоёв и число нейронов в них) и эффектом переобучения (overfitting) — потерей обобщающей способности нейросетью [3]. Для борьбы с переобучением исходное множество пар (x, y) разделяют на обучающие и проверочные, причём вторые не участвуют в обучении и используются для получения оценки ошибки ε.

Сигмоидальная функция активации $s(t) = (1 + e^{-t})^{-1}$ непрерывна и ограничена областью значений $0 < s(t) < 1$, поэтому выходы нейросети не могут быть интерпретированы как булевы. Поскольку наша задача — представление векторных булевых

функций, то будем приводить выход нейросети к булеву виду с помощью функции $b : \mathbb{R} \to \mathbb{B}$, где $\mathbb{B} = \{0, 1\}$:

$$b(x) = \begin{cases} 0, & x < 0{,}5, \\ 1, & x \geqslant 0{,}5. \end{cases}$$

Таким образом, нейросетевая обфускация векторной булевой функции $f(\cdot)$ должна реализовываться последовательным вычислением нейросетью вектора непрерывных значений $x^{(N)}$ и его преобразованием к вектору булевых значений:

$$x^{(N)} = \mathcal{N}(x^{(0)}), \quad y_i = b(x_i^{(N)}),\ i = 1, \ldots, m_N.$$

1.2. Нейронные сети и криптология

Проведём краткий обзор известных примеров применения искусственных нейросетей в задачах разработки и анализа криптографических примитивов. Известны работы, посвящённые криптоанализу с использованием нейросетей [4, 5], однако нейросеть в этом случае выступает в качестве вспомогательного инструмента и не используется для решения задачи защиты информации. В некоторых работах рассматривается создание на основе нейросетей хэш-функций [6, 7] и генераторов случайных чисел [8].

Делались также попытки построить криптосистемы на основе специфических способностей нейронных сетей к обучению и представлению нелинейных зависимостей, однако такие криптосистемы оказались довольно быстро взломаны [9, 10]. Перечень нейросетевых криптосистем и их взломов приведён в табл. 4 в [11].

В работе [12] решается задача синтеза нейросетевой симметричной криптосистемы по критерию минимизации вероятности дешифрования подслушивателем передаваемого сообщения при фиксированном ключе. В качестве подслушивателя рассматривается ещё одна нейросеть, поэтому предложенный подход получил название Adversarial Neural Cryptography (ANC). Отличие нейросети подслушивателя от нейросетей легитимных контрагентов заключается в отсутствии входов с битами секретного ключа. Предложенный алгоритм обучения позволил синтезировать нейросети для шифрования и расшифрования данных. Однако успешное решение поставленной задачи не обеспечило секретность шифруемых данных в критериях, привычных для традиционной криптографии, поскольку роль подслушивателя выполняет нейросеть, что является очень упрощённой моделью криптоаналитика.

С целью устранения отмеченного недостатка в [11] предложено усилить секретность синтезируемого нейросетевого алгоритма шифрования/расшифрования за счёт возможности подслушивателя на этапе обучения использовать атаку с подобранным открытым текстом (Chosen Plaintext Attack). Вторым усовершенствованием было использование нейросети специальной архитектуры, названной *CryptoNet* и обеспечивающей операцию XOR на уровне базового элемента сети, что теоретически позволяет синтезировать в результате обучения даже одноразовый блокнот (one-time pad). Полученные результаты подтвердили возможность синтеза более секретной системы шифрования на основе нейросетей, однако практическая ценность пока невелика, так как в силу большой вычислительной сложности алгоритма обучения нейросети размер блока шифруемых данных и длина используемого ключа не превышали 16 бит.

