Zer0 Zfi

Date	December 2021

1 Executive Summary

This report presents the results of our engagement with Zer0 to review zAuction and zFi.

The review was conducted over two weeks, from 12/6/2021 to 12/17/2021 by Eli Leers. A total of 10 person-days were spent.

During the first week, we familiarized ourselves with the zAuction and zFi systems. We inspected the zAuction contract's new BuyNow feature, as well as ensured that the rest of the contract addressed the recommendations from the previous audit. We also began the inspection of the zFi contracts, and became familiar with the Illuvium pools that the zFi pools were forked from.

During the second week we continued inspection of the zFi contracts. We focused primarily on the changes to the staking and rewards system.

2 Scope

Our review focused on the commit hash `aa6f68d5a9a61d7032e499ec3b4fb883bdc2b185` for zAuction and the commit hash `c12ada109a10941f9510ee5bab6585416a8c321e` for zFi. The list of files in scope can be found in the Appendix.

2.1 Objectives

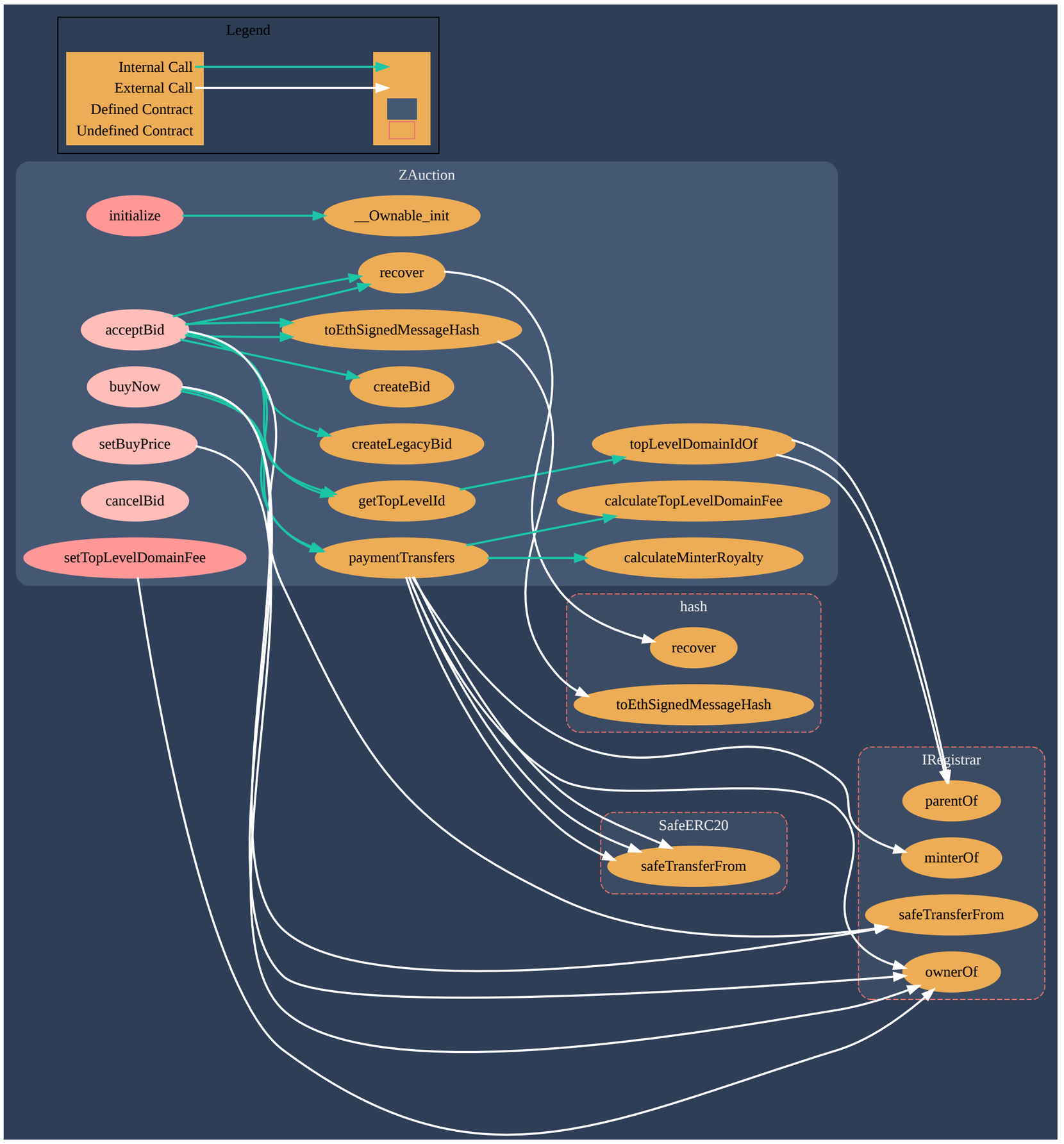
Together with the Zer0 team, we identified the following priorities for our review:

1. Ensure that the system is implemented consistently with the intended functionality, and without unintended edge cases.
2. Identify known vulnerabilities particular to smart contract systems, as outlined in our Smart Contract Best Practices, and the Smart Contract Weakness Classification Registry.
3. The Buy Now feature of zAuction works as intended.
4. Ensure that the changes to the zFi fork of Illuvium's pool contracts are safe and implemented correctly.

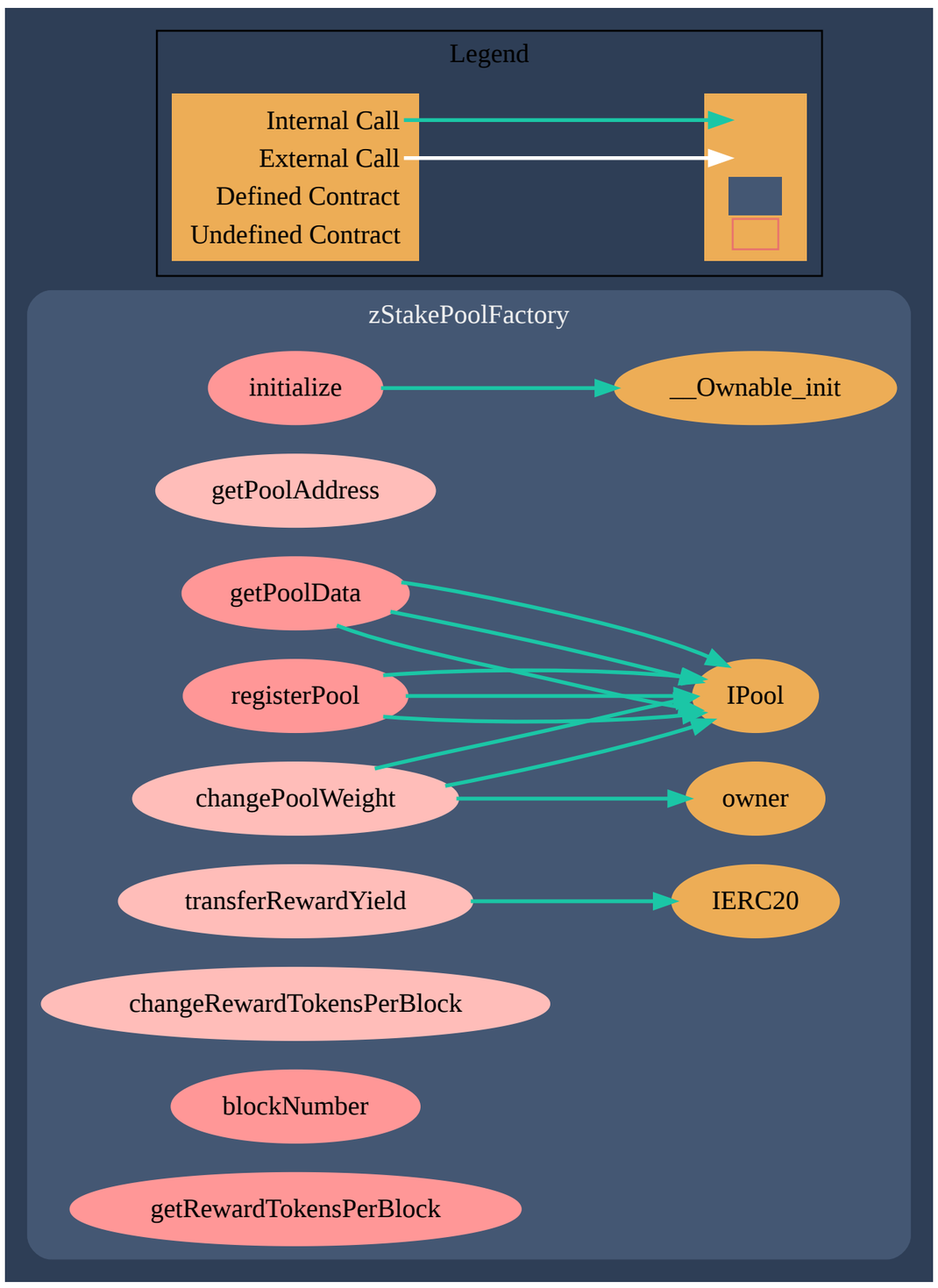
This section describes the top-level/deployable contracts, their inheritance structure and interfaces, actors, permissions and important contract interactions of the initial system under review. This section does not take any fundamental changes into account that were introduced during or after the review was conducted.

