Министерство науки и высшего образования Российской Федерации
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)
Институт прикладной математики и компьютерных наук
Кафедра компьютерной безопасности

ДОПУСТИТЬ К ЗАЩИТЕ В ГЭК
Руководитель ООП
канд. техн. наук, доцент
_____ В.Н. Тренькаев
«_26_» _января_ 2024 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА СПЕЦИАЛИСТА
(ДИПЛОМНАЯ РАБОТА)

БЕЗОПАСНОЕ ВЫЧИСЛЕНИЕ ЗНАЧЕНИЙ ФУНКЦИЙ НА ОСНОВЕ УПОРЯДОЧЕННЫХ ДВОИЧНЫХ ДИАГРАММ РЕШЕНИЙ

по специальности 10.05.01 Компьютерная безопасность,
специализация (профиль) «Анализ безопасности компьютерных систем»

Кумарасами Манодж Кумар

Научный руководитель ВКР
канд. техн. наук, доцент
_____ С.А. Останин
«_15_» _января_ 2024 г.

Автор работы
студент группы № 931825
_____ М.К. Кумарасами
«_15_» _января_ 2024 г.

Томск – 2024

Министерство науки и высшего образования Российской Федерации.
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)
Институт прикладной математики и компьютерных наук

УТВЕРЖДАЮ
Руководитель ООП
канд. техн. наук, доцент

_____ В.Н. Тренькаев

«_04_» _декабря_ 2023 г.

ЗАДАНИЕ

по выполнению выпускной квалификационной работы специалиста обучающемуся
Кумарасами Маноджу Кумару
Фамилия Имя Отчество обучающегося

по специальности 10.05.01 Компьютерная безопасность, специализация (профиль) «Анализ безопасности компьютерных систем»

1 Тема выпускной квалификационной работы
Безопасное вычисление значений функций на основе упорядоченных двоичных диаграмм решений

2 Срок сдачи обучающимся выполненной выпускной квалификационной работы:
а) в учебный офис / деканат – 15.01.2024 б) в ГЭК – 26.01.2024

3 Исходные данные к работе:
Объект исследования – Упорядоченные двоичные диаграммы решений
Предмет исследования – Безопасное вычисление значений функций
Цель исследования – Программная реализация протокола безопасных вычислений значений функций SFE–OBDD

Задачи:
Программная реализация протокола SFE–OBDD и исследование искаженных схем

Методы исследования:
Теоретический и экспериментальный

Организация или отрасль, по тематике которой выполняется работа, –
НИ ТГУ, Институт прикладной математики и компьютерных наук, кафедра компьютерной безопасности

4 Краткое содержание работы
Реализация и проведение экспериментов для протокола безопасных вычислений значений функций на основе упорядоченных двоичных диаграмм решений

Научный руководитель выпускной квалификационной работы
Зав. каф. компьютерной безопасности ТГУ _____ / С.А. Останин
должность, место работы *подпись* *И.О. Фамилия*

Задание принял к исполнению
студент группы № 931825 _____ / М.К. Кумарасами
должность, место работы *подпись* *И.О. Фамилия*

АННОТАЦИЯ

Дипломная работа содержит 25 страницу, 1 рисунка , две таблицы, 7 источников литературы.

Безопасное вычисление значений функций на основе упорядоченных двоичных диаграмм решений

Актуальность: Использование упорядоченной бинарной диаграммы принятия решений в качестве представления булевых функций в искаженных схемах при оценке безопасных функций

Объекты исследования: Искаженные схемы и упорядоченные двоичных диаграмм принятия решений

Цель работы: Программная реализация протокола SFE–OBDD

Метод исследования: Теоретический и экспериментальный на базе ЭВМ

Результат: Пропускная способность связи и затраченное время выполнения реализованного протокола для некоторых функций ниже, чем у системы fairplay

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В современном взаимосвязанном цифровом ландшафте сохранение конфиденциальности при совместном вычислении функций недоверчивыми друг другу субъектами стало первостепенной задачей. Безопасное оценивание функций (Secure Function Evaluation, SFE), представленное Яо, является ключевым методом, позволяющим нескольким сторонам совместно вычислять функции без необходимости взаимного доверия или опоры на доверенную третью сторону (Trusted Third Party, TTP). В рамках SFE клиент/сервер, в котором участвуют две стороны - каждая с приватными входными данными, - стремится вычислить произвольную функцию, сохраняя конфиденциальность этих входных данных и раскрывая только выходной результат, обеспечивая тем самым конфиденциальность и целостность.

Традиционно в области SFE преобладали два основных подхода: Искаженная схема (Garbled Circuits, GC) и гомоморфное шифрование (Homomorphic Encryption, HE). Первый основан на симметричной ключевой криптографии, обеспечивающей эффективные вычисления с постоянным числом раундов. Однако при этом требуется связь, пропорциональная размеру функции. С другой стороны, HE, используя гомоморфные свойства некоторых криптосистем, несет меньшие коммуникационные накладные расходы, но требует ресурсоемких операций с открытым ключом.

Значение SFE вышло за пределы теоретических рамок и нашло практическое применение в областях, критически важных для безопасности и конфиденциальности. Его применение произвело революцию в электронных аукционах, расширило возможности поиска данных, облегчило дистанционную и медицинскую диагностику, укрепило системы распознавания лиц, а также множество других приложений.

