Красненкова Анастасия, аспирантка РТУ МИРЭА.
Специальность «Теоретическая информатика, кибернетика»
Научный руководитель: д.т.н.,проф. Оцоков Ш.А.

Особенности шифрования данных в смарт-контрактах на блокчейне InnoChain.

Базовые определения

Под блокчейном мы будем подразумевать систему распределенного реестра. Системой распределенного реестра называют децентрализованную базу данных – записей о событиях, «содержащих критически важную управленческую, юридическую, финансовую и иную информацию, которые хранятся, одновременно создаются и обновляются на всех носителях у всех участников реестра на основе заданных алгоритмов, обеспечивающих ее тождественность у всех пользователей реестра» (см. [1]). С математической точки зрения, под блокчейном будем понимать непрерывную цепочку блоков данных, связанных между собой с помощью криптографических алгоритмов. Каждый такой блок содержит информацию о транзакциях – командах, изменяющих состояние системы и представляющих собой записи в указанной базе данных.

Базовые определения

Помимо этого, каждый блок содержит криптографическую хэш-сумму предыдущего блока, временные данные (время записи в блокчейн как самого блока, как и составляющих его транзакций), нонс (уникальный идентификатор транзакции, защищающий от её подмены злоумышленником), а также двоичное дерево Меркла, содержащее данные транзакции (в корне дерева находятся хэшированные данные всей транзакции). Решение о корректности вносимых в базу транзакций, а также порядок их исполнения определяются протоколом консенсуса – специальным механизмом, регулирующим работу блокчейна.

Отметим, что транзакции осуществляются смарт-контрактами – специальными программами, реализующими задуманную бизнес-логику. Смарт-контракты могут создаваться с помощью самых разных языков программирования.

Договоримся, что состоянием системы распределенного реестра будет называться состояние соответствующей ей базы данных на текущий момент времени, а изменением этого состояния — внесение новой записи в базу.

Базовые определения

Главные черты блокчейн-технологии, помимо децентрализованности, – прозрачность и неизменяемость. Последняя характеристика означает, что данные не могут быть изменены после того, как они были записаны в базу. Указанное обстоятельство предъявляет повышенные требования к проверке правильности вносимых данных, с чем успешно справляется формальная верификация.

Под *формальной верификацией* будем понимать строгое математическое доказательство того, что программа (или отдельные ее компоненты) соответствуют её спецификации – т.е. тому, как она должна работать, согласно проекту. Отметим, что формальная верификация часто эффективно дополняет тестирование и активный аудит программного кода.

Отличительные особенности блокчейна InnoChain

В общем виде структура аккаунта, который является минимальной образующей единицей блокчейна InnoChain, может быть описана следующим образом (см.[4, стр.476]): «В структуре данных аккаунта хранится: идентификатор (accountId), множество открытых ключей (publicKeys, для отправки транзакций от имени аккаунта ее необходимо подписать каждым соответствующим секретным ключом), тип аккаунта (type), количество транзакций аккаунта (nonce), код смарт-контракта (code), хранилище дополнительных данных аккаунта (storage). В структуре данных транзакции хранится: время создания (timestamp), идентификатор аккаунта, от имени которого отправлена транзакция (from), идентификатор аккаунта для развертывания смарт-контракта (to), порядковый номер транзакции (nonce, который должен соответствовать полю nonce аккаунта отправителя), начальное состояние смарт-контракта (data)».

Отличительные особенности блокчейна InnoChain

Если переходить от устройства аккаунта к самому блокчейну, то его архитектура является пятиуровневой:

1. Уровень языка смарт-контрактов;

2. Уровень исполнения смарт-контрактов;

3. Уровень функционала узла;

4. Уровень протокола консенсуса;

5. Уровень операционной системы.

Первый уровень исчерпывающе описывается следующей цитатой (см.[4, стр.475]): «С точки зрения пользователя, CPP InnoChain представляет собой распределенную децентрализованную базу данных со встроенным языком программирования — предметно-ориентированным языком смарт-контрактов, которые выполняются на всех узлах CPP и изменяют (дополняют) её состояние. Смарт-контракты разрабатываются на предметно-ориентированном языке, встроенном в язык CakeML. Для языка смарт-контрактов требуется наличие формальной семантики, позволяющей формулировать и доказывать свойства функциональной корректности смарт-контрактов. В частности, формальная операционная семантика существует для языка CakeML».

Отличительные особенности блокчейна InnoChain

Что касается второго архитектурного уровня, то в InnoChain код смарт-контрактов напрямую транслируется в машинный код с помощью верифицированного компилятора CakeML, что дает дополнительные гарантии надежности получаемого машинного кода.

