SmartDec

# Soundeon Smart Contracts Security Analysis

This report is public.

Published: June 13, 2018

# Abstract

In this report, we consider the security of the Soundeon project. Our task is to find and describe security issues in the smart contracts of the platform.

# Disclaimer

The audit does not give any warranties on the security of the code. One audit cannot be considered enough. We always recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts. Besides, security audit is not an investment advice.

# Summary

In this report, we have considered the security of Soundeon smart contracts. We performed our audit according to the procedure described below.

The initial audit has shown no critical issues. All of the medium severity issues and low severity issues that could possibly pose a threat were fixed in the latest version of the code. The rest of issues do not endanger project security. Thus, the latest version of the code is ready for the release.

# General recommendations

The latest version of the code does not contain any serious issues. However, if the developer decides to improve the code, we highly recommend addressing low severity issues, i.e. removing Redundant code and improving Code coverage level.

The text below is for technical use; it details the statements made in Summary and General recommendations.

# Procedure

In our audit, we consider the following crucial features of the smart contract code:

1. Whether the code is secure.
2. Whether the code corresponds to the documentation (including whitepaper).
3. Whether the code meets best practices in efficient use of gas, code readability, etc.

We perform our audit according to the following procedure:

- automated analysis
    - we scan project's smart contracts with our own Solidity static code analyzer SmartCheck
    - we scan project's smart contracts with several publicly available automated Solidity analysis tools such as Remix, Oyente, and Solhint
    - we manually verify (reject or confirm) all the issues found by tools
- manual audit
    - we manually analyze smart contracts for security vulnerabilities
    - we check smart contracts logic and compare it with the one described in the whitepaper
    - we check ERC20 compliance
    - we run tests and check code coverage
- report
    - we report all the issues found to the developer during the audit process
    - we check the issues fixed by the developer
    - we reflect all the gathered information in the report

# Checked vulnerabilities

We have scanned Soundeon smart contracts for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that we considered (the full list includes them but is not limited to them):

- Reentrancy
- Timestamp Dependence
- Gas Limit and Loops
- DoS with (Unexpected) Throw
- DoS with (Unexpected) revert
- DoS with Block Gas Limit
- Transaction-Ordering Dependence
- Use of tx.origin
- Exception disorder
- Gasless send
- Balance equality
- Byte array
- Transfer forwards all gas
- ERC20 API violation
- Malicious libraries
- Compiler version not fixed
- Redundant fallback function
- Send instead of transfer
- Style guide violation
- Unchecked external call
- Unchecked math
- Unsafe type inference
- Implicit visibility level
- Address hardcoded
- Using delete for arrays
- Integer overflow/underflow
- Locked money
- Private modifier
- Revert/require functions
- Using var
- Visibility
- Using blockhash
- Using SHA3
- Using suicide
- Using throw
- Using inline assembly

# Project overview

## Project description

In our analysis, we consider Soundeon whitepaper (“Soundeon_WP.pdf”, sha1sum 0925173ce58e7f9851aaec5324edf31d853be239) and smart contracts code (“ico-master-ce31e945b382fc16b42953d978d22a1a8355e13b.zip”, sha1sum f91f9b2c3d41cfaae4263766bb71b9f9cb24d523).

### The latest version of the code

We have performed the check of the fixed vulnerabilities in the latest version of the code (Git repository, version on commit 8994de550270c8377b82a67a0567b85dd9776a9b).

## Project architecture

For the audit, we have been provided with the following set of files:

- SoundeonToken.sol
- SoundeonTokenDistributor.sol
- SoundeonTokenMinter.sol
- SoundeonTokenSender.sol

Files successfully compile with `solc` command (with some warnings, see Compilation output in Appendix) and successfully pass all the tests (see Tests output in Appendix). However, code coverage is not 100% (see Code coverage in Appendix).
*In the latest version of the code, the files were combined in the complete Truffle project, npm was used. All the outputs are performed for the latest version of the code.*

Files contain the following contracts:

- SoundeonToken (inherits StandardBurnableToken, CappedToken, DetailedERC20, PausableToken contracts from the modified version of OpenZeppelin library)
- SoundeonTokenDistributor (inherits Ownable contract the modified version of OpenZeppelin library)
- SoundeonTokenMinter (inherits SoundeonTokenDistributor contract)
- SoundeonTokenSender.sol (inherits SoundeonTokenDistributor contract)

The version of OpenZeppelin library used in these contracts is the original OpenZeppelin library (v1.9.0) with the following modifications:

- usual arithmetic operations were used instead of SafeMath functions, which leads to an overflow issue in MintableToken contract
- `whenNotPaused()` and `whenPaused()` modifiers in Pausable contract do not affect owner’s calls
- newer version (0.4.23 instead of 0.4.21) of compiler was used

The total volume of audited files is 79 lines of Solidity code.

