SMART CONTRACT AUDITING REPORT

Reported To - KMPARDS

Token - EraswapToken.sol

Date - 15.05.2019

Prepared By

OSIZ Technologies Pvt Ltd

Intent

The smart contract auditing is being carried out to decrease the potential risks, security issues and the major vulnerabilities which may breach the confidentiality of a contract. The contract has been verified based on various aspects, majorly on security, logical aspect, solidity code compliance, standard of ERC20, protocol's login.

A smart contract audit is basically the same as a conventional code audit: it aims at finding security vulnerabilities before the code is deployed. Over/Underflows, Reentrancy, and Front Running are among the most widespread smart contract vulnerabilities.

Token Name - Eraswap

Token Index - ES

Token Standard - ERC20

Blockchain - Ethereum

Components of Auditing

1. State variables Shadowing
2. Functions which destructs the contract
3. Controlled delegate call destination
4. Optimization of Code
5. Re-Inforce Authorization
6. Vulnerability check
7. Control of Short address
8. Overflow and Underflow of variables
9. Gas limit and value control to avoid conflict error

Manual Auditing

1. Finding Errors and common Attacks
   - ❏ Re-entrancy attack
   - ❏ Replay attack
   - ❏ Reordering attack
   - ❏ Short address attack
   - ❏ Overflows and Underflows
2. Discovering errors of the Host platform
3. Checking for the possibility of security problems (Current and future)

Findings

Version

Latest version of Solidity is used ^0.5.5. It is a great way to tackle old vulnerabilities which present in old versions.

Constructor() is introduced on and above 0.4.24, here Constructor has been used in version 0.4.13.

Recommended to change the version to ^0.4.24

Standard format

• Using require(), needs “Error message” after the condition.

Example: require((a>b), “a should be greater than b”);

Since require increases gas consumption its usage is recommended in place of need. Also its usage should follow standard format.

• Visibility modifier should used before the modifier.

Example: function() public onlyOwner {}

• Usage of memory is highly recommended for strings then storage because of its transaction cost consumption.

Tools Applied for Automated Auditing

Auditing Succinct

Line by line comments

Line no : 735

This is nice.it protects from many errors and attacks. But the line below function will itself check for condition, which require in this line trying to do.

➔ **Recommendation**

require() condition is not recommended.

Similar statement applies for line no 721 and 709.

Line no : 661

Limiting the array length will help in reduction of gas consumption in great extent.

Example - arr[10];

➔ **Recommendation**

Since only 9 pools are playing roles here, it's a good practice to mention the array length.

Line no : 395

Redeclaration of variable causes, variable underflow which may result in a bug. Also it may result in extra execution cost.

In contract Detailed ERC20, variable name,and symbol are declared. Also in its child contract EraswapERC20 the variables have been declared. Similarly for variable declared in contract Capped Token

Note

Using line no 405, it will assign its values respectively. There is no need to assign in line no 396 and 397.

➔ Recommendation

Remove the redeclared variables.

	High Priority	Medium Priority	Low Priority
Line no : 735			✔
Line no : 661	✔		
Line no : 395	✔		

Re-audit Report

- Check for variable re-declaration in contract EraswapERC20

- Solve the method overriding error in function mint. Contract - MintableToken

  Change from internal to public.

- Use double quotes for error message.Line no 643.