Упорядоченные двоичные диаграммы принятия решений (OBDD) - это графические представления, широко используемые в системах автоматизированного проектирования для представления булевых функций. Известные своей универсальностью, OBDD сыграли решающую роль в проверке символьных моделей, верификации комбинационной логики и валидации в параллельных конечно-составных системах.

Важно отметить, что их адаптивность распространяется на представление функций с произвольными областями и диапазонами, что потенциально может улучшить протоколы SFE. Интеграция OBDD в область SFE представляет собой привлекательную перспективу: использование протокола Яо путем преобразования представлений OBDD в схемы.

Это сближение сигнализирует о потенциальном сдвиге в методологии SFE, предлагая более эффективный подход к совместному вычислению функций при сохранении конфиденциальности входных данных. Это преобразование обещает расширить спектр протоколов, сохраняющих конфиденциальность. Кроме того, интеграция SFE с блокчейн-технологиями и распределенными вычислениями может предложить новые пути для создания децентрализованных, прозрачных и безопасных систем, которые будут способствовать укреплению доверия и сотрудничеству между различными участниками в цифровом пространстве.

1 Основные понятия и определения SFE

В этой главе мы рассмотрим обозначения и термины для основных понятия и определения SFE

1.1 Безопасное вычисление значений функций - SFE

Безопасное вычисление значений функций (Secure Function Evaluation, SFE) - криптографический протокол, разработанный для того, чтобы позволить двум или более сторонам совместно вычислять результат выполнения функции, обеспечивая при этом максимальную конфиденциальность их соответствующих входных данных. В контексте SFE каждая участвующая сторона в частном порядке предоставляет свои входные данные, и протокол организует вычисления таким образом, что стороны получают информацию исключительно о выходных данных функции, в то же время предотвращая любое раскрытие информации об отдельных входных данных друг другу. Этот протокол воплощает в себе основные принципы, включая сохранение конфиденциальности, корректность вычислений, защиту от различных типов атак и независимость от предположений о доверии, выходящих за рамки криптографических примитивов. SFE играет ключевую роль в *безопасных многосторонних вычислениях*, находя применение в таких областях, как защищенные аукционы, конфиденциальные запросы к базам данных и машинное обучение с сохранением конфиденциальности, где стороны хотят взаимодействовать, сохраняя конфиденциальность своих данных.

Предположим, что есть пользователи u, и каждый пользователь i обладает $x_i \in \{0,1\}^n$ и функцией $F_i : \{0,1\}^{nu} \to \{0,1\}^m$. Цель наша - создать протокол, в котором по завершении каждый пользователь i знает $F_i(x_1, \ldots, x_u)$, но не получает никакой дополнительной информации о x_j для $j \neq i$.

1.2 Изскаженных схемы Yao

Искаженные схемы Yao - криптографический протокол, используемый для защищенных двухсторонних вычислений, позволяющий двум недоверчивым сторонам совместно вычислять функцию на своих частных входных данных, не полагаясь на доверенную третью сторону. В этом протоколе функция должна быть определена как логическая схема.

История искаженных схем сложна. Эндрю Яо приписывают изобретение этой концепции, и он представил ее в устной презентации на FOCUS'86. Этот факт был задокументирован Одедом Голдрайхом в 2003 году. Первую письменную документацию по этой технике можно отнести к Goldreich, Micali и Wigderson в STOCK'87. Термин "искаженная схема"был введен Beaver, Micali и Rogaway в STOCK'90. Стоит отметить, что протокол Yao для решения проблемы миллионеров, хотя и является ранним примером безопасных вычислений, напрямую не связан с искаженными схемами.

По своей сути, искаженная схема это логическая схема, таблица истинности которой обфусцировано. Элегантность заключается в том, как это схема обфусцировано, чтобы позволить сторонам вычислять выходные данные, сохраняя при этом конфиденциальность входных данных. Протокол выполняет две роли: искажатель и оценщика.

1. Вычисляемая функция (например, в проблеме миллионеров это функция сравнения) представляется в виде Булевой схемы с двумя входами. Вид схемы известен обеим сторонам. Этот шаг может быть сделан заранее третьей стороной.

2. Алиса искажает (зашифровывает) схемы. Алису называют искажатель.

3. Алиса отправляет искаженную схемы Бобу вместе с ее зашифрованными входными данными.

4. Боб, с помощью забывчивой передачи данных, получает свои зашифрованные входные данные от Алисы.

5. Боб восстанавливает (расшифровывает) схемы и получает зашифрованные результаты вычисления.

1.3 Построения искаженных схем

Основными компонентами построения искаженных схем являются:

Искаженные значения: В *(Garbled Circuits, GC)* вместо использования двоичных значений (0 или 1) вычисления выполняются на "искаженных значениях которые являются секретами случайного вида. Каждому проводу в схеме присваиваются два искаженных значения. Эти значения выглядят случайными и не раскрывают фактическое двоичное значение, которое они представляют. Эффективно сконструированные искаженные значения состоят из симметричного ключа (длиной t бит) и случайного бита перестановки.

