Сергей В. Запечников
Национальный исследовательский ядерный университет «МИФИ»,
Каширское ш., 31, Москва, 115409, Россия
e-mail: SVZapechnikov@mephi.ru, http://orcid.org/0000-0002-7975-6040

КОНФИДЕНЦИАЛЬНОЕ МАШИННОЕ ОБУЧЕНИЕ НА ОСНОВЕ ЧЕТЫРЕХСТОРОННИХ ПРОТОКОЛОВ БЕЗОПАСНЫХ ВЫЧИСЛЕНИЙ*

DOI: http://dx.doi.org/10.26583/bit.2022.2.04

Аннотация. Статья посвящена анализу систем конфиденциального машинного обучения, основанных на концепции безопасных четырехсторонних вычислений. Анализируются преимущества четырехсторонней модели вычислений над двух- и трёхсторонними, приводятся определения, выражающие стойкость протоколов безопасных многосторонних вычислений к воздействию активного противника. В качестве примера системы безопасных четырехсторонних вычислений с развитой функциональностью, наиболее полно реализующей свойства стойкости к активному противнику, рассматривается система Tetrad. Анализируется реализованная в Tetrad концепция вычислений, основанная на идее смешанного использования арифметических, булевых и искаженных схем при вычислении функций. Рассматривается многоуровневая архитектура системы, подробно анализируются протоколы, относящиеся к различным уровням архитектуры. Проводится анализ различных форм разделения секрета. Систематизация протоколов завершается анализом высокоуровневых протоколов, используемых для реализации процедур обучения и применения моделей в конфиденциальном исполнении. Рассматриваемый набор протоколов позволяет выполнять стандартные для систем машинного обучения высокоуровневые операции, но с разделенными секретами. Приведены выводы о достоинствах, недостатках, особенностях и ограничениях четырехсторонних протоколов безопасных вычислений для решения задач конфиденциального машинного обучения.

Ключевые слова: конфиденциальное машинное обучение, безопасные многосторонние вычисления, схемы разделения секрета.

Для цитирования: ЗАПЕЧНИКОВ, Сергей В. КОНФИДЕНЦИАЛЬНОЕ МАШИННОЕ ОБУЧЕНИЕ НА ОСНОВЕ ЧЕТЫРЕХСТОРОННИХ ПРОТОКОЛОВ БЕЗОПАСНЫХ ВЫЧИСЛЕНИЙ. Безопасность информационных технологий, [S.l.], т. 29, № 2, с. 46–56, 2022. ISSN 2074-7136. URL: https://bit.spels.ru/index.php/bit/article/view/1417. DOI: http://dx.doi.org/10.26583/bit.2022.2.04.

***Благодарности.** Работа выполнена при поддержке Министерства науки и высшего образования РФ (проект государственного задания № 0723-2020-0036).*

Sergey V. Zapechnikov
National Research Nuclear University MEPhI (Moscow Engineering Physics Institute),
Kashirskoe sh., 31, Moscow, 115409, Russia
e-mail: SVZapechnikov@mephi.ru, http://orcid.org/0000-0002-7975-6040

Privacy-preserving machine learning based on secure four-party computations*

DOI: http://dx.doi.org/10.26583/bit.2022.2.04

Abstract. The paper is devoted to the analysis of privacy-preserving machine learning systems based on the concept of secure four-party computations. The advantages of the four-party computations over the two- and three-party computations are analyzed. The definitions are given that express the resistance of secure multi-party computation protocols against an active adversary. The Tetrad system is considered as an example of a secure four-party computations system with advanced functionality that most fully implements the properties of security against adversary actions. The concept of computations implemented in Tetrad based on the idea of mixed use of arithmetic as well as Boolean and garbled circuits in the function computations is analyzed. The multi-level architecture of the system is considered. The protocols related to various levels of architecture are analyzed in detail. For this purpose, an analysis

of various forms of secret sharing is carried out. After that the basic protocols with shared secrets are considered. The systematization of protocols is completed with the analysis of high-level protocols used for the private implementation of model learning and inference procedures. This set of protocols allows performing high-level operations that are standard for machine learning systems, but with shared secrets. The paper concludes considering the advantages, disadvantages, features and limitations of four-party secure computations protocols for privacy-preserving machine learning.
Keywords: privacy-preserving machine learning, secure multi-party computations, secret sharing scheme.
For citation: ZAPECHNIKOV, Sergey V. Privacy-preserving machine learning based on secure four-party computations. IT Security (Russia), [S.l.], v. 29, no. 2, p. 46–56, 2022. ISSN 2074-7136. URL: https://bit.spels.ru/index.php/bit/article/view/1417. DOI: http://dx.doi.org/10.26583/bit.2022.2.04.

Acknowledgement. *This work was supported by the Ministry of Science and Higher Education of the Russian Federation (state assignment project No. 0723-2020-0036).*

Введение

Безопасные многосторонние вычисления (БМВ) – одно из важнейших направлений развития современной криптографии. Задача БМВ заключается в обеспечении возможности совместного вычисления несколькими участниками функции произвольного вида при условии, что каждый из участников вычислений знает один из аргументов этой функции, и этот аргумент является его секретом, который не должен быть разглашён в процессе вычислений. В то же время результат вычислений должен стать известен всем участникам.

Одна из главных областей применения БМВ – конфиденциальное машинное обучение (КМО). Задача КМО представляет большой практический интерес, поскольку имеет значительные перспективы практического применения в областях, связанных с обработкой персональных данных, а также данных, составляющих коммерческую, врачебную, банковскую и иные охраняемые законом виды тайны.