TRUFFLE Report

```
devel-karthiga@devel-karthiga:~$ truffle run verify
Compiling ./contracts/EraswapToken.sol...
Compiling ./contracts/SafeMath.sol...
Writing artifacts to ./build/mythx/contracts

     BasicToken |*****************| 100% || Elapsed: 87.3s ✓ completed
  BurnableToken |*****************| 100% || Elapsed: 88.3s ✓ completed
    CappedToken |*****************| 100% || Elapsed: 54.8s ✓ completed
  DetailedERC20 |*****************| 100% || Elapsed: 25.6s ✓ completed
          ERC20 |*****************| 100% || Elapsed: 29.2s ✓ completed
     ERC20Basic |*****************| 100% || Elapsed: 26.6s ✓ completed
   EraswapERC20 |*****************| 100% || Elapsed: 79.2s ✓ completed
   EraswapToken |*****************| 100% || Elapsed: 86.1s ✓ completed
  MintableToken |*****************| 100% || Elapsed: 87.0s ✓ completed
     NRTManager |*****************| 100% || Elapsed: 82.1s ✓ completed
        Ownable |*****************| 100% || Elapsed: 76.3s ✓ completed
PausableEraswap |*****************| 100% || Elapsed: 79.4s ✓ completed
  StandardToken |*****************| 100% || Elapsed: 81.5s ✓ completed
       SafeMath |*****************| 100% || Elapsed: 55.3s ✓ completed

/home/devel-karthiga/contracts/EraswapToken.sol
  42:14  warning  assertion violation                       SWC-110
  46:6   error    The binary addition can overflow          SWC-101
  46:19  warning  A reachable exception has been detected   SWC-110

✖ 3 problems (1 error, 2 warnings)
```

Errors and Warnings has been rectified

Test case has been written

Business Logic Result

After updating the pool address, monthly NRT will be released. This process is been checked for complete 50 years. Refer the attached document for reference.

Initial values of total supply and annual NRT is taken from the contract. And business logic result values is been checked with truffle testcase.

Initial supply is 91,00,00,000

Total 50 Years	Annual NRT	Cumulative Supply Month on Month	Total supply after burning
Year 1	819000000	978250000	960050000
Year 2	737100000	1790425000	1481196471
Year 3	663390000	2521382500	1817365904
Year 4	597051000	3179244250	2015659054
Year 5	537345900	3771319825	2112308841
Year 6	483611310	4304187843	2135092799
Year 7	435250179	4783769058	2105218879
Year 8	391725161	5215392152	2038798729
Year 9	352552645	5603852937	1947997899
Year 10	317297380	5953467643	1841933126
Year 11	285567642	6268120879	1727371665
Year 12	257010878	6551308791	1609275777
Year 13	231309790	6806177912	1491226124
Year 14	208178811	7035560121	1375750536
Year 15	187360930	7242004109	1264578866
Year 16	168624837	7427803698	1158840155
Year 17	151762353	7595023328	1059214824

Year 18	136586118	7745520995	966051803
Year 19	122927506	7880968896	879458401
Year 20	110634756	8002872006	799368970
Year 21	99571280	8112584806	725597129
Year 22	89614152	8211326325	657875239
Year 23	80652737	8300193693	595884035
Year 24	72587463	8380174323	539274668
Year 25	65328717	8452156891	487684900
Year 26	58795845	8516941202	440750842
Year 27	52916261	8575247082	398115267
Year 28	47624635	8627722373	359433347
Year 29	42862171	8674950136	324376434
Year 30	38575954	8717455123	292634381
Year 31	34718359	8755709610	263916783
Year 32	31246523	8790138649	237953436
Year 33	28121870	8821124784	214494210
Year 34	25309683	8849012306	193308547
Year 35	22778715	8874111075	174184667
Year 36	20500844	8896699968	156928617
Year 37	18450759	8917029971	141363206
Year 38	16605683	8935326974	127326899
Year 39	14945115	8951794277	114672691
Year 40	13450603	8966614849	103267008
Year 41	12105543	8979953364	92988635
Year 42	10894989	8991958028	83727695
Year 43	9805490	9002762225	75384684
Year 44	8824941	9012486002	67869568
Year 45	7942447	9021237402	61100937
Year 46	7148202	9029113662	55005223
Year 47	6433382	9036202296	49515985
Year 48	5790044	9042582066	44573241
Year 49	5211039	9048323860	40122866
Year 50	4689935	9053491474	36116032

Result

The contract has been audited based on the above mentioned criteria, checked for any critical issues and made sure it is feasible for business. The identified errors have been rectified based on the business logic for the next 50 years to bring out the best results. The contract is audited and ready to launch it in the market which can be further used for transactions.