Искаженные таблицы: Для каждого элемента в схеме создается "искаженная таблица". Эта таблица позволяет выполнять вычисления с искаженными значениями. Таблица гарантирует, что при наличии искаженных входных данных для элемента можно расшифровать только соответствующее искаженное выходное значение. Таблица построена таким образом, что предотвращает раскрытие какой-либо информации о других искаженных выходных значениях или фактических двоичных входах и выходах вентиля. Записи в искаженной таблице представляют собой зашифрованные тексты для всех возможных комбинаций входных данных, зашифрованные с использованием симметричного шифрования. Расположение этих записей имеет решающее значение и разработано таким образом, чтобы не раскрывать никакой информации о входных значениях.

Схемы искажения: Схема искажения (рисунок 1) представляет собой набор из пяти алгоритмов $G = (Gb, En, De, Ev, ev)$:

Gb (искажение): Этот алгоритм является единственным вероятностным по своей природе (остальные алгоритмы являются детерминированными). Он принимает в качестве входных данных параметр безопасности k и функцию f и возвращает тройку (F, e, d). F - это функция (или схема), представляющая f, e - функция кодирования, а d - функция декодирования. Ключевая часть определения заключается в том, что мы имеем для любого $x \in \{0,1\}^n$:

$$f(x) = d(F(e(x)))$$

Для входных данных x, $X = e(x)$ преобразует входные данные в искаженные входные данные, $Y = F(X)$ оценивает искаженные входные данные с помощью искаженной функции F, а $y = d(Y)$ преобразует искаженный выходной сигнал в конечный выходной y. Конечно, y должно быть равно $f(x)$.

En (Encode): На самом деле, мы уже упоминали функцию кодирования, поэтому этот алгоритм просто берет функцию e и входные данные x и выводит $X = e(x)$, преобразуя любые входные данные в искаженные входные данные:

$$En(e, x) = e(x) = X$$

Ev (вычислять): Этот алгоритм принимает искаженную функцию F и искаженный входной сигнал X и выдает искаженный выходной сигнал Y:

$$Ev(F, X) = F(X) = Y$$

De (Decode): Аналогично, этот алгоритм принимает функцию декодирования d и искаженный выходной сигнал Y и возвращает конечный выходной сигнал y:

$$De(d, Y) = d(Y) = y = f(x)$$

ev (evaluate): Этот алгоритм является тем, который вычисляет $f(x)$:

$$ev(f, x) = f(x)$$

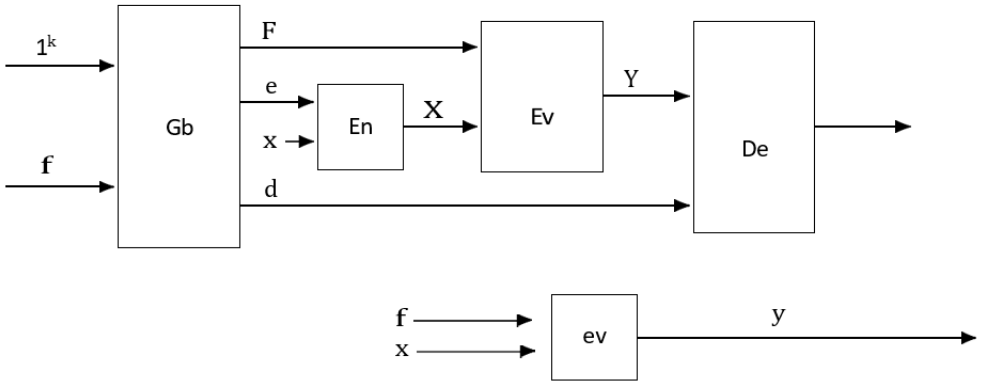

Рисунок 1 - Компоненты схемы искажения

1.4 Протокол забывчивой передачи

Забывчивая передача (Oblivious transfer, OT) - криптографический протокол, позволяющий Получателю (R) выбирать и получать одну часть информации от Отправителя(и), не раскрывая их выбора, гарантируя, что Отправитель не знает о выбранном значении. Он имеет жизненно важное применение в защищенной связи и безопасное многосторонних вычислениях.

Забывчивой передачи (OT) играет важную роль в протоколах безопасных вычислений. Она служит фундаментальным строительным блоком для обеспечения безопасной связи и вычислений между двумя сторонами, обычно называемыми Отправителем

(ями) и получателем (R). OT по своей сути асимметричен по своей природе, что означает, что он предполагает получение информации одной стороной без получения другой стороной каких-либо знаний о том, что было передано. Его значение в криптографии и информатике проистекает из его роли в обеспечении взаимодействия между участниками с сохранением конфиденциальности.

В основополагающем результате Impagliazzo и Rudich установили глубокую связь между OT и теорией сложности, в частности, проблемой P против NP. Они продемонстрировали, что если бы можно было свести забывчивую передачу к примитиву с симметричным ключом, такому как односторонние функции или псевдослучайные функции (PRF), это означало бы разрешение давнего вопроса о соотношении P и NP. Это сокращение предполагает, что присущая OT асимметрия имеет решающее значение для его криптографической стойкости и полезности в защищенных вычислениях.

Однако в работе Beaver появилось практическое соображение, которое показало, что в определенных сценариях пакетное выполнение протоколов OT может быть оптимизировано таким образом, чтобы требовалось лишь небольшое количество операций с открытым ключом. Конструкция Beaver позволила повысить эффективность, но по своей природе она не была похожа на черный ящик. В частности, для этого требовалось представить PRF в виде схемы и оценить ее в рамках многосторонних вычислений (MPC). Хотя этот результат имел значительные теоретические последствия, он не сразу нашел применение в практических криптографических приложениях из-за его нетривиальных вычислительных требований.