## Code logic

**SoundeonToken** is ERC20 compatible (compatibility has been checked during the audit) token contract with the following parameters:

- token name: “Soundeon Token”
- token symbol: “Soundeon”
- token decimals: 18

Besides, some additional functionality was implemented:

1. The contract inherits Ownable contract. This means that the contract has an owner (initially, it is the address the contract was deployed from).
2. The contract includes `increaseApproval()` and `decreaseApproval()` functions. This makes changing the `allowance` mapping possible without using `approve()` function, which has a vulnerability (see ERC20 approve issue).
3. The contract inherits PausableToken contract. This means that the owner of the contract can pause and unpause token transfers and changes of `allowed` mapping for all the users except himself.
4. The contract inherits MintableToken contract. This means that the owner of the token contract can mint any amount of tokens to any address. There is an overflow possibility in `mint()` function in this contract.
5. The contract inherits CappedToken contract. This means that the total amount of tokens are bounded by 1 billion. This contract has an overflow check in `mint()` function, thus SoundeonToken contract has no vulnerability.
6. The contract inherits StandardBurnableToken. This means that a specific amount of tokens can be burned from the target address if it is allowed to `msg.sender`. Also, users can just burn their tokens using `burn()` function.

**SoundeonTokenDistributor** contract creates SoundeonToken in its constructor. It includes `processedTransactions` mapping and `isTransactionSuccessful()` function showing the value of this mapping for specified `id`.
Also, it includes `transferTokenOwnership()` function that transfers token ownership to the owner of the contract.
*In the latest version of the code, **SoundeonTokenDistributor** contract’s constructor has a parameter `_token`. In case the parameter has null value the constructor creates new token, otherwise it uses the given token address.*

**SoundeonTokenMinter** is SoundeonTokenDistributor contract with one additional function:

```
function bulkMint(uint32[] _payment_ids, address[] _receivers,
uint256[] _amounts)
```

- Call restrictions: only owner can call the function
- Parameters requirements: lengths of all the arrays should be equal

- Logic: mints the amount of tokens from `_amounts` array to every address from `_receivers` array; sets the value of `processedTransactions` mapping with the corresponding id from `_payment_ids` array to `true`

*In the latest version of the code, `bulkMint()` function also mints tokens to the hardcoded addresses according to the token distribution mentioned in the whitepaper.*

**SoundeonTokenSender** is SoundeonTokenDistributor contract with a number of additional functions:

```
function bulkTransfer(uint32[] _payment_ids, address[]
_receivers, uint256[] _amounts)
```

- Call restrictions: only owner can call the function
- Parameters requirements: lengths of all the arrays should be equal
- Logic: repeatedly transfers tokens using `_receivers` array for receivers, `_amounts` array for amounts of tokens, and `msg.sender` address for the sender; sets the value of `processedTransactions` mapping with the corresponding id from `_payment_ids` array to `true`

```
function bulkTransferFrom(uint32[] _payment_ids, address
_from, address[] _receivers, uint256[] _amounts)
```

- Call restrictions: only owner can call the function
- Parameters requirements:
  - lengths of all the arrays should be equal
  - `_from` address should allow enough tokens to be transferred to the addresses from `_receivers` array
- Logic: repeatedly transfers tokens using `_receivers` array for receivers, `_amounts` array for amounts of tokens, and `_from` address for the sender; sets the value of `processedTransactions` mapping with the corresponding id from `_payment_ids` array to `true`

```
function transferTokensToOwner()
```

- Call restrictions: only owner can call the function
- Logic: sends all the contract's tokens to the owner of the contract

# Automated analysis

We used several publicly available automated Solidity analysis tools. Here are the combined results of SmartCheck, Solhint, and Remix. Oyente has found no issues.
All the issues found by tools were manually checked (rejected or confirmed).

**False positives** are constructions that were discovered by the tools as vulnerabilities but do not consist a security threat.

