Почти гарантированная доставка в garbled circuit

Светлицкий Михаил[1]

[1]e-mail: minkal2sh@gmail.com

November 24, 2025

Abstract

Мы предлагаем протокол почти гарантированной доставки для двухсторонних вычислений на основе garbled circuits. Протокол преобразует вывод GC в SPDZ-шары с MAC, позволяя поочередно раскрывать биты с проверкой корректности. Даже при нечестном поведении одной из сторон её вычислительное преимущество ограничено не более чем двукратным ускорением получения результата.

Contents

1 Введение

Современные протоколы безопасных вычислений позволяют нескольким сторонам совместно вычислять функции от их приватных данных без раскрытия этих данных друг другу. Одним из наиболее известных подходов является *garbled circuit* (GC), предложенный Яо [1]. В двухсторонней вычислительной модели (*2-party computation*, 2PC) одна сторона, Алиса, играет роль *garbler*, создавая зашифрованную схему, а другая сторона, Боб, является *evaluator*, вычисляющим результат на основе шифрованной схемы.

Несмотря на широкое распространение GC, возникают определённые проблемы, связанные с гарантией получения корректного результата. В классическом подходе одна из сторон первым получает результат вычислений, что создаёт потенциал для атак, например, для публикации ложных значений или задержки раскрытия результата. В контексте критически важных приложений (финансовые вычисления, распределённое голосование, протоколы с ограниченным временем реакции) такие ситуации недопустимы.

Кроме того классический результат Фишера, Линча и Патерсона [2] демонстрирует, что в распределённых системах невозможно достичь гарантированной доставки при наличии хотя бы одного ненадёжного процесса.

В данной работе рассматривается алгоритм *почти гарантированной доставки* результата GC, который обеспечивает:

- защиту от намеренного обмана со стороны любой из сторон;
- возможность корректного получения результата обеими сторонами даже при попытках частичного мошенничества;
- ограничение вычислительного преимущества злоумышленника до не более чем двухкратного ускорения по сравнению с честной стороной;

Алгоритм состоит из двух логических частей:

1. Модификация исходного garbled circuit для последующего перехода к SPDZ-sharing. Это позволяет преобразовать зашифрованный ответ в форму, удобную для поэтапного раскрытия и проверки корректности.
2. Процедура поочередного раскрытия шэров Алисы и Боба, включающая MAC-check'и для проверки достоверности каждого бита. Такая процедура обеспечивает почти гарантированную доставку даже в условиях частичного мошенничества.

2 Переход от GC к SPDZ-sharing

В дальнейшем будем сокращать *garbled circuit* до *GC*. Предполагается, что Алиса играет роль *garbler*, а Боб — *evaluator*. Также считаем, что

по завершении исходной схемы GC выдаёт *незапутанный* (то есть не зашифрованный) ответ, и первым этот ответ получает Боб.

Для корректного последующего раскрытия значений на фазе SPDZ потребуется серия *MAC-check'ов*. Мы будем использовать MAC-check'и над битами, то есть над полем $\mathbb{F}_2$, и применять метод, описанный в работе в работе [3], с рядом упрощений процедуры раскрытия, обусловленных тем, что мы находимся в модели двухсторонних вычислений (2PC), а не в общем MPC.

Как именно мы хотим модифицировать оригинальную GC? Во-первых, мы хотим возвращть *зашифрованный* ответ исходной GC, дополненный несколькими нулями, которые в будущем будут служить сигнализационным механизмом того, что расшифровка была произведена корректно. Кроме того, модернизированная GC должна возвращать только *шар Боба* (его часть) от зашифрованного ответа, а также его часть MAC-check'ов. Шары Алисы известны ей заранее. Для этого необходимо добавить несколько дополнительных входов.

2.1 Структура входов и вычисления внутри модернизированного GC

Зафиксируем параметр безопасности s. Чем он больше, тем надёжнее выполняются MAC-check'и.

Входы Алисы. Для каждого бита (предполагаем, что их n) ответа оригинальной схемы GC Алиса генерирует случайным образом:

- битовую строку длины $s+1$, которая будет её шаром ответа $[enc(x)]$;
- битовую строку длины s, которая будет её шаром MAC-ключа $[a]$;
- битовую строку длины $s+1$, которая будет её шаром ответа $[xa]$.

Все эти данные Алиса подаёт в модернизированную GC.

Входы Боба. Боб, для каждого бита исходного ответа, генерирует случайным образом:

- битовую строку длины s, являющуюся его шаром $[a]$;
- битовую строку длины s, представляющую старшие биты его шара $[x]$ (младшие биты будут вычислены внутри схемы).

Остальные необходимые шары Боб получит непосредственно из выполнения GC.