Параметры:

1. Две стороны:

   — Отправитель (и): Обладает входными секретами x_1 и x_2, каждый из которых состоит из n двоичных значений, т.е. $x_1, x_2 \in \{0,1\}^n$.

   — Приемник (R): Содержит бит выбора b, который может принимать значения из набора $\{0,1\}$.

   Протокол:

   1. Генерация ключа получателем (R):

— R генерирует пару открытого и закрытого ключей, обозначаемую как (sk, pk).

— R также выбирает случайный ключ, pk', из пространства открытых ключей.

— В зависимости от значения бита выбора b:

  — Если $b = 0$, R отправляет пару открытых ключей отправителю S: (pk, pk').

  — Если $b = 1$, R отправляет пару открытых ключей S в обратном порядке: (pk', pk).

2. Шифрование отправителем (ями):

— После получения пары открытых ключей (pk_0, pk_1) отправитель S переходит к шифрованию своих входных секретов:

— S шифрует x_0 с помощью pk_0, в результате чего получается зашифрованный текст e_0: $e_0 = \mathrm{Enc}_{pk_0}(x_0)$.

— S шифрует x_1 с помощью pk_1, получая зашифрованный текст e_1: $e_1 = \mathrm{Enc}_{pk_1}(x_1)$.

3. Расшифровка получателем (R):

— После получения зашифрованных значений e_0 и e_1, получатель R расшифровывает зашифрованный текст e_0, используя свой закрытый ключ sk.

— Примечательно, что R не может расшифровать второй зашифрованный текст e_1, поскольку в нем отсутствует соответствующий секретный ключ, тем самым сохраняя конфиденциальность нераскрытого секрета.

Этот протокол иллюстрирует фундаментальные принципы передачи без уведомления, когда Получатель выборочно получает одну часть информации от Отправителя, не обращая внимания на другую. Использование асимметричного шифрования обеспечивает конфиденциальность передаваемых секретов и поддерживает гарантии конфиденциальности протокола.

2 Упорядоченных двоичных диаграмм решений

Упорядоченные бинарные диаграммы решений (OBDD) структура данных, используемая в компьютерной науке и инженерии для представления и управления булевыми функциями. В отличие от традиционных булевых схем, которые используют логические элементы для вычисления функции, OBDD является направленным ациклическим графом (DAG), который обеспечивает компактное и эффективное представление булевых функций.

Основным преимуществом структуры данных OBDD является ее способность представлять сложные булевы функции с использованием компактного представления, независимого от порядка переменных, используемых в функции. Путем упорядочивания переменных OBDD обеспечивает эффективное и интеллектуальное управление функциями, включая возможность обнаружения эквивалентности между двумя функциями, вычисления подфункций данной функции и выполнения экзистенциальной и универсальной квантификации.

2.1 Представление

Мы предполагаем, что все функции, которые нужно представить, имеют одинаковые n аргументов, обозначенных как $x_1, \ldots, x_n$. При выражении системы, такой как комбинаторная логическая сеть или булево выражение, как булевой функции, нам необходимо установить порядок ввода или атомарных переменных. Этот порядок должен оставаться одинаковым для всех функций, которые нужно представить.

Функция, получающаяся в результате замены некоторого аргумента x_i функции f константой b, называется restriction of f (иногда называемый кофактором) и обозначается $f|x_i = b$. То есть для любых аргументов $x_1, \ldots, x_n$,

$$f|x_i = b(x_1, \ldots, x_n) = f(x_1, \ldots, x_{i-1}, b, x_{i+1}, \ldots, x_n)$$

Используя это обозначение, разложение Шеннона функции вокруг переменной x_i задается формулой:

$$f = x_i \cdot f|x_i = 1 + \overline{x_i} \cdot f|x_i = 0$$

Аналогично, функция, получающаяся в результате замены некоторого аргумента x_i функции f функцией g, называется композицией f и g и обозначается $f|x_i = g$. То есть для любых аргументов $x_1, \ldots, x_n$,

$$f|x_i = g(x_1, \ldots, x_n) = f(x_1, \ldots, x_{i-1}, g(x_1, \ldots, x_n), x_{i+1}, \ldots, x_n)$$

Некоторые функции могут зависеть не от всех аргументов. Набор зависимостей функции f, обозначаемый I_f, содержит те аргументы, от которых зависит функция, т.е.

$$I_f = \{i \mid f|x_i = 0 \neq f|x_i = 1\}$$

2.2 Операция ограничения

Операция ограничения в упорядоченной двоичной диаграмме решений (OBDD) - процедура, которая изменяет диаграмму для представления функции, полученной путем привязки некоторых переменных к определенным логическим значениям. Чтобы выполнить операцию ограничения, мы берем существующий OBDD и выбираем переменную, которую мы хотим исправить. Затем мы разделяем все узлы на диаграмме на две группы: узлы, для которых выбранной переменной присвоено значение true, и узлы, для которых выбранной переменной присвоено значение false. Затем мы отбрасываем узлы из второй группы и сохраняем те, что находятся в первой группе, соответствующим образом обновляя пути к оставшимся узлам.