Третий уровень – уровень узла, на котором разворачиваются смарт-контракты, – также является безопасным, так как узел, в свою очередь, написан на CakeML (см.[5]).

В качестве протокола консенсуса (четвертый уровень) в InnoChain выбран HotStuff BFT (см.[6]). *Протокол консенсуса* определим как алгоритм обмена специальными сообщениями между узлами блокчейна. Данный обмен происходит для того, чтобы часть транзакций принять к исполнению (закоммитить), а часть отбраковать как невалидные (и отказаться от их исполнения). Если хотя бы одна транзакция закоммитилась, то система распределённого реестра переходит в новое состояние, а алгоритм консенсуса продолжает свою работу над новыми транзакциями. Протокол консенсуса HotStuff BFT является оптимальным по сложности (линейное время работы в идеальных условиях; при внедрении криптографических стандартов ГОСТ 34.10 сложность поднимается до квадратичной, но квадратичной сложностью обладает большинство протоколов консенсуса, в частности, TenderMint). Кроме того, HotStuff BFT обладает свойством оптимистичного отклика: если большинство узлов проголосовали за блок, то он принимается (коммитится) и инициируется следующий раунд голосования.

Далее, для протокола консенсуса HotStuff BFT было верифицировано свойство безопасности, т.е. было математически доказано, что два правильных (незлонамеренных) узла никогда не будут одновременно находиться в двух противоречащих друг другу состояниях. В рамках InnoChain были построены формальные модели, в рамках которых возможна дальнейшая верификация протокола HotStuff BFT – например, свойства живости (см.[6]).

Отличительные особенности блокчейна InnoChain

Перейдем теперь к пятому уровню – уроню операционной системы. В InnoChain активно используется формально-верифицированное микроядро seL4 (см. [4]). Термин «микроядро» обозначает операционную систему с урезанным функционалом, тем не менее, достаточным для запуска необходимых процессов.

Чтобы формальная верификация проходила успешно, необходимы наличие формальной семантики и соответствующего инструментария у языка программирования, на котором формальная верификация проводится.

Вот что пишут авторы работы [4, стр.479]: «Поскольку семантика CakeML описана в HOL4, то появляется возможность записывать любые программы на CakeML в HOL4 — о таких программах говорят, что они «глубоко вложены» в HOL. Таким образом, определения, функции и типы данных в логике HOL можно передать на вход транслятору — и получить их CakeML-аналоги, глубоко вложенные в HOL. Важно отметить, что вместе с кодом на CakeML транслятор генерирует сертификат — доказательство того, что функции, записанные в HOL, работают так же, как и полученный код на CakeML. Обычные функции в HOL4 описывают «чистые» математические вычисления, то есть такие, в которых нет ввода-вывода и взаимодействия с внешним миром. В рамках проекта CakeML была разработана do-нотация для HOL4, которая позволяет описывать программы, в которых присутствует ввод-вывод, непосредственно в HOL4».

Поэтому смарт-контракты разрабатываются в стиле, близком к императивному (с некоторыми элементами функционального программирования), что облегчает как разработку, так и тестирование и активный аудит кода. За успешную верификацию отвечает как раз функциональная компонента CakeML.

Особенности формальной верификации в Innochain

В рамках блокчейна Innochain вся верификация также осуществляется с помощью HOL4 — специального автоматизированного средства для доказательства теорем, или, другими словами, прувера (см.[2]). Отличительная особенность данного прувера состоит в том, что он использует верифицированный компилятор на языке функционального программирования CakeML. Следует отметить, что корректность работы HOL4 базируется на корректности метода резолюций для языков высших порядков и корректности алгоритма унификации для них же (см.[10]).

Кроме того, отличительной чертой Innochain является использование формальной верификации на всех уровнях его функционирования, а именно: при разработке смарт-контрактов, разворачивании смарт-контрактов, разработке и функционировании ноды – узла, на котором разворачиваются смарт-контракты и где коммитятся транзакции (подробнее см. предыдущий слайд и работы [3], [4], [5], [6]).

Поэтому при моделировании смарт-контракта с помощью второпорядковой логики предикатов и элементов временной логики становится возможным доказательство теорем о свойствах разработанного смарт-контракта, в том числе и о критически важных его характеристиках.

Особенности формальной верификации в Innochain

Зачастую выделяют пять основных компонент, или групп свойств, подлежащих верификации (см.[7]):

1) Безопасная арифметика;

2) Корректность спецификации функций смарт-контракта (иногда это называют низкоуровневой верификацией);

3) Аутентификация и авторизация пользователей;

4) Доказательство теорем о так называемых инвариантах функций;

5) Доказательство высокоуровневых свойств функций контракта (по-другому данные свойства иногда называют метриками)

Рассмотрим подробно каждый из пяти пунктов.