**True positives** are constructions that were discovered by the tools as vulnerabilities and can actually be exploited by attackers or lead to incorrect contracts operation.

Cases when these issues lead to actual bugs or vulnerabilities are described in the next section.

| Tool | Rule | False positives | True positives |
|---|---|---|---|
| Remix | Gas requirement of function high | 6 | 3 |
| | Potential Violation of Checks-Effects-Interaction pattern | | 3 |
| | Potentially should be constant but is not | 6 | |
| | Variables have very similar names | 1 | |
| Total Remix | | 13 | 6 |
| SmartCheck | Gas Limit And Loops | | 3 |
| | No Payable Fallback | 4 | |
| | Pragmas Version | | 4 |
| | Reentrancy External Call | 3 | |
| | Unchecked Math | 1 | |

| | | | |
|---|---|---|---|
| Total SmartCheck | | 8 | 7 |
| Solhint | Compiler version must be fixed | | 4 |
| Total Solhint | | | 4 |
| **Total Overall** | | **21** | **17** |

# Manual analysis

The contracts were completely manually analyzed, their logic was checked and compared with the one described in the documentation. Besides, the results of the automated analysis were manually verified. All confirmed issues are described below.

## Critical issues

Critical issues seriously endanger smart contracts security. We highly recommend fixing them.

**The audit has shown no critical issues.**

## Medium severity issues

Medium issues can influence smart contracts operation in current implementation. We highly recommend addressing them.

### Discrepancy with the whitepaper

There is a discrepancy between the smart contracts code and the whitepaper:

A lot of crucial crowdsale information is listed in the whitepaper (such as USD rate, Hard Cap, Soft Cap, etc.). However, there is no Crowdsale contract in the project.
*The developer has explained that crowdsale will be held off-chain.*

### Modified standard library/overflow possibility

As it was mentioned above, the audited contracts use OpenZeppelin library with some modifications and these modifications lead to vulnerabilities.

There is a possibility of overflow at CappedToken.sol, line 27:

```
require(totalSupply_ + _amount <= cap);
```

This check makes it possible for the owner of the token to mint large amounts (like 2**256-1) of tokens. Thus, the cap can be exceeded or user's balance can be made equal to 0. In addition, after minting such amounts of tokens, `totalSupply` variable will become incorrect.

We recommend implementing additional check in this case.
*In the latest version of the code, this vulnerability was fixed by the developer.*

## Low severity issues

Low severity issues can influence smart contracts operation in future versions of code. We recommend taking them into account.

### Pragmas version

Solidity source files indicate the versions of the compiler they can be compiled with.

Example:

```
pragma solidity ^0.4.18; // bad: compiles w 0.4.18 and above
pragma solidity 0.4.18; // good: compiles w 0.4.18 only
```

We recommend following the latter example, as future compiler versions may handle certain language constructions in a way the developer did not foresee. Besides, we recommend using the latest compiler version – 0.4.24 at the moment.
*The developer has explained that the recommended version of the compiler will be used for deploy.*

### Redundant code

`isTransactionSuccessful()` function is redundant as `processedTransactions` mapping has `public` modifier and an automatic getter function is generated.

We highly recommend removing redundant code in order to improve code readability and decrease cost of deployment.

### Insufficient code coverage level

The contracts included in the project are not fully covered with tests (see Code coverage in Appendix). Testing is crucial for code security. We highly recommend not only covering the code with tests but also making sure that the coverage is sufficient.

# Notes

## Costly loops

The contracts include traversing through an array of an arbitrary length in `bulkMint()`, `bulkTransfer()`, and `bulkTransferFrom()` functions:

```
for (uint i = 0; i < _receivers.length; i++) { … }
```

The `_receivers` array is passed as the parameter. Therefore, if there are too many items in the `_receivers` array, the execution of these functions will fail due to an out-of-gas exception.

In this case, we recommend separating the call into several transactions.

## ERC20 approve issue

There is ERC20 approve issue: changing the approved amount from a nonzero value to another nonzero value allows a double spending with a front-running attack.

We recommend instructing users to follow one of two ways:

- not to use `approve()` function directly and to use `increaseApproval()/decreaseApproval()` functions instead
- to change the approved amount to 0, wait for the transaction to be mined, and then to change the approved amount to the desired value

This analysis was performed by SmartDec.