`zAuction` is a simple general purpose auction system for NFT's allowing bidders to share bids on an async 2nd layer. Bid price is paid in `WETH` by approving it to the auction contract or in `ETH` by depositing it in a `WETH`-like custom `zAuctionAccountant` contract. Auctions are state-less. The user journey starts with a bidder sharing a bid on a 2nd layer. The current holder of an NFT can then call one of the `accept*` functions to accept a specific bid and transfer ownership to the recipient. Due to the state-less nature of this system, auctions are not explicitly created by an nft owner.

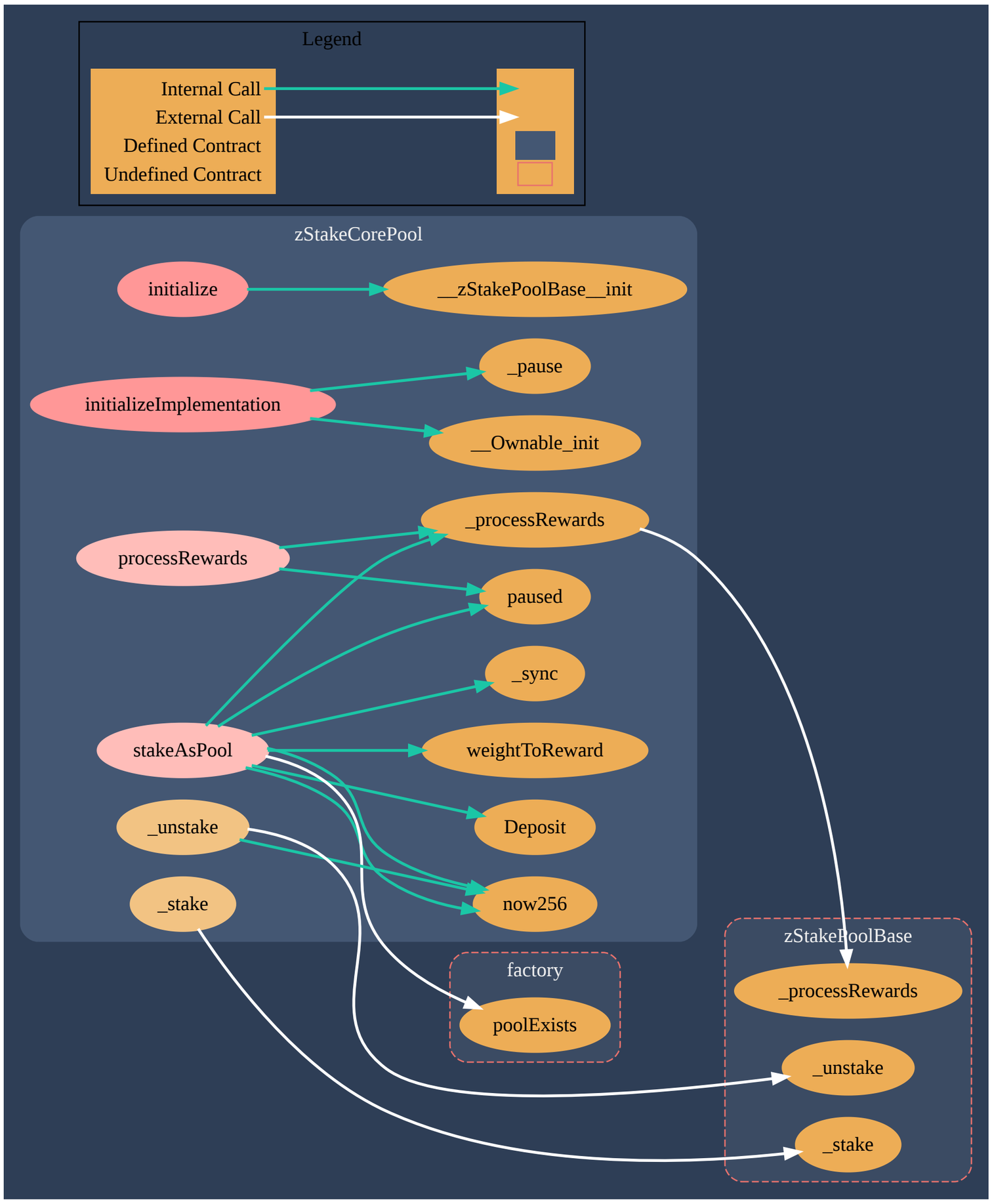
`zFi` consists of a factory and 2 pools: a WILD/ETH Uniswap LP Staking Pool, and a WILD Staking Pool. A user can stake tokens in either pool to generate rewards in WILD tokens. If locked for an entire year, the user will generate rewards at a 2x rate. These rewards will be locked for 1 year (unless the period is changed by the owner of the pool, after which all new rewards will be staked for the new period of time).