Системы КМО активно развиваются. Предложены несколько концепций систем КМО, основанных на разных криптографических инструментах: гомоморфном шифровании, схемах разделения секрета, а также на их комбинациях. В рамках настоящего исследования анализируются системы КМО, в которых в качестве главного криптографического примитива используются схемы разделения секрета, при этом разделение секрета может осуществляться между разным количеством участников: от двух и более.

Предлагаемая вниманию читателя статья является финальной частью этого исследования. Две предыдущие статьи [1, 2] были посвящены КМО на основе двусторонних и трехсторонних протоколов безопасных вычислений.

1. Свойства безопасных многосторонних вычислений

Основной мотивацией увеличения числа участников в протоколах БМВ служит желание достичь более высокой производительности системы и по возможности обеспечить для пользователей более сильные гарантии защищенности их данных.

Четырехсторонние вычисления дают возможность реализовать процедуры распределенных вычислений, стойкие к более сильным моделям противника, чем двусторонние и трехсторонние. Преимущественный интерес представляют протоколы, обеспечивающие стойкость к активному противнику. Стойкость в условиях воздействия активного противника обеспечивается двумя ключевыми свойствами протокола, известными как честность и робастность [3]. Дадим их неформальные определения.

Честность, или *равноправность* (fairness) – это гарантии того, что в результате выполнения протокола либо все участники получают одинаковые выходные данные, либо никто из участников не получает никакого результата, т.е. протокол будет прерван. Есть ослабленное определение честности, которое разрешает противнику довести протокол до конца при прерывании участия в нем честных участников, но эти действия не позволят противнику получить какой-либо дополнительной информации, т.е. фактически он может сгенерировать любой результат по своему усмотрению.

Робастность (robustness) – это гарантии того, что честные участники доведут протокол до конца и получат корректный результат в условиях участия в протоколе активного противника, даже если он полностью прервет свое участие в протоколе.

В качестве примера системы безопасных четырехсторонних вычислений с развитой функциональностью, наиболее полно реализующей свойства стойкости к активному противнику, рассмотрим систему, известную под названием Tetrad.

2. Концепция системы Tetrad и её свойства

Tetrad – система КМО, основанная на четырехсторонних протоколах безопасных вычислений, стойкая при условии, что не более чем один из участников вычислений является активным противником [4]. При этом предположении она обеспечивает свойства честности и опционально – свойство робастности.

Система Tetrad основана на идее смешанного использования арифметических, булевых и искаженных схем при вычислении функций. Арифметические схемы вычисляются над кольцом $\mathbb{Z}_{2^l}$, булевы – над кольцом $\mathbb{Z}_2$. При работе системы предполагается однократное выполнение фазы генерации ключей и предварительных вычислений, которые зависят от тех функций, которые в дальнейшем требуется вычислять. Это открывает возможности для быстрого выполнения интерактивной фазы протокола.

Особенности четырехсторонних вычислений, обеспечивающих стойкость к активному противнику, требуют множественных перекрестных вызовов низкоуровневых протоколов, что делает невозможным предоставление интерфейса к низкоуровневым протоколам по отдельности. Вместо этого реализуются протоколы вычисления функций, специфичных для КМО, а с их использованием уже становится возможным реализовать процедуры конфиденциального обучения и применения моделей. Однако, как отмечают авторы [4], пока ассортимент таких протоколов ограничен, что ограничивает и возможности обеспечения конфиденциальности для тех или иных моделей машинного обучения (в первую очередь разработчики систем КМО стараются предоставить инструментарий для КМО сверточных нейросетей).

Архитектура системы Tetrad показана на рис. 1.

Система Tetrad содержит в себе следующие новые научно-технические решения.

1. *Улучшенные* по сравнению с ранее известными системами, такими как Trident [5], *четырехсторонние протоколы безопасных вычислений* арифметических и булевых схем. Базовая версия Tetrad позволяет обеспечить при вычислении схем свойство честности, а небольшие дополнения в низкоуровневые протоколы пересылки сообщений позволяют добиться свойства робастности без заметного увеличения вычислительной и коммуникационной сложности протокола (чем больше количество вентилей в вычисляемой схеме, тем меньше становятся накладные расходы в пересчете на один вентиль схемы).

Ключевая идея арифметических и булевых вычислений в Tetrad состоит в разделении их на две фазы: фазу предварительных вычислений и онлайн-фазу (интерактивную). Однако при необходимости, в частности, при вычислениях по запросу

(on-demand applications) можно отказаться от предварительной фазы и все вычисления выполнять в интерактивной фазе.

2. *Смешанная модель вычислений*, включающая вычисление как арифметических / булевых, так и искаженных схем. Предполагается, что значительная часть вычислений, как правило, выполняется в арифметической / булевой формах, и только некоторые вычисления, в основном, нелинейных функций, для которых получаются очень сложные арифметические / булевы схемы, выполняются в форме искаженных схем. Перед вычислением искаженных схем необходимо преобразовать все ее входные переменные из арифметического / булева разделения секрета в Яо-разделение, а после вычислений – обратно. С целью сокращения накладных расходов на такие преобразования они встроены в протоколы вычисления искаженных схем, обеспечивая сквозные (end-to-end) преобразования данных: арифметическая / булева форма – Яо-форма – вычисление искаженной схемы – арифметическая / булева форма (во всех возможных сочетаниях).

