HASHLAB

Smart Contract Internal Audit: Even Token

Abstract

In this report we consider the security of the Even Token. Our task is to find and describe any security issues in the Smart Contracts.

Analysis technique

We used several publicly available automated Solidity analysis tools, as well as proceed manual analysis. All of the issues found by the tools were manually checked (rejected or confirmed). Contracts were manually analyzed.

Bugs classification:

CRITICAL - problems leading to stealing funds from any of the participants, or making them inaccessible by anyone
SEVERE - problems that can stop, freeze or break the internal logic of the contract
WARNING - non-critical problems that cannot break the contract
Notes - any other findings

HASHLAB

Smart Contract Internal Audit: Even Token

Description of smart contracts for EvenToken:

1. IER20 – defines the interface that all ERC20 token implementations should conform to.

2. ERC20 – the base implementation of the ERC20 interface.

3. ERC20Detailed – the name(), symbol(), and decimals() getters are optional in the original standard, so ERC20Detailed adds that information to your token.

4. ERC20Mintable – ERC20Mintable allows users with the MinterRole to call the mint() function and mint tokens to users. Minting can also be finished, locking the mint() function's behavior.

5.ERC20Capped – ERC20Capped is a type of ERC20Mintable that enforces a maximum cap on tokens; this is really useful if you want to ensure network participants that there will always be a maximum number of tokens, and is useful for making sure that multiple different minting methods don't accidentally create more tokens than you expected.

6. Roles contracts - a MintableToken could have a minter role that decides who can mint tokens (which could be assigned to a Crowdsale). It could also have a namer role that allows changing the name or symbol of the token (for whatever reason). RBAC gives you much more flexibility over who can do what and is generally recommended for applications that need more configurability. If you're experienced with web development, the vast majority of access control systems are role-based: some users are normal users, some are moderators, and some can be company employee admins.

HASHLAB

Smart Contract Internal Audit: Even Token

Notes:

Use latest Solidity compiler version for deploy

— Description

It is recommended to use the latest version of solidity compiler. It helps to avoid some bugs of previous version compiler.

— Solution

Compile smart contracts using latest solidity compiler version (solidity 0.4.25).

Vulnerability checking

We have checked smart contracts for vulnerabilities described above. None of our checks showed vulnerability.

We have scanned contract for common and several specific vulnerabilities.

Here is a part of them:

— Reentrancy: not found
— Over and under flows: not found
— Replay attack: not found
— Timestamp Dependence: not found
— Gas Limit: not found
— DoS with (Unexpected) Throw: not found
— Transaction-Ordering Dependence: not found
— Exception disorder: not found
— Gasless send: not found

HΔSHLΔB

Smart Contract Internal Audit: Even Token

Test report of all EvenToken smart contracts:

```
  Contract: SafeMath
    add
      ✓ adds correctly
      ✓ throws a revert error on addition overflow

    sub
      ✓ subtracts correctly (80ms)
      ✓ throws a revert error if subtraction result
would be negative

    mul
      ✓ multiplies correctly
      ✓ handles a zero product correctly
      ✓ throws a revert error on multiplication over-
flow

    div
      ✓ divides correctly
      ✓ throws a revert error on zero division

    mod
      ✓ reverts with a 0 divisor
      modulos correctly
      ✓ when the dividend is smaller than the divisor
      ✓ when the dividend is equal to the divisor(72ms)
      ✓ when the dividend is larger than the divisor
      ✓ when the dividend is a multiple of the divisor
```

HΔSHLΔB

Smart Contract Internal Audit: Even Token

Contract: Roles

✓ reverts when querying roles for the null account initially

✓ doesn't pre-assign roles (78ms)

adding roles

✓ adds roles to a single account (116ms)

✓ adds roles to an already-assigned account (102ms)

✓ reverts when adding roles to the null account with added roles

removing roles

✓ removes a single role (95ms)

✓ doesn't revert when removing unassigned roles

✓ reverts when removing roles from the null account

Smart Contract Internal Audit: Even Token

Contract: MinterRole

```
     should behave like public role
       ✓ reverts when querying roles for the null ac-
count access control

       from authorized account
          ✓ allows access

       from unauthorized account
          ✓ reverts

       add
          ✓ adds role to a new account (63ms)
          ✓ emits a MinterAdded event (45ms)
          ✓ adds role to an already-assigned account
(60ms)
          ✓ reverts when adding role to the null account

       remove
           ✓ removes role from an already assigned ac-
count (74ms)
           ✓ emits a MinterRemoved event (41ms)
           ✓ doesn't revert when removing from an unas-
signed account
           ✓ reverts when removing role from the null
account

       renouncing roles
           ✓ renounces an assigned role (60ms)
           ✓ emits a MinterRemoved event (40ms)
           ✓ doesn't revert when renouncing unassigned
role (40ms)
```

HΔSHLΔB

Smart Contract Internal Audit: Even Token

Contract: ERC20

```
      total supply
        ✓ returns the total amount of tokens
      balanceOf
        when the requested account has no tokens
          ✓ returns zero
        when the requested account has some tokens
          ✓ returns the total amount of tokens
      transfer
        when the recipient is not the zero address
          when the sender does not have enough balance
            ✓ reverts
          when the sender has enough balance
            ✓ transfers the requested amount (97ms)
            ✓ emits a transfer event (44ms)
        when the recipient is the zero address
          ✓ reverts
      approve
        when the spender is not the zero address
          when the sender has enough balance
            ✓ emits an approval event (40ms)
            when there was no approved amount before
              ✓ approves the requested amount (68ms)
            when the spender had an approved amount
              ✓ approves the requested amount and re-
places the previous one (56ms)
          when the sender does not have enough balance
            ✓ emits an approval event (83ms)
            when there was no approved amount before
              ✓ approves the requested amount (63ms)
            when the spender had an approved amount
              ✓ approves the requested amount and re-
places the previous one (69ms)
        when the spender is the zero address
          ✓ reverts
      transfer from
```

