Безопасность ICO контрактов (6)

Александр Половьян
alex@ledgers.world

Токен

- Единица ценности созданная организацией для взаимодействия с клиентами

- Правила обращения регулируются смарт-контрактом

Сайты с токенами

- https://etherscan.io/tokens
- https://ethplorer.io/top

- Важно различать
  - Информация из чейна (объективная)
    Байты в storage смарт-контракта
    Например: количество токенов, крупные собственники, объем сделок
  - Информация из других контрактов
    Например: рыночная стоимость, капитализация

Экономика токенов

- Правила обращения токенов регулирует владелец токена
- Токенами управляют люди
- Токены (смарт-контракты) могут интегрироваться в другие смарт-контракты

Интеграция токенов

- Обмен одних токенов на другие токены
- Обмен токенов на Ether
- Платежи токенами в смарт-контракты
- ...
- Нужен стандарт
  ERC = Ethereum Request for Comments
  EIP = Ethereum Improvement Proposal

ERC20

ERC20

- Минимальный контракт для учета токенов допускающий взаимодействие с другими контрактами

- Не стандарт, а convention

- https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md

- https://github.com/ethereum/eips/issues/20

- https://www.reddit.com/r/ethereum/comments/3n8fkn/lets_talk_about_the_coin_standard/

Методы

	returns	**constant**	**mandatory**
name()	**string** name	yes	no
symbol()	**string** symbol	yes	no
decimals()	**uint8** decimals	yes	no
totalSupply()	**uint256** totalSupply	yes	yes
balanceOf(**address** owner**)**	**uint256** balance	yes	yes

Методы перевода

	returns	**constant**	**mandatory**
transfer(**address** to**,** **uint256** value**)**	**bool** success	no	yes
transferFrom(**address** from**,** **address** to**,** **uint256** value**)**	**bool** success	no	yes
approve(**address** spender**,** **uint256** value**)**	**bool** success	no	yes
allowance(**address** owner**,** **address** spender**)**	**uint256** remaining	yes	yes

Зачем transferFrom?

	Инициатор	Списание	Получение	Объем
transfer	msg.sender	msg.sender	**address** to	**uint256** value
transferFrom	msg.sender	**address** from	**address** to	**uint256** value

Обработка ошибок

	Недостаточно средств	Нет прав (allowance)
transfer	throw	—
transferFrom	throw	throw

- **assert** или **require**?
  в зависимости от реализации
- если средства не переведены возвращается **false**
  и ваш код должен уметь с этим работать

Events

	Transfer	**Approval**
transfer	**event Transfer**(**address** indexed from, **address** indexed to, **uint256** value)	
transferFrom		
approve		**event Approval**(**address** _spender, **uint256** _value)
эмиссия (метод не специфицирован)	**event Transfer**(**0x0**, **address** indexed to, **uint256** value)	

Security alert!

Explained

- Разновидность **Frontrunning attack**

- **Начальные условия**
  allowance(0xOWNER, 0xSPENDER) = x, x > 0

- *0xOWNER* хочет изменить *x* на *y (y > 0)* и вызывает метод *approve(0xSPENDER, y)*

- **Атака**
  0xSPENDER отправляет транзакцию *transferFrom(0xOWNER, 0xSPENDER, x)* с высоким *gas price*, что бы майнеры включили её в блок раньше *approve* от *0xOWNER.*

  В результате *0xSPENDER* получает *x* токенов и возможность управлять еще *y* токенами.

Solution

- Методы *increaseAllowance* / *decreaseAllowance* – изменяют allowance, не задавая фиксированное значение

- Всегда делать *approve(…, 0)* перед изменение *approve* может возвращать *false* если предыдущий шаг не выполнен
  Увеличивает количество транзакций: долго и дорого

- *ERC20* широко используется, поэтому отказаться от approve нельзя из-за обратной совместимости :(

Design issues

Потерянные токены

- Нельзя отказаться от перевода токенов
- Можно перевести токены на счет смарт-контракта который не умеет ими управлять
- Распространенный кейс: перевод токенов на счет биржи

Перевод токенов ERC-20 ни к чему не ведет

- Перевод токенов — 1 транзакция

- Использование токенов — 2 транзакции (сначала *approve*)

¬ERC-20

- ERC-23 / ERC-223 token fallback
- ERC-621 изменение totalSupply
- ERC-721 учет невзаимозаменяемых ценностей
- ERC-827 …andCall функции

ERC-23/223

- https://github.com/aragon/ERC23
- https://github.com/ethereum/EIPs/issues/223
- ERC223 is ERC20

token fallback

- *function transfer*
  (address _to, uint _value, bytes _data)
  returns (bool)

- Можно отправлять сообщения вместе с токенами

- Если *to* смарт-контракт
  то внутри *transfer* должна вызывать функцию
  tokenFallback(address, uint256, bytes)

- Если *tokenFallback* не возвращает *true*, то откатываем транзакцию

ERC-621

- https://github.com/ethereum/EIPs/pull/621

- increaseSupply(uint value, address to)
  decreaseSupply(uint value, address from)

- Сейчас используются emit/burn безо всякого стандарта

ERC-721

- https://github.com/ethereum/EIPs/blob/master/EIPS/eip-721.md
- https://github.com/ethereum/EIPs/pull/841
- У токенов есть идентификаторы
- safeTransfer методы с подтверждением

ERC-827

- https://github.com/ethereum/EIPs/issues/827

- function transferAndCall(
    address _to, uint256 _value, bytes _data
) public payable returns (bool)

- Внутри *require(_to.call(_data));*

- Аналогично: *transferFromAndCall*, *approveAndCall*

- *to ≠ this*

Ether Token

- 1 токен = 1 Ether
- *deposit payable*: начисляет токены в объеме msg.value
- *withdraw*: сжигает токены и возвращает эфир