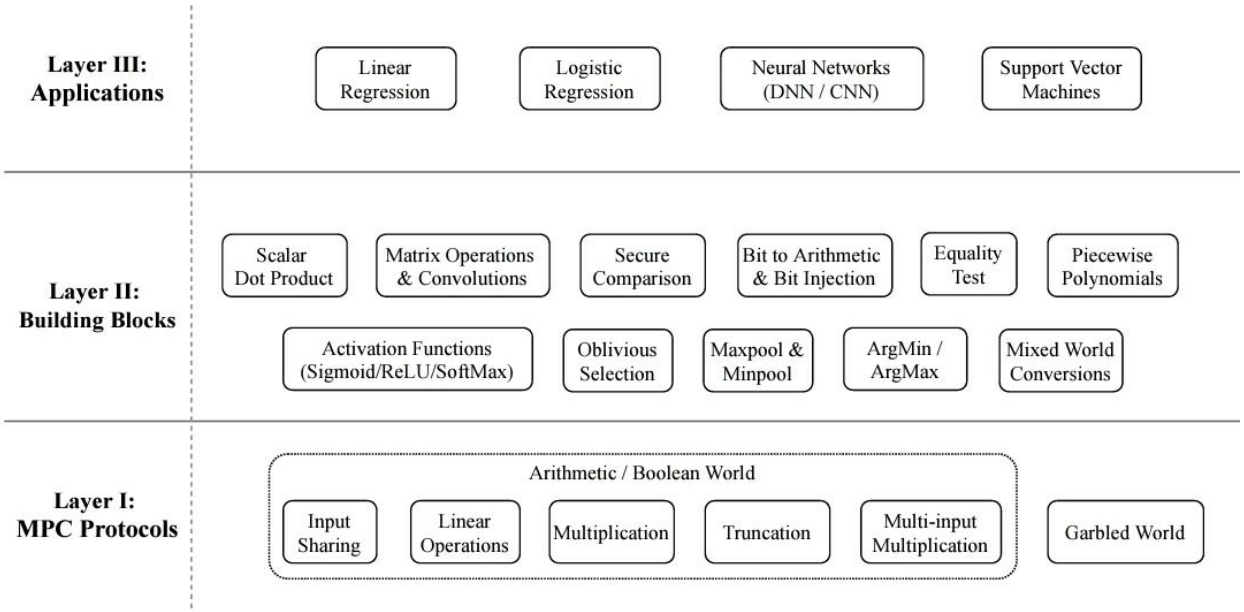

Рис.1. Архитектура системы Tetrad
Fig. 1. The architecture of Tetrad system

Разработчики системы Tetrad предусмотрели два варианта оптимизации протоколов: для минимизации времени вычислений и для минимизации стоимости вычислений.

3. Формы разделения секрета

Рассмотрим основные элементы алгоритмического наполнения системы Tetrad. Пусть $P = \{P_0, P_1, P_2, P_3\}$ – участники системы – серверы, соединенные между собой двусторонними защищенными каналами связи (например, TLS), из которых не более одного участника может быть статическим активным противником. На этапе обучения модели (training) владельцы данных подвергают данные своей обучающей выборки разделению при помощи схемы разделения секрета (СРС) и отсылают их серверам. Обученная модель может быть восстановлена владельцами данных после сбора результатов вычислений от серверов и применения процедур восстановления секретов. На этапе применения модели (inference) владелец модели применяет СРС, чтобы распределить параметры модели между серверами, а после завершения вычислений

применяет процедуры восстановления секрета для получения в открытом виде прогнозного ответа на свой запрос.

Система Tetrad поддерживает вычисления над l-битными числами с фиксированной запятой, представляя их как элементы кольца $\mathbb{Z}_{2^l}$, причем l=64, x=13 – это количество младших битов числа, отведенных для записи дробной части, в остальных l-x-1 битах записана целая часть числа.

Используется следующая семантика разделения секрета (рис. 2):

- $[\cdot]$-разделение – арифметическое разделение переменной между двумя участниками;
- $(\!(\cdot)\!)$-разделение – арифметическое разделение переменной между тремя участниками;
- $\langle\cdot\rangle$-разделение – избыточное разделение секрета между тремя участниками;
- $[\![\cdot]\!]$-разделение – избыточное разделение секрета между четырьмя участниками.

Type	P_0	P_1	P_2	P_3
$[\cdot]$-sharing^a	–	v^1	v^2	–
$(\!(\cdot)\!)$-sharing	–	v^1	v^2	v^3
$\langle\cdot\rangle$-sharing	–	$(\mathsf{v}^1, \mathsf{v}^3)$	$(\mathsf{v}^2, \mathsf{v}^3)$	$(\mathsf{v}^1, \mathsf{v}^2)$
$[\![\cdot]\!]$-sharing^b	$(\lambda_\mathsf{v}^1, \lambda_\mathsf{v}^2, \lambda_\mathsf{v}^3)$	$(\mathsf{m}_\mathsf{v}, \lambda_\mathsf{v}^1, \lambda_\mathsf{v}^3)$	$(\mathsf{m}_\mathsf{v}, \lambda_\mathsf{v}^2, \lambda_\mathsf{v}^3)$	$(\mathsf{m}_\mathsf{v}, \lambda_\mathsf{v}^1, \lambda_\mathsf{v}^2)$

$^a\mathsf{v} = \mathsf{v}_1 + \mathsf{v}_2\ (+\mathsf{v}_3)$ $\quad$ $^b\mathsf{m}_\mathsf{v} = \mathsf{v} + \lambda_\mathsf{v}$

Рис. 2. Семантика разделения секрета в системе Tetrad
Fig. 2. The semantics of secret sharing in Tetrad system

Роли участников в протоколах не одинаковы, подмножество $\mathcal{E} = \{P_1, P_2\}$ называется подмножеством вычислителей и активно участвует во всей интерактивной фазе вычислений. $D = \{P_0, P_3\}$ называется подмножеством помощников: оно помогает вычислителям в верификации результатов, поэтому активно преимущественно в конце протокола. Асимметрия ролей позволяет минимизировать нагрузку на участников в интерактивной фазе протокола, таким образом очень существенно снижая суммарную вычислительную сложность протокола.