Более масштабными и практически полезными выглядят попытки применить нейросети для обработки зашифрованных данных без ограничений на вид операций над ними. В основополагающей работе [13] доказывается возможность использования нейросетей для обработки зашифрованных данных через представление реализуемой нейросетью функции полиномом и дальнейшую реализацию этого полинома с помощью

гомоморфного шифрования (HE). Следует отметить некоторую избыточность приведённых авторами [13] рассуждений, поскольку доказываемый ими факт представимости любых непрерывных функций с помощью нейросетей типа «многослойный перцептрон» давно известен [14]. Тем не менее в [13] проводится важная мысль о подобии возможностей, предоставляемых гомоморфным шифрованием и искусственными нейросетями.

Недавние успехи в развитии и всё более эффективной реализации алгоритмов полного гомоморфного шифрования (FHE) сделали актуальной задачу обработки сложных данных в зашифрованной форме. Одним из инструментов обработки медицинских и финансовых данных являются нейросети, однако такие данные в большинстве случаев не должны раскрываться для третьей стороны — владельца вычислительных ресурсов. По этой причине авторы [15] предлагают компилятор, преобразующий обученную нейросеть в эквивалентную, но работающую над зашифрованными данными. Таким образом, довольно прямолинейный подход авторов [13] по замене нейросетей их гомоморфными полиномиальными аппроксимациями предложен в [15] для практической реализации.

Вопрос о криптостойкости использования нейросетей при операциях над зашифрованными данными в [13, 15] не рассматривается, видимо, из-за надежды на криптостойкость применяемого гомоморфного шифрования. Тем не менее можно утверждать, что криптостойкость в наилучшем случае будет определяться разрядностью используемого ключа гомоморфной криптосистемы.

Резюмируя, можно отметить, что наиболее привлекательными свойствами нейросетей, используемыми в исследованиях, являются обучение на примерах, хаотическая динамика и нелинейное отображение. При этом многие исследователи явно или неявно подразумевают, что реализованные в обученной нейросети преобразования не могут быть извлечены для полезного использования [12].

В [16] доказано, что нейросети обладают свойством обфускатора неразличимости, что позволяет использовать их для сокрытия деталей реализации вычислительных алгоритмов. Основная идея, предлагаемая в настоящей работе, заключается в применении нейронных сетей в качестве стойкого обфускатора алгоритмов и основанного на этом подхода по обработке зашифрованных данных непосредственно нейросетью.

2. Нейросетевая криптографическая обфускация

В качестве удобной абстракции некоторого алгоритма будем рассматривать программу P, реализующую вычисления над булевыми векторами фиксированной длины:

$$\forall x \in X \subseteq \mathbb{B}^n \ (y = P(x),\ y \in Y \subseteq \mathbb{B}^m).$$

Предположим, что программу P необходимо выполнять в недоверенной вычислительной среде, то есть существует риск того, что входные и выходные данные программы, а также, возможно, алгоритм и промежуточные результаты станут доступны злоумышленнику. К подобного рода вычислительным средам можно отнести все без исключения публичные облачные сервисы, доверие к которым основывается исключительно на репутации владельцев, а также технических мерах по разграничению доступа к данным различных пользователей сервиса.

Рассмотрим возможность обработки зашифрованных данных программой P с получением результата также в зашифрованном виде. Для этого определим операции зашифрования E и расшифрования D. Данные функции должны обеспечивать обратимость на полном множестве булевых векторов требуемой длины как для входа, так

и для выхода программы:

$$\forall x \in X \subseteq \mathbb{B}^n \; \exists c_x \in C_x \subseteq \mathbb{B}^n \; (c_x = E_X(x) \;\&\; x = D_X(c_x)),$$
$$\forall y \in Y \subseteq \mathbb{B}^m \; \exists c_y \in C_y \subseteq \mathbb{B}^m \; (c_y = E_Y(y) \;\&\; y = D_Y(c_y)).$$

В таком случае вычисления над зашифрованными данными могли бы быть представлены как

$$c_y = E_Y\Big(P\big(D_X(c_x)\big)\Big),$$

где $_x$ — зашифрованный входной вектор x; c_y — зашифрованный результат y. Операции зашифрования $c_x = E_X(x)$ и расшифрования $y = D_Y(c_y)$ должны выполняться в доверенном окружении.