Katerina Troshina, Chief Executive Officer
Yaroslav Alexandrov, Head of Development Department
Alexander Seleznev, Chief Business Development Officer
Ivan Ivanitskiy, Chief Analytics Officer
Igor Sobolev, Analyst

Sergey Pavlin, Chief Operating Officer 

June 13, 2018

# Appendix

## Code coverage

```
--------------------------------|----------|----------|----------|----------|----------------|
File                            |  % Stmts | % Branch |  % Funcs |  % Lines |Uncovered Lines |
--------------------------------|----------|----------|----------|----------|----------------|
 contracts/                     |    86.36 |     62.5 |    81.82 |    82.14 |                |
  SoundeonToken.sol             |      100 |      100 |      100 |      100 |                |
  SoundeonTokenDistributor.sol  |      100 |      100 |      100 |      100 |                |
  SoundeonTokenMinter.sol       |      100 |     62.5 |      100 |      100 |                |
  SoundeonTokenSender.sol       |       50 |       25 |       50 |    44.44 | 24,25,26,28,34 |
 contracts/base/                |      100 |    90.38 |      100 |      100 |                |
  BasicToken.sol                |      100 |      100 |      100 |      100 |                |
  BurnableToken.sol             |      100 |      100 |      100 |      100 |                |
  CappedToken.sol               |      100 |    66.67 |      100 |      100 |                |
  DetailedERC20.sol             |      100 |      100 |      100 |      100 |                |
  ERC20.sol                     |      100 |      100 |      100 |      100 |                |
  ERC20Basic.sol                |      100 |      100 |      100 |      100 |                |
  MintableToken.sol             |      100 |      100 |      100 |      100 |                |
  PausableToken.sol             |      100 |      100 |      100 |      100 |                |
  SafeERC20.sol                 |      100 |      100 |      100 |      100 |                |
  StandardBurnableToken.sol     |      100 |      100 |      100 |      100 |                |
  StandardToken.sol             |      100 |      100 |      100 |      100 |                |
  TokenTimelock.sol             |      100 |    83.33 |      100 |      100 |                |
  TokenVesting.sol              |      100 |    85.71 |      100 |      100 |                |
 contracts/lifecycle/           |      100 |       75 |      100 |      100 |                |
  Pausable.sol                  |      100 |       75 |      100 |      100 |                |
 contracts/mocks/               |    65.38 |      100 |    67.86 |     69.7 |                |
  BasicTokenMock.sol            |      100 |      100 |      100 |      100 |                |
  BurnableTokenMock.sol         |      100 |      100 |      100 |      100 |                |
  DetailedERC20Mock.sol         |      100 |      100 |      100 |      100 |                |
  PausableMock.sol              |        0 |      100 |        0 |        0 |    13,14,18,22 |
  PausableTokenMock.sol         |      100 |      100 |      100 |      100 |                |
  SafeERC20Helper.sol           |    57.14 |      100 |    68.42 |       70 |... 29,36,52,56 |
  StandardBurnableTokenMock.sol |      100 |      100 |      100 |      100 |                |
  StandardTokenMock.sol         |      100 |      100 |      100 |      100 |                |
 contracts/ownership/           |      100 |       75 |      100 |      100 |                |
  Ownable.sol                   |      100 |       75 |      100 |      100 |                |
--------------------------------|----------|----------|----------|----------|----------------|
All files                       |    92.36 |    80.95 |     85.9 |    91.48 |                |
--------------------------------|----------|----------|----------|----------|----------------|
```