3 Цель и Задача

3.1 Протокол SFE-OBDD

Протокол SFE-OBDD представляет собой криптографический метод, используемый для защищенных двухсторонних вычислений, позволяющий двум сторонам, Алисе и Бобу, совместно вычислять логическую функцию, не раскрывая свои индивидуальные входные данные друг другу. Он использует упорядоченные двоичные диаграммы принятия решений (OBDD) для представления логической функции. Алиса обрабатывает OBDD с помощью своих входных данных, а затем "искажает"его, отправляя искаженную версию Бобу. Боб, используя свои входные данные и секреты, полученные с помощью неявных передач, расшифровывает и вычисляет результат функции. Протокол обеспечивает конфиденциальность и безопасность, поскольку ни одна из сторон не изучает входные данные другой, а результат выполнения функции известен обеим сторонам только в конце вычисления. протокол описывает следующим образом:

Ввод: Ввод обеих сторон включает в себя $OBDD(f)$ для булевой функции $f(x_1, x_2, \cdots, x_n)$ с порядком $x_1 < x_2 < \cdots < x_n$. Кроме того, Алиса обладает входами $(i_1, \ldots, i_k)$, соответствующими первым k переменным $x_1, \ldots, x_k$, и Боб имеет входы $(i_{k+1}, \ldots, i_n)$.

1. Алиса выполняет следующие шаги:

   1.1. Она проходит через $OBDD(f)$, используя свой ввод $(i_1, \cdots, i_k)$, что приводит к узлу v_{init} на уровне k.

   1.2. Она независимо и случайно создает $(n-k)$ пар секретов $(s_1^0, s_1^1), \cdots, (s_{n-k}^0, s_{n-k}^1)$. Кроме того, для каждого узла v в $OBDD(f)$, уровень которого находится между k и $n-1$, Алиса также создает секрет s_v.

   1.3. Она назначает случайную метку каждому узлу, уровень которого находится между k и n. Мы обозначаем случайно назначенную метку узла v как $label(v)$.

   1.4. Затем Алиса дополняет $OBDD(f)$ некоторым количеством фиктивных узлов (чтобы обеспечить, что Боб всегда проходит $n-k$ узлов в своей фазе протокола).

   1.5. Алиса шифрует все узлы, уровень которых находится между k и $n-1$, следующим образом. Пусть v будет узлом в $OBDD(f)$ таким, что $k \leqslant level(v) \leqslant n-1$ и определим $level(v) = \ell$. Шифрование узла v, обозначенное как $E^{(v)}$, является

меткой и случайно упорядоченной парой шифртекстов

$$\Big(label(v)\ ,\ E_{s_v \oplus s^0_{\ell-k+1}}(label(low(v)) \,\|\, s_{low(v)})\ \ ,\ \ E_{s_v \oplus s^1_{\ell-k+1}}(label(high(v)) \,\|\, s_{high(v)})\Big)\ \ ,$$

где метки предваряются секретом с символом-разделителем и порядок шифртекстов определяется броском монеты. Грубо говоря, секреты, соответствующие 0-преемнику и 1-преемнику узла v, шифруются с секретом, соответствующим v и его уровню.

Обратите внимание, что фиктивные узлы имеют ту же структуру, что и обычные узлы, за исключением того, что пара шифртекстов содержит шифрование одного и того же сообщения, поскольку фиктивные узлы имеют одинаковых 0 и 1-преемников. При условии, что схема шифрования семантически безопасна, это не представляет проблемы, так как ключи выбираются равномерно случайным образом.

Наконец, существуют два терминальных узла вида $(b, label(t_b))$ для $b = 0$ или 1. Напомним, что $OBDD(f)$ имеет два терминальных узла, обозначаемых как 0 и 1, которые находятся на уровне n.

1.6. Как только Алиса закончит шифрование, она отправляет Бобу шифрование всех узлов, уровень которых находится между k и n, и секрет $s_{v_{init}}$, соответствующий узлу v_{init} на уровне k. Мы называем это зашифрованным OBDD.

2. Боб выполняет следующие шаги:

2.1. Он принимает участие в $n - k$ 1-из-2 забывчивых передачах, чтобы получить секреты, соответствующие его вводу. Например, если его ввод i_j это 0, то он получает (уровневый) секрет s^0_{j-k}; в противном случае, он получает секрет s^1_{j-k}.

2.2. Теперь Боб готов начать свои вычисления. Предположим, $i_{k+1} = 0$. Используя s^0_1 и $s_{v_{init}}$, он расшифровывает оба шифртекста в $E^{(v_{init})}$ и решает, какой из них дает правильный результат, используя проверяемое свойство диапазона шифрования. Теперь у Боба есть как $s_{low(v)}$ (секрет, соответствующий 0-преемнику v_{init}), так и $label(low(v))$ (которая говорит Бобу, какой зашифрованный узел используется для оценки его следующего ввода). Продолжая таким образом, Боб в конечном итоге получает метку, соответствующую одному из терминальных узлов, которая определяет результат OBDD на общих входных данных. Боб отправляет этот результат Алисе.

Обзор протокола SFE-OBDD

Входы

— У Алисы входы, соответствующие первым k переменным $x_1, \ldots, x_k$.