Эта схема могла бы быть очевидным решением исходной задачи для выполнения программы в недоверенной среде, если бы не два обстоятельства. Во-первых, операции E_Y и D_X являются симметричными криптографическими алгоритмами, что неизбежно потребует хранения рядом с ними секретных ключей, что невозможно сделать доверенным образом в недоверенной среде. Во-вторых, программа P всё равно обрабатывает открытые данные, что предоставляет злоумышленнику достаточно возможностей для её исследования и, например, нелицензионного использования.

Согласно теореме 1 в [16], для любой векторной булевой функции возможно построить нейросетевой обфускатор $\mathcal{N}$, в частности, такой, что

$$\forall x \in X \; \big(c_x = E_X(x) \;\&\; c_y = E_Y(P(x)) \;\&\; \hat{c}_y = \mathcal{N}(c_x)\big) \Rightarrow \big(|c_{yi} - \hat{c}_{yi}| < 0{,}5, \; i = 1, \ldots, m\big).$$

Нейросетевой обфускатор реализует операцию вычисления зашифрованного результата по зашифрованным входным данным. При этом, согласно теореме 2 в [16], он является обфускатором неразличимости, то есть эффективно скрывает способ реализации целевой функции. Таким образом, обученная нейронная сеть $\mathcal{N}$ может выполняться в любом недоверенном окружении, выполняя над передаваемыми ей зашифрованными входными данными операцию, результат которой после расшифрования эквивалентен вычислению исходной программы P над открытыми входными данными.

Зададим последовательность шагов для обучения нейросетевого обфускатора программы, обрабатывающего зашифрованные данные (алгоритм 1).

Для выполнения вычислений над $x \in X$ в недоверенном окружении необходимо выполнить последовательность шагов, представленную алгоритмом 2. При необходимости многих вычислений шаг 1 можно выполнить однократно, повторяя только шаги 2–6.

Алгоритм 1. Обучение нейросетевого криптографического обфускатора

1: Вычислить результат работы программы P для всего множества возможных аргументов $x^{(j)} \in X$, образовав обучающее множество пар $(x^{(j)}, y^{(j)})$: $y^{(j)} = P(x^{(j)})$, $j = 1, \ldots, |X|$.
2: Создать взаимно однозначную таблицу подстановки $E_X : x^{(j)} \to c_x^{(j)}$, где $c_x^{(j)} \in C_x$, $j = 1, \ldots, |X|$.
3: Создать взаимно однозначную таблицу подстановки $E_Y : y^{(j)} \to c_y^{(j)}$, где $c_y^{(j)} \in C_y$, $j = 1, \ldots, |Y|$, $|Y|$ — мощность множества значений P, то есть количество уникальных $y^{(j)}$ в Y, $y^{(j)} = P(x)$, $x \in X$.
4: Выбрать архитектуру $\mathcal{N}$ с n входами, m выходами, не менее чем двумя слоями нейронов с нелинейной функцией активации и достаточным числом весовых коэффициентов.
5: Обучить нейронную сеть $\mathcal{N}$ на множестве пар $(c_x^{(j)}, c_y^{(j)})$, $j = 1, \ldots, |X|$, где $c_x^{(j)} = E_X(x^{(j)})$, $c_y^{(j)} = E_Y\big(P(x^{(j)})\big)$, так, чтобы

$$\forall c_x^{(j)} \in C_x \ \big(\hat{c}_y^{(j)} = \mathcal{N}(c_x^{(j)})\big) \Rightarrow \left|c_{yi}^{(j)} - \hat{c}_{yi}^{(j)}\right| < 0{,}5, \ i = 1, \ldots, m.$$

Алгоритм 2. Применение нейросетевого криптографического обфускатора

1: Передать обученную нейронную сеть $\mathcal{N}$ в недоверенное окружение.
2: Вычислить в доверенном окружении $c_x = E_X(x)$.
3: Передать c_x в недоверенное окружение.
4: Вычислить $c_y = \mathcal{N}(c_x)$ в недоверенном окружении.
5: Передать c_y в доверенное окружение.
6: Вычислить $y = D_Y(c_y)$ в доверенном окружении.

