ERC20 Element Token Audit

OPENZEPPELIN SECURITY | MARCH 27, 2018

Security Audits

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

The audited code is located in the element-group/element-erc20-smart-contract repository. The version used for this report is commit `6a7b48cec1aa8a823b7ec4a03adadc5ca9d22e29`.

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

Constructor could overflow in initialization of total supply

The `ElementToken` constructor does not perform any precondition checks on the tokens argument. Based on how totalSupply is being calculated, this could result in an overflow in the

precondition check to prevent an overflow.

Notes & Additional Information

- The last release of the OpenZeppelin framework includes a parameterized ERC20 token contract called DetailedERC20. Consider inheriting ElementToken from it to reuse that functionality.
- Consider reusing OpenZeppelin's PausableToken, which already makes all token fuctionality pausable.
- It is possible to create an instance of ElementToken with a zero-valued totalSupply. Consider adding a precondition in the constructor function.
- The ElementToken contract initializes sets initial values for the state variables name, symbol and decimals unnecessarily. Consider removing the initialization for each.
- The increaseApproval and decreaseApproval functions of the ElementToken contract define a name for the return value (success) which is unused. Consider removing said variable.
- It is important to specify function visibility modifiers explicitly to avoid confusion. Consider adding `public` to the `ElementToken` constructor function.
- Consider following the conventions proposed by the Solidity style guide for the transfer and transferFrom functions, according to which function visibility modifiers should come before any custom modifiers.
- Consider upgrading to the latest version of Solidity (0.4.19), which comes with the last release of Truffle.

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 ElementToken contract. We have not reviewed the related Element project. The above should not be construed as investment advice. For general information about smart contract security, check out our thoughts here.

Related Posts

Beefy Zap Audit

BeefyZapRouter serves as a versatile intermediary designed to execute users' orders through routes...

Security Audits

OpenBrush Contracts Library Security Review

OpenBrush is an open-source smart contract library written in the Rust programming language and the...

Security Audits

Linea Bridge Audit

Linea is a ZK-rollup deployed on top of Ethereum. It is designed to be EVM-compatible and aims to...

Security Audits