Солодилов Ю. Н.

Смарт-контракт: преимущества и недостатки

Кафедра информационной безопасности ШЕН ДВФУ

Аннотация:

Данная статья посвящена рассмотрению смарт-контрактов, их основных преимуществ и недостатков. Рассмотрен смарт-контракт Ethereum, его основные элементы и наиболее распространённые уязвимости данного типа «умных» контрактов.

Ключевые слова: *смарт-контракт, «умный» контракт, Ethereum, уязвимости смарт-контрактов, преимущества и недостатки, блокчейн.*

Смарт-контракт – это цифровые протоколы для передачи информации, которые используют математические алгоритмы для автоматического выполнения транзакции после соблюдения установленных условий и полного контроля процесса.

Код используется для ввода всех условий договора, заключенного между сторонами сделки, в blockchain. Обязательства участников предоставляются в интеллектуальном контракте в форме «если-то» (например: «если Сторона А переводит деньги, тогда Сторона В, передает права на квартиру»). Могут быть два или более участников, и они могут быть отдельными лицами или организациями. Как только данные условия будут выполнены, смарт-контракт самостоятельно выполняет транзакцию и гарантирует, что соглашение будет соблюдаться.

Основные объекты «умного» контракта:

1) Подписанты – стороны смарт-контракта, которые принимают или отказываются от условий контракта с использование ЭП.
2) Предмет договора. Предметом договора может являться объект, который находится внутри среды существования самого контракта, или

должен обеспечиваться прямой доступ умного контракта к договору без участия человека.

3) Условия. Условия умного контракта должны иметь полное математическое описание, которое возможно запрограммировать в среде существования умного контракта. Здесь описывается логика исполнения пунктов предмета договора.
4) Децентрализованная платформа. Платформа для распределённого хранения контракта необходима его запись в блокчейне платформы.

Преимущества смарт-контрактов:

1) Скорость. Смарт-контракты предполагают автоматизированный процесс, который не требует личного участия сторон.
2) Независимость. Смарт-контракты исключают возможность вмешательства третьих сторон. Гарантия на транзакцию – сама программа.
3) Безошибочность. Автоматический процесс без прямого участия человека обеспечивает высокую точность при выполнении транзакций.
4) Надежность. Данные не могут быть изменены или уничтожены. Если одна из сторон сделки не выполняет свои обязательства, другая сторона защищается условиями умного договора.

Недостатки смарт-контрактов:

1) Отсутствие правового регулирования.
2) Сложность реализации. Сложность интеграции смарт-контрактов с элементами реального мира занимает большое количество средств.
3) Невозможность изменения смарт-контракта. Если участники достигают более выгодного соглашения или появляются новые факторы, то изменить контракт они не смогут.

Смарт-контракты Ethereum. Наиболее распространенные уязвимости. Наиболее известные эксплоиты.

Блокчейн платформа Ethereum позволяет ее пользователям реализовывать на ней процессы разной сложности, поскольку она является

программируемой платформой. Данная платформа позволяет написание своей собственной логики взаимодействия между пользователями в виде смарт-контрактов. Код этих контрактов хранится в распределенном реестре и выполняется виртуальной машиной Ethereum (EVM), которая является основной этой блокчейн платформы.

Основные элементы:

- EVM. Работает с EVM байт-кодом, в котором заложен исходный код контракта. Этот байт-код представляет собой 16-теричную строку. Каждой операции соответствует цена ее исполнения – gas. Это гарантирует защиту от угроз типа Denial of Service (Отказ в обслуживании, DoS)
- Плата за выполнение кода. Существует для мотивирования узлов сети Ethereum выполнять вычисления. Это гарантирует вознаграждение узлу, который исполняет транзакцию, а также решает проблему, при которой выполнение смарт-контракта может зациклиться.
- Структура смарт-контракта. Смарт контракт состоит из набора переменных, описывающих его состояние, и набора функций, каждая из которых содержит в себе ту или иную информацию, заложенную в нее разработчиком контракта. Переменные контракта могут быть публичными и приватными, но по-настоящему приватными они быть не могут, так как состояние смарт-контракта хранится в публичном реестре. Функции контракта определяют «входные точки», через них можно инициировать выполнение кода. Он также содержит функцию конструктор, которая вызывается при создании контракта, и fallback функцию, у которой пустая сигнатура, она вызывается в особых случаях, таких как: при переводе на контракт денег и, если вызываемая функция не найдена.