## Compilation output

```
$ truffle compile
Compiling .\contracts\Migrations.sol...
Compiling .\contracts\SoundeonToken.sol...
Compiling .\contracts\SoundeonTokenDistributor.sol...
Compiling .\contracts\SoundeonTokenMinter.sol...
Compiling .\contracts\SoundeonTokenSender.sol...
Compiling .\contracts\base\BasicToken.sol...
```

```
Compiling .\contracts\base\BurnableToken.sol...
Compiling .\contracts\base\CappedToken.sol...
Compiling .\contracts\base\DetailedERC20.sol...
Compiling .\contracts\base\ERC20.sol...
Compiling .\contracts\base\ERC20Basic.sol...
Compiling .\contracts\base\MintableToken.sol...
Compiling .\contracts\base\PausableToken.sol...
Compiling .\contracts\base\SafeERC20.sol...
Compiling .\contracts\base\StandardBurnableToken.sol...
Compiling .\contracts\base\StandardToken.sol...
Compiling .\contracts\base\TokenTimelock.sol...
Compiling .\contracts\base\TokenVesting.sol...
Compiling .\contracts\lifecycle\Pausable.sol...
Compiling .\contracts\mocks\BasicTokenMock.sol...
Compiling .\contracts\mocks\BurnableTokenMock.sol...
Compiling .\contracts\mocks\DetailedERC20Mock.sol...
Compiling .\contracts\mocks\PausableMock.sol...
Compiling .\contracts\mocks\PausableTokenMock.sol...
Compiling .\contracts\mocks\SafeERC20Helper.sol...
Compiling .\contracts\mocks\StandardBurnableTokenMock.sol...
Compiling .\contracts\mocks\StandardTokenMock.sol...
Compiling .\contracts\ownership\Ownable.sol...

Compilation warnings encountered:

./contracts/base/TokenTimelock.sol:23:5: Warning: Defining
constructors as functions with the same name as the contract
is deprecated. Use "constructor(...) { ... }" instead.
    function TokenTimelock(ERC20Basic _token, address
_beneficiary, uint256 _releaseTime) public {
    ^ (Relevant source part starts here and spans across
multiple lines).
,./contracts/base/TokenVesting.sol:43:5: Warning: Defining
constructors as functions with the same name as the contract
is deprecated. Use "constructor(...) { ... }" instead.
    function TokenVesting(
    ^ (Relevant source part starts here and spans across
multiple lines).
,./contracts/mocks/BasicTokenMock.sol:9:5: Warning: Defining
constructors as functions with the same name as the contract
is deprecated. Use "constructor(...) { ... }" instead.
    function BasicTokenMock(address initialAccount, uint256
initialBalance) public {
    ^ (Relevant source part starts here and spans across
multiple lines).
,./contracts/mocks/BurnableTokenMock.sol:8:5: Warning:
Defining constructors as functions with the same name as the
```

```
contract is deprecated. Use "constructor(...) { ... }"
instead.
    function BurnableTokenMock(address initialAccount, uint
initialBalance) public {
    ^ (Relevant source part starts here and spans across
multiple lines).
,./contracts/mocks/DetailedERC20Mock.sol:8:5: Warning:
Defining constructors as functions with the same name as the
contract is deprecated. Use "constructor(...) { ... }"
instead.
    function DetailedERC20Mock(string _name, string _symbol,
uint8 _decimals) DetailedERC20(_name, _symbol, _decimals)
public {}
    ^----------------------------------------------------------
----------------------------------------------------------
---^
,./contracts/mocks/PausableMock.sol:12:5: Warning: Defining
constructors as functions with the same name as the contract
is deprecated. Use "constructor(...) { ... }" instead.
    function PausableMock() public {
    ^ (Relevant source part starts here and spans across
multiple lines).
,./contracts/mocks/PausableTokenMock.sol:9:5: Warning:
Defining constructors as functions with the same name as the
contract is deprecated. Use "constructor(...) { ... }"
instead.
    function PausableTokenMock(address initialAccount, uint
initialBalance) public {
    ^ (Relevant source part starts here and spans across
multiple lines).
,./contracts/mocks/SafeERC20Helper.sol:67:5: Warning: Defining
constructors as functions with the same name as the contract
is deprecated. Use "constructor(...) { ... }" instead.
    function SafeERC20Helper() public {
    ^ (Relevant source part starts here and spans across
multiple lines).
,./contracts/mocks/StandardBurnableTokenMock.sol:8:5: Warning:
Defining constructors as functions with the same name as the
contract is deprecated. Use "constructor(...) { ... }"
instead.
    function StandardBurnableTokenMock(address initialAccount,
uint initialBalance) public {
    ^ (Relevant source part starts here and spans across
multiple lines).
,./contracts/mocks/StandardTokenMock.sol:9:5: Warning:
Defining constructors as functions with the same name as the
```

```
contract is deprecated. Use "constructor(...) { ... }"
instead.
    function StandardTokenMock(address initialAccount, uint256
initialBalance) public {
    ^ (Relevant source part starts here and spans across
multiple lines).

Writing artifacts to .\build\contracts
```