Smart Contract Internal Audit: Even Token

```
  when the recipient is not the zero address
          when the spender has enough approved balance
            when the owner has enough balance
              ✓ transfers the requested amount (88ms)
              ✓ decreases the spender allowance (73ms)
              ✓ emits a transfer event (63ms)
            when the owner does not have enough balance
              ✓ reverts
          when the spender does not have enough ap-
proved balance
            when the owner has enough balance
              ✓ reverts
            when the owner does not have enough balance
              ✓ reverts
        when the recipient is the zero address
          ✓ reverts
      decrease allowance
        when the spender is not the zero address
          when the sender has enough balance
            when there was no approved amount before
              ✓ reverts
            when the spender had an approved amount
              ✓ emits an approval event (60ms)
              ✓ decreases the spender allowance sub-
tracting the requested amount (78ms)
              ✓ sets the allowance to zero when all al-
lowance is removed (70ms)
              ✓ reverts when more than the full al-
lowance is removed
          when the sender does not have enough balance
            when there was no approved amount before
              ✓ reverts
            when the spender had an approved amount
              ✓ emits an approval event (50ms)
              ✓ decreases the spender allowance sub-
tracting the requested amount (67ms)
              ✓ sets the allowance to zero when all al-
```

Smart Contract Internal Audit: Even Token

```
lowance is removed (65ms)

    ✓ reverts when more than the full allowance is re-
moved
        when the spender is the zero address
          ✓ reverts
      increase allowance
        when the spender is not the zero address
          when the sender has enough balance
            ✓ emits an approval event (53ms)
            when there was no approved amount before
              ✓ approves the requested amount (68ms)
            when the spender had an approved amount
              ✓ increases the spender allowance adding
the requested amount (80ms)
          when the sender does not have enough balance
            ✓ emits an approval event (44ms)
            when there was no approved amount before
              ✓ approves the requested amount (69ms)
            when the spender had an approved amount
              ✓ increases the spender allowance adding
the requested amount (77ms)
        when the spender is the zero address
          ✓ reverts
      _mint
        ✓ rejects a null account
        for a non null account
          ✓ increments totalSupply
          ✓ increments recipient balance
          ✓ emits Transfer event
      _burn
        ✓ rejects a null account
        for a non null account
          ✓ rejects burning more than balance
          for entire balance
            ✓ decrements totalSupply
            ✓ decrements owner balance (43ms)
            ✓ emits Transfer event
```

Smart Contract Internal Audit: Even Token

```
        for less amount than balance
          ✓ decrements totalSupply

✓ decrements owner balance (40ms)
          ✓ emits Transfer event
    _burnFrom
      ✓ rejects a null account (74ms)
      for a non null account
        ✓ rejects burning more than allowance (63ms)
        ✓ rejects burning more than balance
        for entire allowance
          ✓ decrements totalSupply
          ✓ decrements owner balance
          ✓ decrements spender allowance
```

Smart Contract Internal Audit: Even Token

Contract: ERC20Detailed
    ✓ has a name
    ✓ has a symbol
    ✓ has an amount of decimals

Contract: ERC20Mintable
      minter role
        should behave like public role
          ✓ reverts when querying roles for the null account
          access control
            from authorized account
              ✓ allows access
            from unauthorized account
              ✓ reverts
          add
            ✓ adds role to a new account (66ms)
            ✓ emits a MinterAdded event
            ✓ adds role to an already-assigned account (80ms)
            ✓ reverts when adding role to the null account
          remove
            ✓ removes role from an already assigned account (80ms)

  ✓ emits a MinterRemoved event
            ✓ doesn't revert when removing from an unassigned account (38ms)
            ✓ reverts when removing role from the null account
          renouncing roles
            ✓ renounces an assigned role (52ms)
            ✓ emits a MinterRemoved event
            ✓ doesn't revert when renouncing unassigned role (42ms)

HASHLAB

Smart Contract Internal Audit: Even Token

```
      as a mintable token
        mint
          when the sender has minting permission
            for a zero amount
              ✓ mints the requested amount
              ✓ emits a mint and a transfer event
            for a non-zero amount
              ✓ mints the requested amount
              ✓ emits a mint and a transfer event
          when the sender doesn't have minting permis-
sion
            ✓ reverts (55ms)


 Contract: ERC20Capped
    ✓ requires a non-zero cap (57ms)
    once deployed
      capped token
        ✓ should start with the correct cap
        ✓ should mint when amount is less than cap
(64ms)
        ✓ should fail to mint if the ammount exceeds
the cap (77ms)
        ✓ should fail to mint after cap is reached
(86ms)
      as a mintable token
        mint
          when the sender has minting permission
            for a zero amount
              ✓ mints the requested amount
              ✓ emits a mint and a transfer event
            for a non-zero amount
              ✓ mints the requested amount
              ✓ emits a mint and a transfer event
          when the sender doesn't have minting permis-
sion
            ✓ reverts (74ms)
```

HΔSHLΔB

Smart Contract Internal Audit: Even Token

Conclusion

After complete manual/automated analysis and testing of smart contracts and vulnerability checking we conclude that this contract has EXTRA high code quality and security.