— У Боба входы, соответствующие последним $n-k$ переменным $x_{k+1}, \ldots, x_n$.

— У обеих сторон есть OBDD для булевой функции $f(x_1, x_2, \ldots, x_n)$ с заданным порядком переменных.

Шаги протокола

Шаги Алисы

1. Обход OBDD: Используя свои входные данные, Алиса проходит по OBDD и достигает узла v_{init} на уровне k.

2. Генерация секретов:

   — Создание $(n-k)$ пар секретов $(s_1^0, s_1^1), \ldots, (s_{n-k}^0, s_{n-k}^1)$.

   — Для узлов между уровнями k и $n-1$ создание секрета s_v для каждого из таких узлов.

3. Присвоение случайных меток: Случайное присвоение меток узлам между уровнями k и n.

4. Дополнение OBDD: Добавление фиктивных узлов для обеспечения того, что Боб всегда проходит $n-k$ узлов на своем этапе.

5. Шифрование узлов:

   — Шифрование узлов между уровнями k и $n-1$.

   — Шифрование включает метки и пары шифротекста, содержащие секреты последующих узлов.

   — Фиктивные узлы имеют ту же структуру, но содержат зашифрованные пары с одинаковым сообщением.

6. Отправка зашифрованного OBDD: Передача зашифрованных узлов и секрета $s_{v_{\text{init}}}$ Бобу.

Шаги Боба

1. Протокол неведения: Проведение $n - k$ 1-из-2 протоколов неведения для получения секретов, соответствующих его входам.

2. Вычисление:

   — С использованием полученных секретов и $s_{v_{\text{init}}}$ дешифрует шифротексты для обхода зашифрованного OBDD.

   — Проверочный выбор верных последующих узлов на каждом уровне на основе его секретов ввода.

   — Продолжение до достижения терминального узла, определяющего результат OBDD на общих входа

   x.

Фиктивные узлы

Включение фиктивных узлов обеспечивает, что Боб проходит одинаковое количество узлов независимо от входа Алисы, предотвращая получение Бобом информации обо входах Алисы из образца обхода.

Безопасное вычисление

— Протокол обеспечивает безопасное вычисление без раскрытия индивидуальных входов, используя шифрование, секретное разделение и безопасную схему обхода в зашифрованном OBDD.

— Структура направлена на сохранение конфиденциальности и приватности входных данных при возможности вычислений на общих данных.

Трудности и решения

— Пропуск уровней: Протокол решает проблему узлов, позволяющих Бобу пропускать уровни в OBDD, вводя фиктивные узлы и поддерживая постоянную длину обхода.

Вывод

Протокол обеспечивает возможность для обеих сторон вычислить булеву функцию без раскрытия их входных данных друг другу. Это достигается путем шифрования узлов, генерации и обмена секретами и обеспечения постоянного обхода OBDD.

3.2 Описание программная реализация

Реализация протокола искаженный OBDD Реализация протокола безопасного оценки упорядоченных двоичных диаграмм принятия решений (OBDD) между двумя сторонами, Алисой и Бобом. Протокол позволяет безопасно вычислять BDD, не раскрывая приватная информации о входных данных, используемых в вычислении. Реализация выполнена на языке программирования Java

```
main/
└── sfe/
    ├── bdd/
    │   ├── BDD.java
    │   ├── BDD_Inference.java
    │   └── BDD_Writer.java
    └── proto/
        ├── BDD_Crypt.java
        ├── OBDD_Evaluation.java
        ├── Obfuscated_BDD.java
        └── Protocol.java
```

Файл Protocol.java является основным файлом в пакете ‘sfe.bdd.proto‘. В первую очередь он служит точкой входа в приложение, а его функциональность зависит от определенных системных свойств. Файл определяет общедоступный класс с именем "Protocol"в пакете "sfe.bdd.proto". Метод ‘main‘, который является точкой входа программа, проверяет наличие системных свойств с именами ‘BOB‘ или ‘ALICE‘. В зависимости от того, какое свойство задано, программа переходит на разные пути выполнения: - Если задано значение "BOB он выполняет метод "main"вложенного или связанного класса с именем "Bob". - Если задано значение ‘ALICE‘, он выполняет метод ‘main‘ вложенного или связанного класса с именем ‘Alice‘. Если ни одно из свойств не задано, выводится сообщение инструкции: "Необходимо использовать -DALICE или -DBOB". Файл предназначен для сценария, включающего два разных режима работы или роли, как указано ‘Alice‘ и ‘Bob‘.

3.2.1 Общий функционал

Общение Алисы и Боба: Протокол обеспечивает обмен данными между Алисой и Бобом по сетевому соединению для безопасного выполнения вычислений BDD. Шифрование и вычисление: Код использует различные техники шифрования для шифрования и оценки узлов BDD, обеспечивая конфиденциальность во время вычислений. Частичная оценка: Поддерживает частичную оценку BDD на основе предоставленных значений входных данных. Оптимизация и сжатие: Включает техники оптимизации, такие как устранение узлов и использование сжатия (GZIP) для эффективной передачи данных. Обработка ошибок: Код обрабатывает различные сценарии для входных данных и рандомизации, обеспечивая надежность выполнения.

3.2.2 Основные компоненты

- Классы и пакеты: Код организован в различные классы и пакеты (‘sfe.bdd.proto‘), отвечающие за функциональность Алисы и Боба.