1) Часто при работе с числовыми значениями возникает так называемая «ошибка переполнения» сверху или снизу, ошибки при делении на ноль. Поэтому, если в смарт-контрактах происходит операция деления, то её надо «обернуть» в безопасную функцию, запрещающую деление на ноль. Следует также избегать использования действительных чисел в качестве входных параметров и выходных данных. Необходимо доказывать утверждения о корректной имплементации функций, обеспечивающих безопасную арифметику: во-первых, они не должны менять состояние контракта, а во-вторых – возвращать ожидаемый разработчиком результат.

2) Свойства второй группы для каждой функции исследуемого смарт-контракта утверждают следующее – функция работает как положено: в случае правильно подобранных параметров возвращает нужный результат и меняет состояние блокчейна; в случае ошибки – выбрасывает соответствующее исключение и оставляет состояние неизменным. Иногда теоремы такого вида называют теоремами корректности. В зависимости от архитектуры смарт-контракта возможны доказательства теорем как в импликативной форме (импликация выступает формальным аналогом союза «если, то»), так и в форме эквивалентностей (более сильной форме).

Особенности формальной верификации в Innochain

3) Теоремы авторизации доказывают, что выполнение тех или иных действий, прописанных в смарт-контракте (например, перевод денежных средств или же создание новой скидки) разрешено только определенным пользователям (например, администратору сети). Это позволяет снизить вероятность хакерской атаки при условии, что злоумышленник не получил root-привилегии.

4) Часто при правильном выполнении функций контракта прослеживается некоторая закономерность: например, какая-то характеристика, несмотря на применение ряда операций, остается неизменной. Свойство программы, которое сохраняется при проведении вычислений, называется инвариантом (например, вычисления в теле цикла сохраняют некоторое свойство). Инвариантность свойств смарт-контракта можно строго доказать, в основном, моделируя детерминированный конечный автомат – ДКА (см.[9]).

Например, состояние «Отказ в обслуживании» является неизменяемым для произвольной транзакции, т.е., раз попав в него, мы уже не можем данное состояние покинуть (это финальное состояние ДКА). Другой пример инварианта: чем больше идентификатор транзакции, тем позже она была создана.

Следует отметить, что чем больше инвариантов нам удалось выявить, тем выше вероятность того, что контракт функционирует правильно.

Особенности формальной верификации в Innochain

5) Еще одним эффективным механизмом, демонстрирующим, что бизнес-логика смарт-контракта реализована корректно, является доказательство так называемых метрик. Под метриками мы будем понимать утверждения о «высокоуровневых» свойствах функций, т.е. свойствах, не связанных напрямую с корректной программной имплементацией.

Классический пример высокоуровневого свойства – "Для успешной оплаты заказа баланс клиента должен быть больше или равен сумме, рассчитанной смарт-контрактом".

Приведем чуть более специальный пример метрики: "Итоговая цена, рассчитанная за литр топлива (параметр «Цена за единицу услуги»), не может быть больше, чем «Цена со стеллы»".

Чем сложнее архитектура контракта, тем больше метрик можно доказать относительно работы его функций.

Особенности формальной верификации в Innochain

Под понятие метрики подпадают и свойства, указывающие на невозможность осуществления хакерских атак, например, «Невозможна reentrancy атака, позволяющая неограниченно списывать средства с клиентского баланса, или: если по транзакции с данным идентификатором уже произошли налив топлива и списание средств с баланса, то повторные налив и списание средств невозможны». Последнее свойство можно формализовать так:

Theorem no_reentrancy_attack:

∀ ident cardIdent azs fact_volume data state balance state_new.∃ transaction.

let

SUCCESS = (Success (SCInt balance), state_new);

in

(recording_data [SCTransactionId ident; SCCardId cardIdent; SCInt azs; SCInt fact_volume] state = SUCCESS ∧ list_find (λt.t.id) state.transactions ident = SOME transaction) ⇒ transaction.order_status ≠ OrderIsCompleted

где функция recording_data осуществляет окончательный расчет итоговой суммы транзакции с помощью фактически налитого объема топлива, изменяет баланс клиента и осуществляет запись финальной транзакции в блокчейн, поменяв состояние системы (из state – в state_new). При этом для успешной отработки функции необходимо, чтобы статус заказа не был равен значению «Заказ завершён» (OrderIsCompleted); в противном случае функция не сможет начать свою работу и выбросит соответствующее исключение, оставив состояние системы неизменным.

Особенности шифрования чувствительных данных в Innochain

Однако, несмотря на достаточно высокую надежность смарт-контрактов на данном блокчейне, актуальной остается проблема шифрования чувствительных данных, таких, как: пин-коды карт, балансы клиентов, значения клиентских скидок и т. п.