Все используемые в Tetrad СРС линейны в том смысле, что по заданным долям переменных $v_1, \dots, v_m$ и общеизвестным константам $c_1, \dots, c_m$ доли линейной комбинации переменных $\sum_{i=1}^{m} c_i v_i$ могут быть вычислены локально для $\forall m$.

Также используются вспомогательные обозначения:

- для $[\![\cdot]\!]$-долей n переменных $a_1, \dots, a_n$ обозначают $\gamma_{a_1 \cdots a_n} = \prod_{i=1}^{n} \lambda_{a_i}$;
- булево и Яо-разделения секрета обозначаются $[\![\cdot]\!]^B$ и $[\![\cdot]\!]^G$ соответственно.

Семантика булева разделения полностью аналогична арифметическому разделению, за исключением того, что операции выполняются в кольце $\mathbb{Z}_2$, а операция сложения заменяется на XOR.

4. Базовые протоколы с разделенными секретами

Протокол Joint-Send (jsnd) позволяет двум участникам P_i, P_j ретранслировать сообщение третьему участнику P_k, гарантируя либо доставку сообщения, либо прерывание протокола в случае несовместности данных. Для этого P_i отсылает участнику P_k сообщение v, а P_j отсылает участнику P_k хэш-код того же сообщения $H(v)$. Участник

P_k принимает сообщение, если вычисленный им хэш-код сообщения, присланного P_i, совпадает с хэш-кодом, присланным P_j. В противном случае он прерывает протокол.

Протокол Joint-Send (jsnd) *для робастных протоколов* позволяет двум участникам P_i, P_j ретранслировать общее для них сообщение $v \in \mathbb{Z}_{2^l}$ третьему участнику P_k, либо гарантируя доставку сообщения v, либо идентифицируя конфликтующую пару участников, один из которых гарантированно является противником. Это означает, что оставшиеся два участника – честные, и одному из них поручается довести вычисления до конца, принимая на себя роль доверенной третьей стороны P_{TP}. Протокол jsnd состоит из двух фаз (send, verify), где фаза send состоит в пересылке сообщения v от P_i к P_k., а оставшаяся часть протокола – фаза verify, которая гарантирует либо корректное выполнение фазы send, либо назначение P_{TP}.

Во всех остальных протоколах используется одна из двух версий протокола jsnd в зависимости от того, какое свойство – честность или робастность – необходимо обеспечить для протокола. Протокол обозначается $\Pi_{jsnd}(P_i, P_j, v, P_k)$.

Протокол разделения секрета Π_{Sh} позволяет участнику P_i сгенерировать $[\![\cdot]\!]$-доли переменной v. Во время фазы предварительных вычислений неинтерактивно вырабатываются случайные доли λ таким образом, что P_i получает полную маску λ_v. Во время интерактивной фазы P_i вычисляет $m_v = v + \lambda_v$ и отсылает его участникам P_1, P_2, P_3, которые обмениваются между собой хэш-кодами для проверки совместности. Если от протокола требуется свойство честности, то в случае несовместности участники прерывают протокол. Если требуется свойство робастности, то участники заканчивают протокол с принятием значения по умолчанию (default value).

Протокол совместного разделения секрета Π_{JSh} позволяет участникам P_i, P_j сгенерировать $[\![\cdot]\!]$-доли переменной v. Протокол подобен протоколу Π_{Sh} за исключением того, что P_j гарантирует корректность разделения, выполненного P_i. Во время предварительных вычислений доли λ выбираются таким образом, что оба участника P_i, P_j получают полную маску λ_v. Во время интерактивной фазы P_i, P_j вычисляют и при помощи протокола jsnd отсылают $m_v = v + \lambda_v$ участникам P_1, P_2, P_3. Если переменной v обладает участник P_0 совместно с одним из оставшихся участников протокола, то протокол может быть оптимизирован:

1) для пары (P_0, P_1): подмножество участников $P \backslash \{P_2\}$ выбирает $\lambda_v^1 \in_R \mathbb{Z}_{2^l}$, полагает $\lambda_v^2 = m_v = 0$, участники P_0, P_1 при помощи протокола jsnd отсылают $\lambda_v^3 = -v - \lambda_v^1$ участнику P_2;

2) для пары (P_0, P_2): подмножество участников $P \backslash \{P_3\}$ выбирает $\lambda_v^3 \in_R \mathbb{Z}_{2^l}$, полагает $\lambda_v^1 = m_v = 0$, участники P_0, P_2 при помощи протокола jsnd отсылают $\lambda_v^2 = -v - \lambda_v^3$ участнику P_3;

3) для пары (P_0, P_3): подмножество участников $P \backslash \{P_1\}$ выбирает $\lambda_v^2 \in_R \mathbb{Z}_{2^l}$, полагает $\lambda_v^3 = m_v = 0$, участники P_0, P_3 при помощи протокола jsnd отсылают $\lambda_v^1 = -v - \lambda_v^2$ участнику P_2.

Протокол восстановления секрета $\Pi_{Rec}(P, v)$ позволяет участникам из множества P вычислить переменную v по заданному ее $[\![\cdot]\!]$-разделению. Каждому из участников протокола не хватает одной доли, чтобы восстановить v, но оставшиеся три участника имеют эту долю. Два из трех участников выполняют протокол jsnd, чтобы переслать недостающую часть секрета.