## Tests output

```
 Contract: SoundeonToken
    ✓ should have predefined name, symbol, maximup cap and
decimails (196ms)

  Contract: SoundeonTokenDistributor
    transfer token ownership
      ✓ should return ownership (101ms)
      ✓ should fail to return ownership when called by not an
owner (86ms)

  Contract: SoundeonTokenMinter
    when called not by an owner
      ✓ should fail to mint (126ms)
      ✓ should fail to return ownership
    when created
      ✓ should set pool addresses properly (141ms)
    when owning a token
      mint
        ✓ should mint corresponding amount of tokens (262ms)
        ✓ should mint tokens for each pool proportionally
(351ms)
        ✓ should fail to mint when called by not an owner
(111ms)
        ✓ should check if all arrays contains the same number
of items (179ms)
        ✓ should check if receivers are zero addresses (46ms)
        ✓ should fail, do not mint, nor record transaction as
successful when called to mint more than cap (326ms)
        ✓ should be able to mint cap number of tokens in one
transaction (317ms)
        ✓ should record processed transactions (295ms)
```

```
        should not process transaction with known id
          ✓ when it supplied in different batches (337ms)
          ✓ when it supplied in the batch (176ms)

  Contract: SoundeonTokenSender
    when called not by an owner
      ✓ should fail to transfer (118ms)
      ✓ should fail to return tokens to owner (78ms)
    when called by an owner
      transfer
        ✓ should transfer corresponding amount of tokens
(276ms)
        ✓ should check if all arrays contains the same number
of items (264ms)
        ✓ should check if receivers are zero addresses (160ms)
        ✓ should fail when called to transfer more than it
holds (178ms)

  Contract: StandardToken
    total supply
      ✓ returns the total amount of tokens (44ms)
    balanceOf
      when the requested account has no tokens
        ✓ returns zero (45ms)
      when the requested account has some tokens
        ✓ returns the total amount of tokens (42ms)
    transfer
      when the recipient is not the zero address
        when the sender does not have enough balance
          ✓ reverts (47ms)
        when the sender has enough balance
          ✓ transfers the requested amount (139ms)
          ✓ emits a transfer event (82ms)
      when the recipient is the zero address
        ✓ reverts (57ms)

  Contract: BurnableToken
    as a basic burnable token
      when the given amount is not greater than balance of the
sender
        ✓ burns the requested amount
        ✓ emits a burn event
        ✓ emits a transfer event
```

```
      when the given amount is greater than the balance of the
sender
        ✓ reverts (72ms)

  Contract: Capped
    ✓ should start with the correct cap (75ms)
    ✓ should mint when amount is less than cap (79ms)
    ✓ should fail to mint if the ammount exceeds the cap
(131ms)
    ✓ should fail to mint after cap is reached (113ms)

  Contract: DetailedERC20
    ✓ has a name (56ms)
    ✓ has a symbol (72ms)
    ✓ has an amount of decimals

  Contract: Mintable
    minting finished
      when the token is not finished
        ✓ returns false (54ms)
      when the token is finished
        ✓ returns true
    finish minting
      when the sender is the token owner
        when the token was not finished
          ✓ finishes token minting (82ms)
          ✓ emits a mint finished event (55ms)
        when the token was already finished
          ✓ reverts
      when the sender is not the token owner
        when the token was not finished
          ✓ reverts (40ms)
        when the token was already finished
          ✓ reverts
    mint
      when the sender is the token owner
        when the token was not finished
          ✓ mints the requested amount (113ms)
          ✓ emits a mint finished event (59ms)
        when the token minting is finished
          ✓ reverts (63ms)
      when the sender is not the token owner
        when the token was not finished
```

```
          ✓ reverts (54ms)
        when the token was already finished
          ✓ reverts (77ms)

  Contract: PausableToken
    pause
      when the sender is the token owner
        when the token is unpaused
          ✓ pauses the token (213ms)
          ✓ emits a Pause event (56ms)
      when the sender is not the token owner
        ✓ reverts (44ms)
    unpause
      when the sender is the token owner
        when the token is paused
          ✓ unpauses the token (65ms)
          ✓ emits an Unpause event (53ms)
      when the sender is not the token owner
        ✓ reverts
    pausable token
      paused
        ✓ is not paused by default
        ✓ is paused after being paused (67ms)
        ✓ is not paused after being paused and then unpaused
(110ms)
      transfer
        ✓ allows to transfer when unpaused (133ms)
        ✓ allows to transfer when paused and then unpaused
(193ms)
      approve
        ✓ allows to approve when unpaused (70ms)
        ✓ allows to transfer when paused and then unpaused
(184ms)
      transfer from
        ✓ allows to transfer from when unpaused (92ms)
        ✓ allows to transfer when paused and then unpaused
(326ms)
        ✓ reverts when trying to transfer from when paused