Уязвимости смарт-контрактов

Некоторые наиболее распространенные типы уязвимостей смарт-контрактов на основе EVM (Ethereum Virtual Machine):

1) **Re-Entrancy (Повторный вход).** Когда смарт-контракт «вызывает» другой аккаунт он может выбрать количество gas'а, которое разрешит использовать вызываемой стороне. Если целевой счет является контрактом, он будет исполнен и может использовать gas. Если же контракт вредоносный и у него высокий gas, он может попытаться «перевызвать» вызывающему ему абоненту – повторный вызов. Так злоумышленник может использовать данную уязвимость для вывода средств из уязвимого контракта.
2) **Unhandled Exceptions (Необработанные исключения).** Некоторые низкоуровневые операции в Solodity, например send, которая используется для отправки Ethereum'а не выдает исключение при сбое, а возвращает логическое значение. Если возвращаемое значение не обработано, вызывающая сторона продолжает выполнение контракта.
3) **Locked Ethereum (Заблокированный эфириум).** Смарт-контракты могут получать эфириум. Но существует несколько причин, по которым полученные средства могут быть заблокированными. Первая причина – это то, что контракт может зависеть от другого контракта, который был разрушен, то есть его код был удален. Если это был единственный способ отправки эфира, то это приведет к блокировке средств. Вторая причина – когда контракт всегда выполняется и заканчивается gas при попытке отправить эфириум, тогда это тоже приводит к блокировке средств.
4) **Transaction Order Dependency (Зависимость порядка транзакции).** Так как в Ethereum несколько транзакций включается в один блок, то это означает что состояние контракта можно обновить несколько раз в одном блоке. Если порядок двух транзакций вызывает один и тот же контракт, изменяя его конечное состояние. То злоумышленник может использовать данную ошибку.
5) **Integer overflow (Целочисленное переполнение).** Целочисленное переполнение – это распространенный тип ошибка в различных языках

программирования, но в контексте Ethereum, это может иметь серьёзные последствия. Например, если счетчик цикла был переполнен, создавая бесконечный цикл, то средства контракта могут быть полностью заморожены. Это может быть использовано злоумышленником, если у него есть способ увеличение количество итераций цикла, например, зарегистрировав достаточное количество пользователей для того, чтобы вызвать переполнение.

6) **Unrestricted Action (Неограниченные действия).** Контракты часто выполняют авторизацию пользователей, проверяя отправителя сообщения, чтобы ограничить действия, которое может предпринять пользователь. Обычно только хозяин может уничтожить контракт или установить нового владельца смарт-контракта. Эта проблема может возникнуть не только в том случае, если разработчик забывает выполнять определенные проверки, но и в том случае, если злоумышленник может выполнять произвольный код, например имея возможность управлять адресом вызова.

Известные атаки

TheDAO exploit. Эксплойт DAO - одна из самых печально известных ошибок в блокчейне Ethereum.

DAO представляет собой полностью автоматический инвестиционный фонд. Желающие могут выставлять свои предложения на публику, аналог Kickstarter. Предложения, которые получили поддержку сообщества, получают финансирование, а часть их прибыли делится между «инвесторами». Все процессы платформы происходят автоматически: голосование, финансирование и распределение прибыли. Сотрудники DAO не влияют на них.

DAO полностью построен на базе Ethereum, что дает полную поддержку смарт-контрактов этой технологии.

Злоумышленник использовал уязвимость в функции контракта splitDAO, которая предназначалась для создания дочерних версий проекта.

Уязвимость типа повторный вход позволяет вызывать splitDAO рекурсивно в процессе каждого отделения и, таким образом, получать «эфиры» в течение одной единственной транзакции.

По логике, в которой оперирует Ethereum, данная атака не является преступлением. Правила системы DAO – это код, и непреднамеренные уязвимости, в результате ошибки программиста является частью контракта, как и описанные условия. Действия, которые проделал злоумышленник, предусмотрены правилами проекта DAO.

Проблема заключается в том, что в исполнение умного контракта невозможно вмешаться, так же, как и вернуть средства с уже проведенных транзакций, так как это гарантирует блокчейн Ethereum.

Для уменьшения ущерба от данной атаки, создатель Ethereum предложил провести «мягкое ветвление» (soft fork) криптовалюты, для того чтобы отменить любые транзакции, которые привели к утечке средств. А затем провести и «жесткое ветвление» (hard fork), которое вернет пострадавшим утекшие средства.

Список используемых источников

1) Осмоловская, А. С. Смарт-контракты: функции и применение. [Электронный ресурс]. – Режим доступа: https://www.elibrary.ru/download/elibrary_34872429_69687020.pdf // Бизнес образование в экономике знаний. – 2018. – №2(10). – С. 54-56.
2) Daniel Perez., Benjamin Livshits. Smart Contract Vulnerabilities: Vulnerable Does Not Imply Exploited [Electronic resource] – Mode of Access: https://arxiv.org/pdf/1902.06710.pdf
3) Noama Fatima Samreen, Manar H. Alalf, A Survey of Security Vulnerabilities in Ethereum Smart Contracts [Electronic resource] – Mode of Access: https://arxiv.org/pdf/2105.06974.pdf