Процедура разделения аддитивных долей нуля позволяет участникам P_0, P_i получить $Z_i, i = \overline{1,3}$, такие что $Z_1 + Z_2 + Z_3 = 0$. При использовании предварительного распределения

ключей эта процедура неинтерактивна: участники $P\backslash\{P_j\}$ выбирают случайные переменные $r_j, j = \overline{1,3}$, а доли определяются как $Z_1 = r_3 - r_2, Z_2 = r_1 - r_3, Z_3 = r_2 - r_1$.

Протокол умножения разделенных чисел $\langle a\rangle, \langle b\rangle$, *выполняемого в открытом виде участником* P_0, Π_{MulR}. Чтобы выполнить умножение $\langle a\rangle, \langle b\rangle$, где $a, b \in \mathbb{Z}_{2^l}$, в открытом виде участником P_0 и сгенерировать $\langle z\rangle$, такое что $z = ab$, участники локально генерируют $((\cdot))$-разделение переменной z, где P_0 знает все три $((\cdot))$-доли. Чтобы завершить генерацию $\langle z\rangle$, участники $P_0, P_i, i = \overline{1,3}$ рандомизуют свои $((\cdot))$-доли z, используя $((\cdot))$-разделение нуля и выполняя протокол $\mathrm{jsnd}((z))^i$.

Основной протокол умножения чисел, разделенных на доли Π_{Mult}. По заданным долям чисел a, b требуется сгенерировать доли их произведения $z = ab$. Протокол сконструирован таким образом, что участники P_1, P_2 получают замаскированную версию величины z, например, $z - r$, в интерактивной фазе, а участники P_0, P_3 получают маску r на фазе предварительных вычислений. Затем участники генерируют $[\![\cdot]\!]$-разделение этих переменных, выполняя протокол Π_{JSh}, и локально вычисляют $[\![z - r]\!] + [\![r]\!]$, чтобы получить ответ.

Система Tetrad реализует полный комплект протоколов для преобразования разделенных секретов между арифметической, булевой и Яо-формами представления.

5. Высокоуровневые функции для систем КМО

Для реализации КМО в системе Tetrad реализован набор протоколов, выполняющих стандартные для систем машинного обучения высокоуровневые операции с разделенными секретами. Рассмотрим примеры таких протоколов.

Протокол скалярного умножения векторов Π_{dotp} по заданным $[\![\vec{a}]\!], [\![\vec{b}]\!]$, где $|\vec{a}| = |\vec{b}| = d$, вычисляет $[\![z]\!]$, где $z = \vec{a} \odot \vec{b}$. Частичные произведения, полученные от d протоколов умножения компонент векторов, комбинируются и пересылаются в одном сообщении, что делает коммуникационную сложность протокола независимой от длины векторов.

Протокол умножения матриц реализуется как расширение протокола скалярного умножения. Для $\boldsymbol{X}^{u\times v}$ обозначают $m_{\boldsymbol{X}} = \boldsymbol{X} \oplus [\lambda_{\boldsymbol{X}}^1] \oplus [\lambda_{\boldsymbol{X}}^2] \oplus [\lambda_{\boldsymbol{X}}^3]$, где $m_{\boldsymbol{X}}$, $[\lambda_{\boldsymbol{X}}^1]$, $[\lambda_{\boldsymbol{X}}^2]$, $[\lambda_{\boldsymbol{X}}^3]$ – матрицы размерности $u \times v$, $\oplus$ – операция сложения матриц. Произведение матриц $\boldsymbol{X}^{u\times v}\boldsymbol{Y}^{v\times w}$ – совокупность uw независимых операций скалярного умножения векторов длины v.

В частности, в сверточных нейросетях операция свертки может быть сведена к умножению матриц: пусть слой свертки имеет ядро размерности $f \times f$, вход имеет размерность $w \times h$, полоса дополнения матрицы входных данных имеет ширину $p \times p$, ширина шага при прохождении ядра по матрице равна $s \times s$, i – количество входных каналов, o – количество выходных каналов. Тогда слой свертки может быть вычислен как операция умножения матриц размерности $(w' \cdot h') \times (i \cdot f \cdot f)$ и $(i \cdot f \cdot f) \times (o)$, где $w' = \frac{w-f+2p}{s} + 1$ и $h' = \frac{h-f+2p}{s} + 1$.

Протокол умножения нескольких чисел, разделенных на доли. В системе Tetrad реализованы протоколы умножения трех и четырех чисел.

Протокол умножения трех чисел генерирует $[\![\cdot]\!]$-разделение числа $z = abc$ по заданным $[\![a]\!]$, $[\![b]\!]$, $[\![c]\!]$ без необходимости выполнять два последовательных умножения. Для этого участники выполняют протокол, похожий на умножение двух чисел, где вычисляют $[\![z]\!] = [\![z - r]\!] + [\![r]\!]$, где $z - r = abc - r = (m_a - \lambda_a)(m_b - \lambda_b)(m_c - \lambda_c) - r = m_{abc} - m_{ac}\lambda_b - m_{bc}\lambda_a - m_{ab}\lambda_c + m_a\gamma_{bc} + m_b\gamma_{ac} + m_c\gamma_{ab} - \gamma_{abc} - r$.

Подобно протоколу Π_{Mult}, цель фазы предварительных вычислений – сгенерировать аддитивные доли $\gamma_{ab}, \gamma_{ac}, \gamma_{bc}, \gamma_{abc}$ силами P_1, P_2.

