SOLIDIFIED

Audit Report for Solidified. June 13, 2018.

Summary

Audit report prepared internally by two independent Solidified auditors for Solidified Token Sale contracts

Process and Delivery

Two (2) independent Solidified experts performed an isolated audit of the below contracts. The debrief took place on June 13, 2018 and the final results are presented here.

Audited Files

The following files were covered during the audit:

```
contracts
 ├──── Distributable.sol
 ├──── SolidToken.sol
 └──── TokenSale.sol
```

Intended Behavior

The intended behavior of the token sale and token can be found here.

The audit was based on commit `f46573fdbc2c79600e88a0a321ed40ca9ba84d76`.

Issues Found

Major

1. If the sale is finalized by updateStage, setTransferEnablingDate will never be called

---

The code contains function `updateStage` to handle the edge case in which the amount needed to close the sale is less than the minimum contribution, however if this function is used to finalize the sale only `currentStage = Stages.FINALAIZED;` is executed as opposed to function `finalizeSale` which also calls `setTransferEnablingDate` in `SolidToken`. This means using `updateStage` to finalize the sale will result in the tokens never being transferable.

TokenSale.sol / Line 212-216

```
function finalizeSale() atStage(Stages.MAINSALE) internal {
    mainSale_EndDate = now;
    require(SolidToken(token).setTransferEnablingDate(now + 182 days));
    currentStage = Stages.FINALAIZED;
}
```

Recommendation
On line 138, call `finalizeSale` instead of only setting `currentStage` to `Stages.FINALAIZED`

AMENDED [2018-6-26]:
This issue is no longer present in commit `9ca5e0bb28efba1693ce3d506c0df46866f11a25`

2. TokenSale extends Pausable but it's never used

`TokenSale` extends `Pausable` which implies intent to give the ability to emergency stop accepting token purchases, however modifier `whenNotPaused` is not applied to any functions.

Recommendation
Add modifier `whenNotPaused` to function `saleOpen()`, if the ability to block all purchases in an emergency is desired. Note that the sale itself will not “pause”, sale time will continue to run.

AMENDED [2018-6-26]:
This issue is no longer present in commit `9ca5e0bb28efba1693ce3d506c0df46866f11a25`

3. In OpenZeppelin's TokenVesting contract, revoking the vest immediately transfers the unvested tokens to the owner

---

In OpenZeppelin's `TokenVesting` contract, revoking the vest immediately transfers the unvested tokens to the owner of the contract. Solidified could abuse this functionality to bypass token vesting entirely by revoking the vesting for a given address, then immediately transferring those tokens from the owner to said address.

Recommendation
As an alternative to revoking, allow the owner to change the beneficiary of the vesting contract. This ensures Solidified will be unable to break commitment of the vesting schedule, while allowing them to redistribute tokens when a team member leaves before their vest schedule runs its course.

AMENDED [2018-6-26]:
This issue is no longer present in commit `9ca5e0bb28efba1693ce3d506c0df46866f11a25`

4. TokenSale only checks that the address that receives purchased tokens is whitelisted, not the address that contributed the ETH

TokenSale does not check whether the address that contributed the ETH to the sale is whitelisted, and allows purchasing on the behalf of other addresses. This could pose an issue for regulatory compliance.

Recommendation
Check that the contribution address (the address that sent the ETH) and the beneficiary address (the address set to receive the tokens) are the same.

AMENDED [2018-6-26]:
This issue is no longer present in commit `9ca5e0bb28efba1693ce3d506c0df46866f11a25`

Note

6. Minimum contribution could be enforced per address rather than per transaction

---

Just as maximum contribution is enforced on a per address level, so could minimum contribution. We believe this better captures the intent of the minimum contribution limit.

AMENDED [2018-6-26]:
This issue is no longer present in commit `9ca5e0bb28efba1693ce3d506c0df46866f11a25`

7. presale_TokenCap and mainsale_TokenCap are redundant

This functionality is covered the ETH caps, these variables are never checked (only set) and therefore do not enforce anything additional.

8. Intent of _postValidatePurchase violated

We believe the intent of the design of the original zeppelin crowdsale contract is for `_postValidatePurchase` to not influence the contract state; instead state changing actions should be in `_updatePurchasingState`, return of excess ETH could be moved to `forwardFunds`

9. If there was a min contribution exception when topping up caps, function updateState would be unnecessary

`updateState` is not a clean workaround for the sale stalling as described in issue #1, instead there could be an exception to the minimum contribution requirement when topping up caps.

AMENDED [2018-6-26]:
This issue is no longer present in commit `9ca5e0bb28efba1693ce3d506c0df46866f11a25`

10. capReached doesn't trigger if cap is reached exactly

`capReached` is not set to `true` if the cap is met exactly. The sale can be “unstuck” by `updateState`, but this state is otherwise not ideal.

Recommendation
This conditional:
TokenSale.sol / 239

```
if(raised.add(acceptedValue) > currentCap){
```

should become

```
if(raised.add(acceptedValue) >= currentCap){
```

AMENDED [2018-6-26]:
This issue is no longer present in commit `9ca5e0bb28efba1693ce3d506c0df46866f11a25`

11. Typos

`TokesSold` should be `TokensSold`.
`FINALAIZED` should be `FINALIZED`.

AMENDED [2018-6-26]:
This issue is no longer present in commit `9ca5e0bb28efba1693ce3d506c0df46866f11a25`

12. Prefer constants

The contribution limits, the ETH caps, the token address, and the rate could all be set as constants rather than set through functions. As there is no need for multiple deployments with different values, constants are preferred for their explicitness.

AMENDED [2018-6-26]:
This issue is no longer present in commit `9ca5e0bb28efba1693ce3d506c0df46866f11a25`

Audit Report for Solidified. June 13, 2018.

Closing Summary

Multiple issues were found during the audit that can break the desired behaviour. It's strongly advised that these issues be corrected before proceeding to production. SolidToken has been verified as fully ERC20 compliant.

AMENDED [2018-6-26]:
All major issues have been addressed.

Disclaimer

This audit is not a security warranty, investment advice, or an endorsement of Solidified or its products. This audit does not provide a security or correctness guarantee of the audited smart contracts. Securing smart contracts is a multistep process, therefore running a bug bounty program as a complement to this audit is strongly recommended.

The individual audit reports are anonymized and combined during a debrief process, in order to provide an unbiased delivery and protect the auditors of Solidified platform from legal and financial liability.

Solidified Ltd.