CWV Token Audit

The CWV Group team asked us to review and audit their ERC20 CWVToken contract. We looked at the code and now publish our results.

Good job using Armors-solidity to write minimal extra code. Excellent automated tests code coverage, with 100% for CWVToken.

Here is our assessment and recommendations, in order of importance.

Critical severity

No critical severity issues were found.

High severity

No high severity issues were found.

Medium severity

No medium severity issues were found.

Low severity

No low severity issues were found.

Testing process

```
Contract: CWVToken
  total supply
    ✓ returns the total amount of tokens (86ms)
  balanceOf
    when the requested account has no tokens
      ✓ returns zero (49ms)
    when the requested account has some tokens
      ✓ returns the total amount of tokens (50ms)
```

```
    transfer
      when the recipient is not the zero address
        when the sender does not have enough balance
          ✓ reverts (68ms)
        when the sender has enough balance
          ✓ transfers the requested amount (217ms)
          ✓ emits a transfer event (171ms)
      when the recipient is the zero address
        ✓ reverts (53ms)
    approve
      when the agent is not the zero address
        when the sender has enough balance
          ✓ emits an approval event (63ms)
          when there was no approved amount before
            ✓ approves the requested amount (104ms)
          when the agent had an approved amount
            ✓ approves the requested amount and replaces the previous one (9
0ms)
        when the sender does not have enough balance
          ✓ emits an approval event (60ms)
          when there was no approved amount before
            ✓ approves the requested amount (107ms)
          when the agent had an approved amount
            ✓ approves the requested amount and replaces the previous one (9
9ms)
      when the agent is the zero address
        ✓ approves the requested amount (95ms)
        ✓ emits an approval event (61ms)
    transfer from
      when the recipient is not the zero address
        when the agent has enough approved balance
          when the owner has enough balance
            ✓ transfers the requested amount (175ms)
            ✓ decreases the agent allowance (201ms)
            ✓ emits a transfer event (138ms)
          when the owner does not have enough balance
            ✓ reverts (54ms)
        when the agent does not have enough approved balance
          when the owner has enough balance
            ✓ reverts (56ms)
          when the owner does not have enough balance
            ✓ reverts (53ms)
      when the recipient is the zero address
        ✓ reverts (40ms)
    decrease approval
      when the agent is not the zero address
        when the sender has enough balance
          ✓ emits an approval event (114ms)
```

```
          when there was no approved amount before
            ✓ keeps the allowance to zero (110ms)
          when the agent had an approved amount
            ✓ decreases the agent allowance subtracting the requested amount
 (115ms)
        when the sender does not have enough balance
          ✓ emits an approval event (77ms)
          when there was no approved amount before
            ✓ keeps the allowance to zero (102ms)
          when the agent had an approved amount
            ✓ decreases the agent allowance subtracting the requested amount
 (115ms)
      when the agent is the zero address
        ✓ decreases the requested amount (140ms)
        ✓ emits an approval event (79ms)
    increase approval
      when the agent is not the zero address
        when the sender has enough balance
          ✓ emits an approval event (82ms)
          when there was no approved amount before
            ✓ approves the requested amount (108ms)
          when the agent had an approved amount
            ✓ increases the agent allowance adding the requested amount (113
ms)
        when the sender does not have enough balance
          ✓ emits an approval event (71ms)
          when there was no approved amount before
            ✓ approves the requested amount (110ms)
          when the agent had an approved amount
            ✓ increases the agent allowance adding the requested amount (119
ms)
      when the agent is the zero address
        ✓ approves the requested amount (110ms)
        ✓ emits an approval event (77ms)

  Contract: CWVTokenKeeper
    batchTransfer
      when the account is not the owner
        ✓ revert (62ms)
      when the keeper has no token
        ✓ revert (83ms)
      when the keeper does not have enough token
        ✓ revert (228ms)
      when the funders.length and amounts.length are not the same
        ✓ revert (130ms)
      when the keeper has enough token
        ✓ funders got tokens (700ms)
```

```
  Contract: Ownable
    ✓ should have an owner
    ✓ changes owner after transfer (100ms)
    ✓ should prevent non-owners from transfering (63ms)
    ✓ should guard ownership against stuck state (75ms)

  Contract: SafeERC20
    ✓ should throw on failed transfer (51ms)
    ✓ should throw on failed transferFrom (55ms)
    ✓ should throw on failed approve (49ms)
    ✓ should not throw on succeeding transfer (66ms)
    ✓ should not throw on succeeding transferFrom (109ms)
    ✓ should not throw on succeeding approve (70ms)

  Contract: SafeMath
    add
      ✓ adds correctly (44ms)
      ✓ throws an error on addition overflow (43ms)
    sub
      ✓ subtracts correctly (38ms)
      ✓ throws an error if subtraction result would be negative (39ms)
    mul
      ✓ multiplies correctly (42ms)
      ✓ handles a zero product correctly
      ✓ throws an error on multiplication overflow (39ms)
    div
      ✓ divides correctly (42ms)
      ✓ throws an error on zero division


  62 passing (17s)

|--------------------|----------|----------|----------|----------|
| File               |  % Stmts | % Branch |  % Funcs |  % Lines |
|--------------------|----------|----------|----------|----------|
| contracts/         |      100 |      100 |      100 |      100 |
|  CWVToken.sol      |      100 |      100 |      100 |      100 |
|  CWVTokenKeeper.sol|      100 |      100 |      100 |      100 |
|--------------------|----------|----------|----------|----------|
| All files          |      100 |      100 |      100 |      100 |
|--------------------|----------|----------|----------|----------|
```


Classic case test result

Use safe math

```
PASSED!
```

It may not be possible to stake tokens on an invalid outcome

```
PASSED!
```

Integer index types are unnecessarily small

```
PASSED!
```

Unbound iteration in arrays

```
PASSED!
```

Reentrancy risk

```
PASSED!
```


Naming issues

```
PASSED!
```

Unused boolean return values

```
PASSED!
```

Unsolved TODO comments

```
PASSED!
```

Inconsistent usage of getter functions and state variables

```
PASSED!
```

Use a standard toolchain for building contracts

```
PASSED!
```

No assertions for detecting broken invariants

```
PASSED!
```

Outdated Armors' contracts

```
PASSED!
```

Armors standard tokens were modified

```
PASSED!
```


Conclusion

No critical or high severity issues were found. Some changes were proposed to follow best practices and reduce potential attack surface.

Note that as of the date of publishing, the above review reflects the current understanding of known security patterns as they relate to the ERC20 CWV Token contract. The above should not be construed as investment advice.