Доли γ-величин, которые P_1, P_2 не могут вычислить локально, могут быть вычислены участниками P_0, P_3. Далее они разделяют доли этих величин с участниками P_1, P_2 на фазе предварительных вычислений. Участники P_1, P_2 продолжают выполнять интерактивную фазу протокола аналогично Π_{Mult}, чтобы вычислить $z - r$. Таким образом, сложность интерактивной фазы протокола не изменяется – увеличивается только коммуникационная сложность фазы предварительных вычислений.

Протокол для умножения четырех чисел $z = abcd$ требует вычисления аддитивных долей величин $\gamma_{ab}, \gamma_{ac}, \gamma_{ad}, \gamma_{bc}, \gamma_{bd}, \gamma_{cd}, \gamma_{abc}, \gamma_{acd}, \gamma_{bcd}, \gamma_{abcd}$ на фазе предварительных вычислений. Протокол очень похож на протокол умножения трех чисел.

Протокол инъекции бита Π_{bitInj} позволяет вычислить $[\![bv]\!]$ по заданному булеву разделению $[\![b]\!]^B$ бита b и арифметическому разделению $[\![v]\!]$ переменной $v \in \mathbb{Z}_{2^l}$. Заметим, что $(bv)^R = (m_b \oplus \lambda_b)^R(m_v - \lambda_v) = \left(m_b^R + (\lambda_b)^R(1 - 2m_b^R)\right)(m_v - \lambda_v) = m_b^R m_v - m_b^R \lambda_v + (2m_b^R - 1)((\lambda_b)^R \lambda_v - m_v(\lambda_b)^R)$, где обозначение b^R символизирует представление бита b в кольце $\mathbb{Z}_{2^l}$ (все биты b^R равны нулю, кроме самого младшего бита, который равен b).

Во время предварительной обработки P_0 генерирует $\langle\cdot\rangle$-разделение $(\lambda_b)^R$, затем проверяет его корректность. $\langle\cdot\rangle$-доли величины $(\lambda_b)^R \lambda_v$ генерируются путем умножения $\langle(\lambda_b)^R\rangle$ и $\langle\lambda_v\rangle$, используя для этого протокол Π_{MulR}. В интерактивной фазе протокола каждая пара участников (P_1, P_3), (P_2, P_3) и (P_1, P_2) локально вычисляет аддитивное разделение величины $(bv)^R$, генерируя ее $[\![\cdot]\!]$-разделение с использованием протокола Π_{JSh}, и локально складывает эти доли, чтобы получить $[\![(bv)^R]\!]$.

Протокол забывающего выбора (oblivious selection) Π_{obv} позволяет по заданным $[\![\cdot]\!]$-долям переменных $x_0, x_1 \in \mathbb{Z}_{2^l}$ и $[\![b]\!]^B$, где $b \in \{0,1\}$, сгенерировать рандомизованные $[\![\cdot]\!]$-доли величины $z = x_b$. По сути своей протокол похож на протокол OT. Заметим, что $z = b(x_1 - x_0) + x_0$. Участники используют протокол Π_{bitInj}, чтобы вычислить $[\![b(x_1 - x_0)]\!]$, и суммируют их с $[\![x_0]\!]$, чтобы получить $[\![z]\!]$.

Протокол вычисления кусочно-полиномиальной функции. Кусочно-полиномиальная функция определяется следующим образом:

$$f(x) = \begin{cases} f_1(x), x < c_1 \\ f_2(x), c_1 \le x < c_2 \\ \dots\dots\dots \\ f_m(x), c_{m-1} \le x \end{cases},$$

где $f_1, \dots, f_m$ – общеизвестные многочлены, и $c_0 < c_1 < \dots < c_{m-1} < c_m$ – константы. Таким образом, f может быть вычислена в виде: $f(x) = \sum_i b_i f_i(x)$, где $f_0 = 0$, $f_m = 1$, а $b_i = 1$, если $y \ge c_i$ и 0 в противном случае для $\forall i = \overline{1, m}$. По заданным $[\![\cdot]\!]$-долям величины y можно получить $[\![\cdot]\!]^B$-доли битов $b_1, \dots, b_m$, используя протокол сравнения. Доли переменных $b_i \cdot (f_i - f_{i-1})$ могут быть сгенерированы путем m-кратного вызова протокола Π_{bitInj}, после которого локально выполняется сложение.

Нелинейные функции активации нейросетей: ReLU и сигмоидная – рассматриваются как частные случаи кусочно-полиномиальной функции.

Протокол отыскания минимального / максимального элемента вектора Π_{argmin} позволяет участникам вычислить индекс наименьшего элемента вектора $\vec{x} = (x_1, \dots, x_m)$, где каждый элемент $x_i \in \mathbb{Z}_{2^l}$ подвергнут $[\![\cdot]\!]$-разделению. Протокол выдает $[\![\cdot]\!]^B$-

разделенный вектор битов $\vec{b}$ длины m, который имеет 1 в позиции, в которой находится минимальная величина среди представленный в векторе $\vec{x}$, и 0 в противном случае. Для этого используется древовидный подход к поиску минимума, при котором вектор $\vec{b}$ корректируется с учетом вновь найденных минимумов. Первоначально все биты вектора $\vec{b}$ устанавливаются равными 1, затем элементы $\vec{x}$ попарно группируются и сравниваются, чтобы найти минимальное из двух чисел. Используя эту информацию, $\vec{b}$ обновляется таким образом, что b_j устанавливаются в 0 для тех $x_j \in \vec{x}$, которые не являются минимальными в паре. Остальные биты по-прежнему равны 1. Протокол рекурсивно выполняется для всех оставшихся $x_j \in \vec{x}$, которые оказались минимальными в паре. В конечном итоге остается только один бит $b_j \in \vec{b}$, равный 1, что свидетельствует о том, что x_j является абсолютным минимумом. Его индекс j и выдается в качестве ответа. Вычисление максимума выполняется аналогично при помощи протокола Π_{argmax}.