- Обработка входных данных: Анализирует входные данные из командной строки для настройки протокола, включая указание ролей (Алиса/Боб), данных соединения, файлов BDD и значений входных данных.

- Манипулирование BDD: Использует методы для манипулирования BDD, такие как создание полной BDD, нормализация и частичная оценка на основе предоставленных значений входных данных.

- Шифрование и передача данных: Применяет криптографические техники, такие как шифрование/дешифрование ключей, OT (Забывчивая передача), сериализация для безопасной передачи информации BDD между Алисой и Бобом.

- Метрики производительности: Измеряет и отображает метрики производительности, такие как время шифрования, время оценки и размер передаваемых данных.

Ход выполнения:

- Сторона Алисы: Устанавливает соединение с Бобом и подготавливает BDD для оценки на основе значений входных данных. Шифрует узлы BDD, выполняет частичную оценку и безопасно передает зашифрованные данные Бобу.

- Сторона Боба: - Ожидает подключение Алисы и получает зашифрованные данные BDD. - Участвует в процессе оценки с использованием полученных данных, выполняя OT и дешифрацию для получения оцененного BDD. Предложения по оптимизации: Реализация стратегий для устранения узлов, определения максимального количества

для частичной оценки и перестановка индексов для повышения эффективности. Предоставление возможности для сжатия (GZIP) для минимизации размера передаваемых данных

3.3 Экспериментальные результаты

В таблице 1 представлено время выполнения (EET) в секундах и его разбивка на подзадачи для SFE-OBDD. Оцениваемые функции включают Add, And, Eq и Mil с вариациями для 4, 8 и 16 бит. Данные каждой функции разбиты на компоненты: CC, OT, Eval и EET

Таблица 1 - Прошедшее время выполнения (EET) в секундах и их разбивка на подзадачи для SFE-OBDD

Functions	PG	CC	OT	Eval	EET
Add4	15.90%	5.44%	77.69%	0.94%	0.35
Add8	13.07%	3.80%	82.07%	1.05%	0.57
Add16	11.39%	2.36%	84.42%	1.83%	0.66
And4	12.42%	5.73%	81.21%	0.64%	0.41
And8	9.37%	4.02%	85.94%	0.67%	0.45
And16	12.35%	2.75%	84.39%	0.41%	0.74
Eq4	7.26%	5.38%	81.33%	0.64%	0.42
Eq8	11.30%	3.90%	84.12%	0.65%	0.47
Eq16	8.92%	2.48%	88.14%	0.45%	0.62
Mil4	13.2%	5.62%	80.20%	0.96%	0.34
Mil8	9.35%	3.86%	86.12%	0.67%	0.47
Mil16	8.03%	3.03%	87.88%	0.52%	0.77

— PG (Parsing and Garbling) в таблицах 1 и 2 представляет процент от общего времени выполнения, выделенного на фазу парсинг и искажения в протоколе. Этот показатель имеет решающее значение для оценки эффективности и быстродействия протокола, поскольку он показывает, сколько общего времени вычислений затрачивается на эти задачи начальной настройки.

— CC (Circuit Communication): Этот компонент измеряет объем передаваемых данных, когда Алиса отправляет искаженную схему или структуру Бобу в процессе оценки защищенной функции.

— OT (Oblivious Transfer): Эта часть относится к компоненту протокола, в котором Боб получает секреты, соответствующие его вводимым данным, без того, чтобы Алиса знала, какие конкретные секреты получил Боб. Это важный шаг в обеспечении конфиденциальности вводимых данных.

— Eval (Оценка): Здесь измеряется время или ресурсы, необходимые Бобу для оценки искаженной структуры или схемы, которую он получает от Алисы. Этот шаг имеет решающее значение для вычисления выходных данных функции при сохранении конфиденциальности отдельных входных данных.

— EET (прошедшее время выполнения): Это общее время, затраченное на весь процесс оценки защищенной функции, охватывающий все вышеперечисленные компоненты (CC, OT и Eval).

Таблица 2 - Прошедшее время выполнения (EET) в секундах и их разбивка на подзадачи для Fairplay

Functions	PG	CC	OT	Eval	EET
Add4	18.6%	17.00%	64.12%	0.23%	0.54
Add8	15.1%	15.96%	68.38%	0.53%	0.64
Add16	11.79%	9.67%	78.22%	0.48%	0.83
And4	12.02%	21.44%	66.30%	0.24%	0.41
And8	9.37%	16.07%	70.96%	0.19%	0.64
And16	12.75%	7.35%	83.48%	0.38%	0.77
Eq4	18.26%	12.85%	68.16%	0.49%	0.43
Eq8	15.30%	16.52%	67.68%	0.36%	0.66
Eq16	13.92%	9.74%	76.70%	0.23%	0.86
Mil4	25.2%	8.36%	64.28%	0.19%	0.54
Mil8	16.35%	9.25%	74.40%	0.38%	0.63
Mil16	18.03%	9.01%	72.99%	0.22%	0.91

В таблице 2 показано время выполнения (EET) в секундах и его разбивка для Fairplay. Аналогично таблице 1, она включает те же функции (Add, And, Eq, Mil) и их битовые вариации. Компонентами являются PG, CC, OT, Eval и EET.