(66ms)
      decrease approval
        ✓ allows to decrease approval when unpaused (59ms)
        ✓ allows to decrease approval when paused and then
unpaused (135ms)
```

```
      increase approval
        ✓ allows to increase approval when unpaused (59ms)
        ✓ allows to increase approval when paused and then
unpaused (155ms)

  Contract: SafeERC20
    ✓ should throw on failed transfer
    ✓ should throw on failed transferFrom
    ✓ should throw on failed approve (47ms)
    ✓ should not throw on succeeding transfer (59ms)
    ✓ should not throw on succeeding transferFrom
    ✓ should not throw on succeeding approve (55ms)

  Contract: StandardBurnableToken
    as a basic burnable token
      when the given amount is not greater than balance of the
sender
        ✓ burns the requested amount
        ✓ emits a burn event
        ✓ emits a transfer event
      when the given amount is greater than the balance of the
sender
        ✓ reverts
    burnFrom
      on success
        ✓ burns the requested amount
        ✓ decrements allowance
        ✓ emits a burn event
        ✓ emits a transfer event
      when the given amount is greater than the balance of the
sender
        ✓ reverts (93ms)
      when the given amount is greater than the allowance
        ✓ reverts (75ms)

  Contract: StandardToken
    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 (98ms)
          ✓ emits a transfer event
      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
          when there was no approved amount before
            ✓ approves the requested amount (50ms)
          when the spender had an approved amount
            ✓ approves the requested amount and replaces the
previous one (57ms)
        when the sender does not have enough balance
          ✓ emits an approval event
          when there was no approved amount before
            ✓ approves the requested amount (66ms)
          when the spender had an approved amount
            ✓ approves the requested amount and replaces the
previous one (66ms)
      when the spender is the zero address
        ✓ approves the requested amount (69ms)
        ✓ emits an approval event (41ms)
    transfer from
      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 (86ms)
            ✓ decreases the spender allowance (48ms)
            ✓ emits a transfer event (38ms)
          when the owner does not have enough balance
            ✓ reverts
        when the spender does not have enough approved 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 approval
      when the spender is not the zero address
        when the sender has enough balance
          ✓ emits an approval event (39ms)
          when there was no approved amount before
            ✓ keeps the allowance to zero (60ms)
          when the spender had an approved amount
            ✓ decreases the spender allowance subtracting the
requested amount (58ms)
        when the sender does not have enough balance
          ✓ emits an approval event (42ms)
          when there was no approved amount before
            ✓ keeps the allowance to zero (57ms)
          when the spender had an approved amount
            ✓ decreases the spender allowance subtracting the
requested amount (67ms)
      when the spender is the zero address
        ✓ decreases the requested amount (62ms)
        ✓ emits an approval event
    increase approval
      when the spender is not the zero address
        when the sender has enough balance
          ✓ emits an approval event (44ms)
          when there was no approved amount before
            ✓ approves the requested amount (61ms)
          when the spender had an approved amount
            ✓ increases the spender allowance adding the
requested amount (44ms)
        when the sender does not have enough balance
          ✓ emits an approval event (142ms)
          when there was no approved amount before
            ✓ approves the requested amount (63ms)
            ✓ should approve maximum theoretical amount (76ms)
            ✓ should not increase approval over maximum
theoretical amount (103ms)
          when the spender had an approved amount
            ✓ increases the spender allowance adding the
requested amount (52ms)
      when the spender is the zero address
        ✓ approves the requested amount (66ms)
        ✓ emits an approval event (41ms)
```

```
  Contract: TokenTimelock
    ✓ cannot be released before time limit
    ✓ cannot be released just before time limit (216ms)
    ✓ can be released just after limit (205ms)
    ✓ can be released after time limit (262ms)
    ✓ cannot be released twice (269ms)

  Contract: TokenVesting
    ✓ cannot be released before cliff
    ✓ can be released after cliff (295ms)
    ✓ should release proper amount after cliff (469ms)
    ✓ should linearly release tokens during vesting period
(1273ms)
    ✓ should have released all after end (275ms)
    ✓ should be revoked by owner if revocable is set (81ms)
    ✓ should fail to be revoked by owner if revocable not set
(87ms)
    ✓ should return the non-vested tokens when revoked by
owner (270ms)
    ✓ should keep the vested tokens when revoked by owner
(304ms)
    ✓ should fail to be revoked a second time (312ms)

  143 passing (38s)
```