3. Шифрование данных нейросетевой криптографической обфускации

Рассмотрим способы, которые могут быть использованы для шифрования данных, подаваемых на вход нейросетевого криптографического обфускатора и получаемых в качестве результата его работы. Сравним стойкость шифрования с ключом и с помощью таблицы случайных взаимно однозначных подстановок.

Взаимно однозначные таблицы подстановки могут быть созданы с помощью симметричного блочного шифра с секретным ключом K. В таком случае $c_x = E_K(x)$ и $x = D_K(c_x)$. Поскольку обученная нейронная сеть $\mathcal{N}$ реализует обфускатор неразличимости, то информация о ключе шифрования не будет представлена в извлекаемом для злоумышленника виде. Стойкость шифра в этом случае определяется длиной ключа.

Таблицы подстановок могут быть также созданы с помощью генератора случайных чисел, и секретным ключом будет являться сама таблица целиком. Для практического использования в криптографических приложениях данный подход выглядит слишком ресурсоёмким по сравнению с коротким ключом длиной k бит, поскольку ключом выступает таблица размера $n \cdot 2^n$ бит, где n — длина шифруемого вектора. Однако с точки зрения ресурсов, затрачиваемых на обучение нейронной сети, эти подходы не отличаются, поскольку длительность обучения зависит от мощности (размера) обучающей выборки, то есть от размера таблицы, задающей число пар подстановок: $E_X : x^{(j)} \to c_x^{(j)}$, $j = 1, \ldots, |X|$.

Сравним стойкость шифрования булева вектора $x \in \mathbb{B}^n$ таблицей случайных подстановок со стойкостью шифрования с помощью ключа длиной k бит. Стойкость идеального блочного шифра определяется перебором всех возможных ключей, что для ключа длиной k бит составляет 2^k.

Для таблицы случайных подстановок количество вариантов ключа, то есть различных таблиц, составляет $(2^n)!$, то есть переборная стойкость определяется размерностью шифруемого вектора.

Наиболее распространённые современные блочные шифры имеют длину ключа $k = 256$ бит. Найдём, при каком размере таблицы случайных подстановок n достигается сравнимая переборная стойкость. Для этого решим неравенство $(2^n)! > 2^k$ для $k = 256$:

$$\log_2\bigl((2^n)!\bigr) > 256,$$

что даёт $117{,}6 < 256$ при $n = 5$ и $295{,}9 > 256$ при $n = 6$. Таким образом, при размере вектора $n \geqslant 6$ переборная стойкость шифра таблицы случайных подстановок превышает переборную стойкость современных шифров AES-256 и ГОСТ 34.12-2018 «Кузнечик» и «Магма».

Конечно, не все возможные таблицы случайных подстановок обеспечивают секретность шифрования, кроме того, некоторые из них реализуют тривиальные преобразования вроде шифра Цезаря и других, для которых найдены простые алгоритмы взлома. Следует также отметить, что для малых n можно успешно применять частотный анализ. Именно поэтому размер блока современных шифров составляет 128 бит.

Частотный анализ применим в предположении, что известно частотное распределение открытых данных. В случае нейросетевой обфускации прикладной программы P смысл зашифрованных данных для злоумышленника неизвестен, поскольку скрыта реализация алгоритма. Если априорно нет предположений о семантике обрабатываемых данных и назначении программы P, то ценность частотного анализа представляется незначительной.