ЗАКЛЮЧЕНИЕ

В данной работе были реализованы протокол искажения схем на основе упорядоченных двоичных диаграмм решений. Эксперименты проведены с функциями, такими как ADD, AND, Eq, Mil. Эксперименты показали что пропускная способность связи и затраченное время выполнения (EET) реализованного протокола ниже, чем у системы fairplay.

Экспериментальные результаты дают важную информацию. Примечательно, что протокол продемонстрировал улучшенные показатели производительности как с точки зрения пропускной способности, так и времени, затраченного на выполнение (EET), при сопоставлении с системой Fairplay. Этот вывод особенно примечателен, поскольку он предполагает, что подход, основанный на OBDD, потенциально может предложить более эффективные вычислительные решения в области безопасных многопартийных вычислений.

Более низкая пропускная способность и EET означают, что протокол SFE-OBDD может более эффективно выполнять задачи передачи и обработки данных, что является решающим фактором в приложениях, требующих быстрой и безопасной обработки данных. Такое повышение производительности можно объяснить присущей структуре OBDD эффективностью при обработке логических функций, которая составляет суть этого протокола.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ

1. R.E. Bryant. Graph-based algorithms for Boolean function manipulation. IEEE Trans. on Computers, C-35:677–691, 1986.

2. A. C.-C. Yao. How to generate and exchange secrets. In Proceedings of the 27th IEEE Annual Symposium on Foundations of Computer Science (FOCS), pages 162–167. IEEE Computer Society.

3. D. Evans, W. Melicher, and S. Zahur. Garbled circuit intermediate language. - URL: http://www.mightbeevil.org/gcparser (дата обращения 10.01.2024 г.)

4. Y. Lindell and B. Pinkas. An efficient protocol for secure two-party computation in the presence of malicious adversaries. In In EUROCRYPT 2007, pages 52–78, 2007.

5. L. Kruger, S. Jha, E.-J. Goh, and D. Boneh. Secure function evaluations with ordered binary decision diagram. In Proceedings of the 13th ACM conference on Computer and communications security (CCS'06), Alexandria, VA, Oct. 2006.

6. M. Naor and B. Pinkas. Efficient oblivious transfer protocols. In Proceedings of SODA '01, Washington, DC, 2001.

7. C.Y. Lee, Representation of Switching Circuits by Binary-Decision Programs, Bell System Technical Journal, Vol. 38, July 1959, pp. 985-999

АНТИПЛАГИАТ
ОБНАРУЖЕНИЕ ЗАИМСТВОВАНИЙ

СПРАВКА

Томский Государственный Университет

о результатах проверки текстового документа
на наличие заимствований

ПРОВЕРКА ВЫПОЛНЕНА В СИСТЕМЕ АНТИПЛАГИАТ.ВУЗ

Автор работы: Кумарасами Манодж Кумар
Самоцитирование рассчитано для: Кумарасами Манодж Кумар
Название работы: Кумарасами М.К Дипломная работа
Тип работы: Дипломная работа
Подразделение: НИ ТГУ, Институт прикладной математики и компьютерных наук

РЕЗУЛЬТАТЫ

■ **ОТЧЕТ О ПРОВЕРКЕ КОРРЕКТИРОВАЛСЯ:** НИЖЕ ПРЕДСТАВЛЕНЫ РЕЗУЛЬТАТЫ ПРОВЕРКИ ДО КОРРЕКТИРОВКИ

СОВПАДЕНИЯ	13%		СОВПАДЕНИЯ	13%	
ОРИГИНАЛЬНОСТЬ	87%		ОРИГИНАЛЬНОСТЬ	87%	
ЦИТИРОВАНИЯ	0%		ЦИТИРОВАНИЯ	0%	
САМОЦИТИРОВАНИЯ	0%		САМОЦИТИРОВАНИЯ	0%	

ДАТА ПОСЛЕДНЕЙ ПРОВЕРКИ: 23.01.2024

ДАТА И ВРЕМЯ КОРРЕКТИРОВКИ: 23.01.2024 16:11

Структура документа: Проверенные разделы: основная часть с.3-22

Модули поиска: Перефразирования по СПС ГАРАНТ: аналитика; Перефразированные заимствования по коллекции Интернет в английском сегменте; Переводные заимствования по коллекции Гарант: аналитика; Переводные заимствования IEEE; Переводные заимствования по коллекции Интернет в русском сегменте; Перефразирования по Интернету; Переводные заимствования по коллекции Интернет в английском сегменте; Перефразированные заимствования по коллекции Интернет в русском сегменте; Интернет Плюс*; Перефразирования по коллекции IEEE; Патенты СССР, РФ, СНГ; Переводные заимствования по Интернету (EnRu); Перефразирования по Интернету (EN); Цитирование; Шаблонные фразы; Издательство Wiley; Переводные заимствования по eLIBRARY.RU (EnRu); СПС ГАРАНТ: нормативно-правовая документация;

Работу проверил: Пахомова Елена Григорьевна

ФИО проверяющего

Дата подписи: 23.01.2024

Подпись проверяющего

Чтобы убедиться
в подлинности справки, используйте QR-код,
который содержит ссылку на отчет.

Ответ на вопрос, является ли обнаруженное заимствование корректным, система оставляет на усмотрение проверяющего. Предоставленная информация не подлежит использованию в коммерческих целях.