6. Другие четырехсторонние системы КМО

Система FLASH [6] предназначена для конфиденциального обучения и применения глубоких нейросетей. Основана на четырехсторонних протоколах безопасных вычислений в предположении о наличии среди участников не более одного получестного либо злоумышленного нарушителя. Система FLASH позволяет добиться свойства робастности.

Система FLASH использует аддитивное разделение секрета над кольцами $\mathbb{Z}_{2^l}$ и $\mathbb{Z}_2$. Разделение секретов происходит между участниками системы попарно. Для этого предполагается наличие двусторонних защищенных каналов передачи данных между каждой парой участников. Для достижения свойства робастности используются специальные примитивы пересылки сообщений, разделения входных данных протокола, восстановления отсутствующих либо поврежденных злоумышленником долей секретов из информации, доступной честным участникам.

Основные криптографические протоколы, реализованные в ядре системы:

- тензорное умножение векторов, составленных из разделенных секретов;
- извлечение старшего бита (MSB) разделенного числа;
- деление разделенного числа на степень числа 2 (усечение);
- умножение разделенного целого числа на разделенный бит;
- преобразование разделенного бита в разделенный арифметический эквивалент.

Авторы протестировали систему FLASH на алгоритмах линейной, логистической регрессий и глубоких нейронных сетях. Для тестирования использовался датасет MNIST. Эксперименты показали превосходство производительности системы FLASH над ближайшим трехсторонним аналогом ABY^3 [7]. С другими системами КМО сравнение не проводилось.

7. Применение систем КМО

Использование готовых систем КМО позволяет с наименьшими временными и трудовыми затратами решать практические вопросы обеспечения информационной безопасности интеллектуальных систем. Среди доступных на сегодняшний день систем КМО есть значительная доля тех, которые доступны для использования по свободной лицензии, а также тех, у которых открыт исходный код, что предоставляет возможности для дальнейшей доработки, модификации и оптимизации таких систем. Рассмотренные выше системы Tetrad и FLASH не относятся к числу продуктов с открытым исходным кодом. Однако, кроме неё, есть ряд других четырехсторонних систем КМО с открытым исходным кодом, в частности, Slalom [8], MP2ML [9].

Одновременно с этим следует отметить, что практически все существующие системы являются экспериментальными разработками, прототипами, но не создавались для широкого промышленного применения. Сведения о производительности компьютерных систем и пропускной способности каналов связи, требуемых для выполнения протоколов, чаще всего определялись по результатам относительно небольшой серии экспериментов с целью подтверждения возможности их практической реализации и сравнения с производительностью ранее известных систем. Экспериментальные данные нередко приводятся для тех архитектур нейросетей, на которых вновь предложенные решения показывают выгодный разрыв в производительности с ранее известными, но не опробуются на сложных архитектурах, которые представляют реальный практический интерес.

Другая трудность связана с тем, что ПО, разрабатываемое как экспериментальное, нередко является «сырым», вызывает значительные сложности при попытке инсталлировать его и воспроизвести проведенные авторами эксперименты. Очевидно, что готовые системы КМО в большинстве случаев не достигли уровня зрелости, при котором их можно было бы использовать для решения практических задач обеспечения информационной безопасности машинного обучения. В связи с этим актуальным остается продолжение исследований, разработка на основе результатов исследований новых методов и алгоритмов защиты данных и самих систем их обработки [10, 11].

Заключение

В ходе работы выполнено поисковое исследование и проведен обзор существующих систем КМО, которые реализованы преимущественно в виде прототипов и лабораторных образцов, основанных на четырехсторонних протоколах безопасных вычислений.

В качестве референс-модели выбрана система Tetrad, обладающая развитой функциональностью, имеющая подробно описанные и доказанные свойства стойкости к активному противнику. Система Tetrad обеспечивает стойкость в условиях, если не более чем один из четырех участников протоколов является активным нарушителем. Система позволяет обеспечить, как минимум, свойство честности, или равноправности, протокола (fairness), а как максимум – свойство робастности. С целью минимизации вычислительной и коммуникационной нагрузки участников протоколов их роли в протоколах, как правило, не симметричны. Особенности четырехсторонних вычислений, обеспечивающих стойкость к активному противнику, требуют множественных перекрестных вызовов низкоуровневых протоколов, что делает затруднительным или даже невозможным предоставление интерфейса к низкоуровневым протоколам по отдельности. Вместо этого в системе Tetrad реализуются протоколы безопасного вычисления функций, специфичных для КМО, таких как скалярное умножение векторов, умножение матриц, вычисление нелинейных функций активации, нахождение максимального или минимального элемента множества и др. С использованием этих функций становится возможным реализовать процедуры конфиденциального обучения и применения моделей машинного обучения, включая глубокие нейросетевые модели.

Системы КМО на основе БМВ продолжают активно развиваться и обладают большим потенциалом практического применения.

СПИСОК ЛИТЕРАТУРЫ:

1. Запечников Сергей В.; Щербаков Андрей Ю. Конфиденциальное машинное обучение на основе двусторонних протоколов безопасных вычислений. Безопасность информационных технологий, [S.l.], т. 28, № 4, p. 39–51, 2021. ISSN 2074-7136.DOI: http://dx.doi.org/10.26583/bit.2021.4.03.