Дополнительным усложнением задачи взлома для злоумышленника является возможность использования различных таблиц подстановок E_X и E_Y для входных и выходных данных. Таким образом, даже в случае тождественной программы $P : y = x$ шифрующие подстановки сделают взлом нетривиальным:

$$c_y = E_Y(D_X(c_x)).$$

К числу приёмов, позволяющих усложнить взлом, можно отнести увеличение размерности n входного вектора данных больше минимально необходимого для представления всего множества входных векторов X. Трудоёмкость процедуры обучения $\mathcal{N}$ при этом не вырастет, так как объём обучающей выборки останется таким же. Для того чтобы скрыть частотный состав шифруемых данных, можно с некоторой периодичностью подавать на вход нейросети случайные векторы c_x, в том числе не являющиеся результатом шифрования каких-либо допустимых $x \in X$. Выход нейросети $\mathcal{N}$ в этом случае не определён, и его тоже можно считать случайным.

Таким образом, в качестве подхода шифрования входных и выходных данных для нейросетевой криптографической обфускации целесообразно использовать таблицы случайных подстановок.

4. Обсуждение

Предложенный подход к реализации нейросетевого шифрования, совмещённого с обфускацией, обеспечивает реализацию функций, традиционно возлагаемых на го-

моморфное шифрование. В то же время нельзя сказать, что нейросетевая криптографическая обфускация может быть напрямую сопоставлена с гомоморфным шифрованием. Во-первых, гомоморфное шифрование основано на использовании ключа и его стойкость требует изучения, подобно всем криптографическим алгоритмам. В предложенном подходе алгоритм шифрования данных может быть произвольным, включая таблицы случайных подстановок, являющиеся идеальным блочным шифром [17].

Во-вторых, в гомоморфном шифровании алгоритм обработки зашифрованных данных не является секретом. Предложенный подход неделимо совмещает шифрование и обфускацию, защищая конфиденциальность как данных, так и алгоритма, который их обрабатывает.

Нейросетевая реализация криптографических алгоритмов уже находилась в фокусе внимания исследователей. Попытки синтеза криптографических систем на основе специфических свойств нейросетей, будь то хаотическая динамика [9, 10] или конкурентное обучение [11, 12], преследовали целью создание каких-то новых шифрующих алгоритмов. В противоположность этому, нейросетевая криптографическая обфускация не вводит нового алгоритма шифрования, предлагая использование существующих с ключом или на основе таблицы случайных подстановок.

Предлагаемый подход представляется фундаментальным и комплексным решением задачи защиты интеллектуальной собственности и конфиденциальных данных, частные подходы к которым с использованием искусственных нейронных сетей описаны ранее [13, 15].

Выводы

В работе предложен подход по нейросетевой криптографической обфускации алгоритмов, обрабатывающих зашифрованные данные. Настоящий механизм в целом преследует цели гомоморфного шифрования, но не эквивалентен ему. Представляется, что на основе этого подхода возможно реализовать доверенные вычисления в недоверенном вычислительном окружении, к которым относятся публичные современные облачные среды.

ЛИТЕРАТУРА

1. *Николенко С., Кадурин А., Архангельская Е.* Глубокое обучение. Погружение в мир нейронных сетей. СПб.: Питер, 2020.
2. *Алексеев Д. В.* Приближение функций нескольких переменных нейронными сетями // Фундаментальная и прикладная математика. 2009. Т. 15. № 3. С. 9–21.
3. *Хайкин С.* Нейронные сети: полный курс. 2-е изд. М.: Вильямс, 2008.
4. *Focardi R. and Luccio F. L.* Neural cryptanalysis of classical ciphers // Proc. ICTCS. 2018. http://ceur-ws.org/Vol-2243/paper10.pdf
5. *Danziger M. and Henriques M. A. A.* Improved cryptanalysis combining differential and artificial neural network schemes // Intern. Telecommun. Symp. (ITS). Sao Paulo, 2014. P. 1–5.
6. *Turcanik M.* Using recurrent neural network for hash function generation // Intern. Conf. Appl. Electronics (AE). Pilsen, 2017. P. 1–4.
7. *Lian S., Sun J., and Wang Z.* One-way hash function based on neural network // arXiv:0707.4032. 2007.
8. *Karras D. A. and Zorkadis V.* On neural network techniques in the secure management of communication systems through improving and quality assessing pseudorandom stream generators // Neural Networks. 2003. V. 16. Iss. 5–6. P. 899–905.