## Solhint output

```
contracts/SoundeonToken.sol
   1:17   warning  Compiler version must be
fixed
 compiler-fixed
   9:93   error    Open bracket must be on same line. It must
be indented by other constructions by space  bracket-align
  10:30   error    Visibility modifier must be first in list
of modifiers                                  visibility-
modifier-order
  10:105  warning  Code contains empty
block
      no-empty-blocks

contracts/SoundeonTokenDistributor.sol
```

```
   1:17  warning  Compiler version must be
fixed
     compiler-fixed
   5:1   error    Definition must be surrounded with two blank
line indent                                        two-lines-top-
level-separator
  18:37  error    Function param name must be in
mixedCase                                                      f
unc-param-name-mixedcase
  25:5   error    Function order is incorrect, external
function can not go after external constant function  func-
order

contracts/SoundeonTokenMinter.sol
   1:17  warning  Compiler version must be
fixed                           compiler-fixed
  14:96  warning  Code contains empty
block                                 no-empty-blocks
  14:56  error    Visibility modifier must be first in list of
modifiers  visibility-modifier-order
  16:32  error    Function param name must be in
mixedCase                 func-param-name-mixedcase
  18:9   error    Expected indentation of 12 spaces but found
8          indent
  20:9   error    Expected indentation of 12 spaces but found
8          indent
  21:13  error    Expected indentation of 16 spaces but found
12         indent
  23:13  error    Expected indentation of 16 spaces but found
12         indent
  24:17  error    Expected indentation of 20 spaces but found
16         indent
  26:17  error    Expected indentation of 20 spaces but found
16         indent
  28:17  error    Expected indentation of 20 spaces but found
16         indent
  29:13  error    Expected indentation of 16 spaces but found
12         indent
  30:9   error    Expected indentation of 12 spaces but found
8          indent
  32:9   error    Expected indentation of 12 spaces but found
8          indent
  33:9   error    Expected indentation of 12 spaces but found
8          indent
  34:9   error    Expected indentation of 12 spaces but found
8          indent
```

```
  35:9   error    Expected indentation of 12 spaces but found
8           indent
  36:9   error    Expected indentation of 12 spaces but found
8           indent
  37:9   error    Expected indentation of 12 spaces but found
8           indent
  38:5   error    Expected indentation of 8 spaces but found
4            indent

contracts/SoundeonTokenSender.sol
   1:17  warning  Compiler version must be
fixed                           compiler-fixed
   7:96  warning  Code contains empty
block                                 no-empty-blocks
   7:56  error    Visibility modifier must be first in list of
modifiers  visibility-modifier-order
   9:36  error    Function param name must be in
mixedCase                 func-param-name-mixedcase
  12:9   error    Expected indentation of 12 spaces but found
8           indent
  13:13  error    Expected indentation of 16 spaces but found
12          indent
  14:17  error    Expected indentation of 20 spaces but found
16          indent
  16:17  error    Expected indentation of 20 spaces but found
16          indent
  17:13  error    Expected indentation of 16 spaces but found
12          indent
  18:9   error    Expected indentation of 12 spaces but found
8           indent
  19:5   error    Expected indentation of 8 spaces but found
4            indent
  21:40  error    Function param name must be in
mixedCase                 func-param-name-mixedcase
  24:9   error    Expected indentation of 12 spaces but found
8           indent
  25:13  error    Expected indentation of 16 spaces but found
12          indent
  26:17  error    Expected indentation of 20 spaces but found
16          indent
  28:17  error    Expected indentation of 20 spaces but found
16          indent
  29:13  error    Expected indentation of 16 spaces but found
12          indent
  30:9   error    Expected indentation of 12 spaces but found
8           indent
```

```
  31:5   error    Expected indentation of 8 spaces but found
4            indent

✖ 47 problems (40 errors, 7 warnings)
```

## Solium output

```
No issues found.
```