Владимир Рачкин и Павел Балай

Построение TLA⁺ модели для смарт-контракта Phoenix Vault

Проект по курсу математической логики, весна 2022

04.06.2022



Факультет математики и компьютерных наук СПбГУ Программа «Современное программирование»

Постановка задачи

- 1. Изучить смарт-контракт, описанный в статье¹
- 2. Построить его TLA+ модель и проверить свойства
- 3. Смоделировать специальные события: потери ключа и атаки злоумышленника

¹Phoenix: A Formally Verified Regenerating Vault (2021) за авторством Uri Kirstein, Shelly Grossman, Michael Mirkin, James Wilcox, Ittay Eyal, Mooly Sagiv



Что такое Phoenix Vault

Это обёртка для Etherium-кошелька

- 2 тира ключей: обычные для отправки средств, привилегированные для управления контрактом
- Возможность создавать новые ключи и удалять старые
- Задержка перед отправкой средств
- Возможность отменить отправку средств
- Возможность заблокировать отправку средств



Модель

Action	Key
Deposit	
Request	T_2
Withdraw	
Cancel request	T_1
Cancel all requests	T_1
Cancel self request	T_2
Lock	T_1
Add a T_1 key	T_1
Add a T_2 key	T_1
Remove a T_2 key	T_1

Events:

Tier-1 Key Loss

Tier-2 Key Loss

Type-1 Attack

Type-2 Attack



Состояния модели

- balance
- block_number
- tier_one_addresses
- tier_two_addresses
- delay
- unlock_block
- requests



Хранение предыдущего действия

```
Lock(address1, new unlock block) ==
  ∧ previous command' = (<<"lock", address1>>)

∧ unlock block' = new unlock block

  \(\text{UNCHANGED} \) < balance, tier one addresses, tier two addresses, delay,
   requests, special vars>>
OnlyTierOneCanLock ==
  [][previous command'[1] = "lock" => previous command'[2] \in
   tier one addresses] previous command
```



• /



- A
- ¬A



- A
- ¬A
- □(¬A)



- A
- ¬A
- $\Box(\neg A)$
- $\neg \Box (\neg A)$



- A
- ¬A
- $\Box(\neg A)$
- $\neg \Box (\neg A)$
- $B \Rightarrow \neg \Box (\neg A)$

- A
- ¬A
- $\Box(\neg A)$
- $\bullet \neg \Box (\neg A)$
- $B \Rightarrow \neg \Box (\neg A)$
- $\Box(B \Rightarrow \neg \Box(\neg A))$

```
TierOneCanCancelAnyRequestAnyTime ==
[]((requests /= {} ∧ block number < MAX BLOCK NUMBER) =>
     LET b == block number IN
       A r \in \text{request type}
          r \in requests =>
         (~[](~(
            \land block number = b + 1
            ∧ previous command[1] = "cancel request"

∧ previous command[2] \in tier one addresses

            \land previous command[3] = r[1])))))
```



Использование 'ENABLED'

```
TierOneCanCancelAnyRequestAnyTime ==

[](block_number < MAX_BLOCK_NUMBER

=> \A <<address1, req>> \in tier_one_addresses \X requests:

ENABLED CancelRequest(address1, req[1]))
```



Изменение модели

Добавили 2 множества ключей о которых знает владелец и злоумышленник

Добавили в Actions запуск событий и разделили действия владельца и злоумышленника

Добавили свойства, гарантирующие обработку всех событий



Безопасность

```
Defence ==
```

Next ==

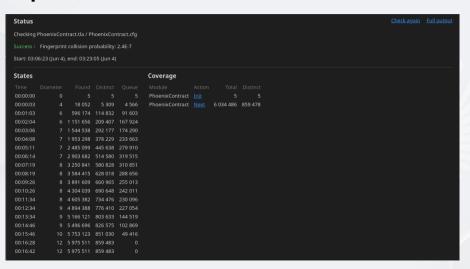
IF ENABLED Defence

THEN Defence

ELSE Actions ∨ ActionTick



Проверка модели





Результаты работы

- 1. Построена и проверена модель контракта
- 2. Смоделированы атаки и потери ключей

Владимир Рачкин @robozmey
Павел Балай @Koropok
Ссылка на Github проекта phoenix_proof