2. Запечников Сергей В. Конфиденциальное машинное обучение на основе трехсторонних протоколов безопасных вычислений. Безопасность информационных технологий, [S.l.], т. 29, № 1, с. 30–43, 2022. ISSN 2074-7136. DOI: http://dx.doi.org/10.26583/bit.2021.4.03.
3. Evans D., Kolesnikov V., Rosulek M. A pragmatic introduction to secure multi-party computation. – 182 p. URL: https://securecomputation.org/docs/pragmaticmpc.pdf (дата обращения: 19.05.2022).
4. Koti Nishat & Patra Arpita & Rachuri Rahul & Suresh Ajith. (2021). Tetrad: Actively Secure 4PC for Secure Training and Inference. URL: https://eprint.iacr.org/2021/755.pdf (дата обращения: 19.05.2022).
5. Chaudhari Harsh & Rachuri Rahul & Suresh Ajith. (2019). Trident: Efficient 4PC Framework for Privacy Preserving Machine Learning. DOI: http://dx.doi.org/10.14722/ndss.2020.23005 (дата обращения: 19.05.2022).
6. Byali M., Chaudhari H., Patra A., & Suresh A. (2020). FLASH: Fast and Robust Framework for Privacy-preserving Machine Learning. Proceedings on Privacy Enhancing Technologies, 2020(2), p. 459–480. URL: https://eprint.iacr.org/2019/1365 (дата обращения: 19.05.2022).
7. Mohassel Payman & Rindal Peter. (2018). ABY 3: A Mixed Protocol Framework for Machine Learning. P. 35–52. DOI: http://dx.doi.org/10.1145/3243734.3243760.
8. Tramèr F., Boneh D. Slalom: Fast, verifiable and private execution of neural networks in trusted hardware. DOI: https://doi.org/10.48550/arXiv.1806.03287.
9. Fabian Boemer, Rosario Cammarota, Daniel Demmler, Thomas Schneider, and Hossein Yalame. 2020. MP2ML: a mixed-protocol machine learning framework for private inference. In Proceedings of the 15th International Conference on Availability, Reliability and Security (ARES '20). Association for Computing Machinery, New York, NY, USA, Article 14, p. 1–10. DOI: http://dx.doi.org/10.1145/3407023.3407045.
10. Xu Runhua & Baracaldo Nathalie & Joshi James. (2021). Privacy-Preserving Machine Learning: Methods, Challenges and Directions. URL: https://arxiv.org/pdf/2108.04417.pdf (дата обращения: 19.05.2022).
11. ACM CCS’21 conference program. URL: https://www.sigsac.org/ccs/CCS2021/conference-program.html (дата обращения: 19.05.2022)

REFERENCES:

[1] Zapechnikov Sergey V.; Shcherbakov Andrey Yu. Privacy-preserving machine learning based on secure two-party computations. IT Security (Russia), [S.l.], v. 28, no. 4, p. 39–51, 2021. ISSN 2074-7136. DOI: http://dx.doi.org/10.26583/bit.2021.4.03 (in Russian).
[2] Zapechnikov Sergey V. Privacy-preserving machine learning based on secure three-party computations. IT Security (Russia), [S.l.], v. 29, no. 1, p. 30–43, 2022. ISSN 2074-7136. DOI: http://dx.doi.org/10.26583/bit.2022.1.04 (in Russian).
[3] Evans D., Kolesnikov V., Rosulek M. A pragmatic introduction to secure multi-party computation. – 182 p. URL: https://securecomputation.org/docs/pragmaticmpc.pdf (accessed: 19.05.2022).
[4] Koti Nishat & Patra Arpita & Rachuri Rahul & Suresh Ajith. (2021). Tetrad: Actively Secure 4PC for Secure Training and Inference. URL: https://eprint.iacr.org/2021/755.pdf (accessed: 19.05.2022).
[5] Chaudhari Harsh & Rachuri Rahul & Suresh Ajith. (2019). Trident: Efficient 4PC Framework for Privacy Preserving Machine Learning. DOI: http://dx.doi.org/10.14722/ndss.2020.23005 (accessed: 19.05.2022).
[6] Byali M., Chaudhari H., Patra A., & Suresh A. (2020). FLASH: Fast and Robust Framework for Privacy-preserving Machine Learning. Proceedings on Privacy Enhancing Technologies, 2020(2), p. 459–480. URL: https://eprint.iacr.org/2019/1365 (accessed: 19.05.2022).
[7] Mohassel Payman & Rindal Peter. (2018). ABY 3: A Mixed Protocol Framework for Machine Learning. P. 35–52. DOI: http://dx.doi.org/10.1145/3243734.3243760.
[8] Tramèr F., Boneh D. Slalom: Fast, verifiable and private execution of neural networks in trusted hardware. DOI: https://doi.org/10.48550/arXiv.1806.03287.
[9] Fabian Boemer, Rosario Cammarota, Daniel Demmler, Thomas Schneider, and Hossein Yalame. 2020. MP2ML: a mixed-protocol machine learning framework for private inference. In Proceedings of the 15th International Conference on Availability, Reliability and Security (ARES '20). Association for Computing Machinery, New York, NY, USA, Article 14, p. 1–10. DOI: http://dx.doi.org/10.1145/3407023.3407045.
[10] Xu Runhua & Baracaldo Nathalie & Joshi James. (2021). Privacy-Preserving Machine Learning: Methods, Challenges and Directions. URL: https://arxiv.org/pdf/2108.04417.pdf (accessed: 19.05.2022).
[11] ACM CCS’21 conference program. URL: https://www.sigsac.org/ccs/CCS2021/conference-program.html (accessed: 19.05.2022).

Поступила в редакцию – 11 апреля 2022 г. Окончательный вариант – 22 мая 2022 г.
Received – April 11, 2022. The final version – May 22, 2022.