9. *Kanter I., Kinzel W., and Kanter E.* Secure exchange of information by synchronization of neural networks // Europhys. Lett. 2002. No. 57. P. 141–147.
10. *Klimov A., Mityagin A., and Shamir A.* Analysis of neural cryptography // LNCS. 2002. V. 2502. P. 288–298.
11. *Coutinho M., De Oliveira A. R., Borges F., et al.* Learning perfectly secure cryptography to protect communications with adversarial neural cryptography // Sensors. 2018. No. 18. Article 1306. https://pubmed.ncbi.nlm.nih.gov/29695066/
12. *Abadi M. and Andersen D. G.* Learning to protect communications with adversarial neural cryptography // arXiv:1610.06918. 2016.
13. *Xie P., Bilenko M., Finley T., et al.* Crypto-nets: Neural networks over encrypted data // arXiv:1412.6181. 2014.
14. *Hecht-Nielsen R.* Kolmogorov's mapping neural network existence theorem // IEEE First Annual Int. Conf. on Neural Networks, San Diego, 1987. V. 3. P. 11–13.
15. *Dathathri R., Saarikivi O., Chen H., et al.* CHET: an optimizing compiler for fully-homomorphic neural-network inferencing // Proc. PLDI 2019. N.Y.: ACM, 2019. P. 142–156.
16. *Елисеев В. Л.* Искусственные нейронные сети как механизм обфускации вычислений // Прикладная дискретная математика. Приложение. 2019. № 12. С. 165–169.
17. *Фергюсон Н., Шнайер Б.* Практическая криптография. М.: Диалектика, 2005.

УДК 004.75 DOI 10.17223/2226308X/13/26

МЕТОД СОКРЫТИЯ ПРИВАТНЫХ ДАННЫХ ДЛЯ БЛОКЧЕЙН-СИСТЕМЫ ПРОВЕДЕНИЯ ТЕНДЕРОВ[1]

Д. О. Кондырев

Предложен новый метод, позволяющий решить проблему приватности информации в открытых блокчейн-системах с использованием криптографического протокола доказательства с нулевым разглашением zk-SNARK. Метод реализован в виде криптографической схемы на основе библиотеки libsnark и интегрирован в модифицированный Ethereum C++ клиент.

Ключевые слова: *тендеры, распределённые системы, блокчейн, доказательство с нулевым разглашением, zk-SNARK, платформа Ethereum.*

На сегодняшний день большинство конкурсных закупок и электронных торгов проводятся через специализированные информационные системы. В таких системах участники должны быть уверены в том, что никто не имеет возможности нарушить правила проведения тендера или получить доступ к конфиденциальной информации. Решить проблему доверия при проведении тендеров позволяет блокчейн. Однако при использовании этой технологии все данные сохраняются в открытом виде и доступны всем участникам. В случае с тендерами открытость информации нарушает тайну заявок, которая должна быть сохранена до окончания этапа запроса предложений.

Ранее была разработана блокчейн-система для проведения тендеров с шифрованием заявок [1]. Однако такой подход не позволяет проверить корректность зашифрованной заявки в момент её подачи. Ещё одним недостатком является то, что все участники могут наблюдать факт подачи заявки пользователем.

[1]Работа выполнена при поддержке Математического центра в Академгородке (г. Новосибирск), соглашение с Министерством науки и высшего образования Российской Федерации № 075-15-2019-1613, и Лаборатории криптографии JetBrains Research.