Общий ключ шифрования. Алиса и Боб договариваются о симметричном ключе шифрования, известном обоим участникам. Этот ключ также подан в схему (для упрощения секркута считаем, что это делает Алиса).

2.2 Вычисления внутри модернизированной GC.

В начале работы модернизированная GC полностью вычисляет исходную GC и получает её ответ x. Далее выполняются следующие шаги:

1. Значение x дополняется заданным числом нулей (индикатор верной расшифроки в будущем) шифруется с использованием согласованного ключа, что даёт $enc(x)$.

2. Необходимо выполнить SPDZ-sharing значения $enc(x)$. Для этого берутся по 1 младшему биту из каждой битовой строки Алисы из ее шера [enc(x)] и выполняется операция XOR с $enc(x)$. Полученные биты являются младшими битами шара Боба от $enc(x)$. Объединяя их с битовыми строками, поданными Бобом, получаем полный шар Боба от $enc(x)$, который возвращается в выход схемы.

3. Вычисляются MAC-check'и. Сначала складываются шары MAC-ключа Алисы и Боба, поданные ими на вход:

$$a = a_1 + a_2 \bmod 2^{s+1},$$

получая общий MAC-ключ a.

4. Затем вычисляется произведение:

$$a \cdot enc(x) \bmod 2^{s+1}.$$

5. Вычитая из $a \cdot enc(x) \bmod 2^{s+1}$ выходные строки Алисы (также по модулю 2^{s+1}), получаем шар Боба от $[xa]$, который также подаётся в выход схемы.

Результат выполнения модернизированного GC. После завершения схемы:

- Алиса знает свои шары от $[enc(x)]$, $[a]$, $[xa]$, так как она сама их сгенерировала и подала в GC.

- Боб знает свой шар от $[a]$ и получает из GC свой шар от $[enc(x)]$ и $[xa]$.

Таким образом, стороны переходят в состояние SPDZ-sharing относительно значений $[enc(x)]$, $[a]$ и $[xa]$.

Несомненным преимуществом данного подхода является то, что Бобу на текущем этапе не требуется ничего публиковать. Следовательно, у него отсутствует возможность обмануть Алису, предоставив правдоподобно выглядящие, но ложные результаты.

3 Раскрытие шаров Алисы и Боба

На данном этапе мы находимся в модели SPDZ-sharing по модулю 2 и располагаем n зашаренными битами зашифрованного ответа. Для этих битов уже вычислены MAC-check'и. Симметричный ключ известен и Алисе, и Бобу, поэтому после полного раскрытия всех битов зашифрованного ответа обе стороны смогут расшифровать его и получить исходный ответ.

3.1 Идея алгоритма

Ключевая идея алгоритма заключается в том, что Алиса и Боб раскрывают биты поочередно, проверяя на каждом этапе, что никто не обманывает другую сторону.

3.2 Пошаговая процедура

Процедура раскрытия выполняется следующим образом:

1. Алиса публикует свой шар i-го бита.
2. Боб публикует свой шар того же бита.
3. Производится проверка корректности с помощью MAC-check'ей: обе стороны раскрывают соответствующие шары $[xa]$ и сравнивают их с $[x] \cdot [a]$.
    - Если равенство выполняется, алгоритм продолжается к следующему биту.
    - Если равенство не выполняется, значит кто-то попытался обмануть, и взаимодействие немедленно прекращается.

Таким образом, биты раскрываются последовательно. Если на каждом этапе никто не нарушает протокол, то в конце алгоритма как Алиса, так и Боб получают правильный ответ.

3.3 Гарантия почти гарантированной доставки

Если на каком-либо этапе одна из сторон начинает отправлять неверные данные или перестаёт взаимодействовать, то в лучшем случае мошенник получает на один бит правильного ответа больше, чем честная сторона. Благодаря индикатору корректной расшифровки который мы добавили ранее, оставшийся суффикс приходится можно найти через полный перебор, что требует вдвое больше вычислительных усилий для обманутой стороны, чем для мошенника. Таким образом достигается свойство *почти гарантированной доставки*.

References

[1] A. C.-C. Yao, *How to Generate and Exchange Secrets*, in Proceedings of the 27th Annual Symposium on Foundations of Computer Science (FOCS), 1986, pp. 162–167, IEEE, doi:10.1109/SFCS.1986.25.

[2] Michael J. Fischer, Nancy A. Lynch, Michael S. Paterson, *Impossibility of distributed consensus with one faulty process*, Journal of the ACM (JACM), 32(2):374–382, 1985.

[3] Ronald Cramer, Ivan Damgård, Daniel Escudero, Peter Scholl, Chaoping Xing, *SPDZ2k: Efficient MPC mod* 2^k *for Dishonest Majority*, Cryptology ePrint Archive, Report 2018/482, 2018.