Проблема заключается в том, что непосредственно в коде смарт-контракта сложно реализовать алгоритмы шифрования указанных данных. Если же в качестве входных параметров для функций произвольного смарт-контракта, которые работают с чувствительными данными, подавать уже зашифрованные данные, то может возникнуть проблема, связанная с арифметическими вычислениями, особенно с безопасным целочисленным делением, которое необходимо, например, при вычислении объема топлива для заправки транспортного средства и др. Здесь на помощь может прийти концепция гомоморфного шифрования (см.[8]). Гомоморфное шифрование позволяет осуществлять арифметические операции над зашифрованными данными и после расшифровки получать корректный с математической точки зрения результат.

Также в ряде случаев могут быть полезны решения, связанные с таким нетривиальным подходом в шифровании, как протоколы с нулевым разглашением (Zero-Knowledge Proof, ZKP) – подробнее см., например, [11].

Особенности шифрования чувствительных данных в InnoChain

Отсюда сразу же вытекает интересная задача: как согласовать формальную верификацию кода смарт-контракта и алгоритмов шифрования, используемых для части данных смарт-контракта? Не нарушается ли при осуществлении шифрования корректность работы функций контракта, а также их работа с API Web-приложений? Представляется интересным доказать серию математических теорем, описывающих данное взаимодействие и устанавливающих корректность процедур шифрования; тем более, что для протокола консенсуса подобные попытки успешно предпринимались в рамках блокчейна InnoChain (см. [3]).

Также важно доказать тот факт, что для реализации функций произвольного смарт-контракта, осуществляющих арифметические операции, достаточно не полностью гомоморфного шифрования (в противном случае, нарушались бы требования ГОСТ по шифрованию с помощью эллиптических кривых, и сильно возрастала бы алгоритмическая сложность). При этом будет использоваться весьма сильное допущение о том, что у клиентских приложений всегда есть бесперебойный доступ в Интернет.

Актуальное решение по сокрытию данных

Предлагается создать второй блокчейн и развернуть на нём вспомогательный смарт-контракт, работающий исключительно с зашифрованными данными; в первом – основном – контракте всё расчёты производятся без шифрования. Данное решение позволит избежать полностью гомоморфного шифрования, не желательного в силу его высокой сложности. Для реализации данной архитектуры планируется использовать промежуточную структуру – т. н. «мост», связывающий гетерогенные блокчейны.

Известно, что «мосты» являются ахиллесовой пятой многих блокчейнов в плане информационной безопасности (например, см. [12]), поэтому мы планируем провести всестороннюю формальную верификацию указанного архитектурного решения (путем соответствующего математического моделирования) и тем самым повысить надежность сокрытия чувствительных данных.

Список литературы:

1. https://www.cnews.ru/news/top/2019-07-17_blokchejn_prineset_rossijskoj_ekonomike_16_trillionov?ysclid=lawnx82wad290471241

2. https://hol-theorem-prover.org/

3. Ziborov K.V., Dunaev G.A., Vasiliev N.K., Rezin R.M. Approaches of formal verification of smart-contracts in HOL. Lobachevskii Journal of Mathematics, ISSN: 1995-0802 (Print), 1818-9962 (Online).

4. Merkin-Janson L. A. Rezin R. M.,Vasilyev N. K. Architecture of the Formally-Verified Distributed Ledger System InnoChain. MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 27, NO. 4, 2020.

5. Sadykov R.F., Dunaev G.A., Rezin R.M., Vasiliev N.K. Formal verification of the distributed leger system in HOL4. Lobachevskii Journal of Mathematics, ISSN: 1995-0802 (Print), 1818-9962 (Online).

6. Kukharenko V. A., Ziborov K. V., Sadykov R. F., Naumchev A. V., Rezin R. M., Merkin-Janson L. A. InnoChain: a Distributed Ledger for Industry with Formal Verification on all Implementation Levels. Modeling and analysis of information systems, vol. 27, no. 4, pp. 454-471, 2020.

7. Tianyu Sun, Wensheng Yu. A Formal Verification Framework for Security Issues of Blockchain Smart Contracts. Electronics, no 9, pp.255-278, 2020.

8. Chizhov I., Garazha A., Gerasimov I, Nikolaev M. On the Usage of Fully Homomorphic Encryption Libraries // International Journal of Open Information Technologies. Vol.9, No 3. 2021

9. Кудрявцев В.В, Алешин С. В., Подколзин А. С. Введение в теорию автоматов. М. Наука, 1985.

10. Крупский В.Н., Плиско В.Е. Математическая логика и теория алгоритмов. Москва, «Академия», 2013.

11. https://z.cash/technology/zksnarks/

12. https://swarm.ptsecurity.com/binance-smart-chain-token-bridge-hack/

Спасибо за внимание!