zAuction



zPoolFactory



zStakeCorePool

3 Recommendations

4 Findings

Each issue has an assigned severity:

- Minor issues are subjective in nature. They are typically suggestions around best practices or readability. Code maintainers should use their own judgment as to whether to address such issues.
- Medium issues are objective in nature but are not security vulnerabilities. These should be addressed unless there is a clear reason not to.
- Major issues are security vulnerabilities that may not be directly exploitable or may require certain conditions in order to be exploited. All major issues should be addressed.
- Critical issues are directly exploitable security vulnerabilities that need to be fixed.

4.1 rewardsLockPeriod can be set to a length longer than 1 year Medium

Description

Much of the math on rewards calculation are dependent on the stake being locked for up to 1 year. The rewardsLockPeriod can be set to a longer time, which may negatively affect those reward calculations.

Examples

code/contracts/zStakePoolBase.sol:L477-L483

```
/**
 * @dev Allows for the rewardLockPeriod to be modified.
 */
function changeRewardLockPeriod(uint256 _rewardLockPeriod) external onlyOwner {
  require(rewardLockPeriod != _rewardLockPeriod, "same rewardLockPeriod");
  rewardLockPeriod = _rewardLockPeriod;
}
```

code/contracts/zStakePoolBase.sol:L446-L449

```
// stake weight formula rewards for locking
uint256 stakeWeight = (((lockUntil - lockFrom) * WEIGHT_MULTIPLIER) /
  365 days +
  WEIGHT_MULTIPLIER) * addedAmount;
```

Recommendation

Add a check to ensure the new rewardsLockPeriod is less than or equal to 365 days in the changeRewardsLockPeriod() function.

5 Security Specification

This section describes, **from a security perspective**, the expected behavior of the system under audit. It is not a substitute for documentation. The purpose of this section is to identify specific security properties that were validated by the audit team.

5.1 Actors

The relevant actors are listed below with their respective abilities:

zAuction

- Admin
  - Same entity as deployer
  - Can upgrade the contract
- Seller
  - Can create Auctions on a layer 2 dApp.
  - Can set the buy price for a token that they own
  - Can accept a bid created by a buyer on a layer 2 dApp
- Buyer
  - Can create bids on a Layer 2 dApp, and share a signed message with their bid to a Seller.
  - Can cancel bid
  - Can Buy Now at a price set by seller

zFi

- Admin
  - Same entity as deployer
  - Can upgrade the contract
  - Can change the reward staking period
- WILD Token Holder / Staker
  - Can stake tokens
  - Can increase the period that a stake they own is locked
- Factory
  - Can update the rewards weight of the pool

5.2 Trust Model

In any system, it's important to identify what trust is expected/required between various actors. For this audit, we established the following trust model:

- Users are trusting Zer0 team `admin` as they have the capability to upgrade any contract in this system at will. They may also change the reward lock period, and by upgrading the factory could modify the pool weight at will.

Appendix 1 - Files in Scope

This audit covered the following files:

zAuction/contracts/Zauction.sol

zFI/contracts/zStakePoolFactory.sol

zFI/contracts/zStakePoolBase.sol

zFI/contracts/zStakeCorePool.sol

Appendix 2 - Disclosure

ConsenSys Diligence ("CD") typically receives compensation from one or more clients (the "Clients") for performing the analysis contained in these reports (the "Reports"). The Reports may be distributed through other means, including via ConsenSys publications and other distributions.

The Reports are not an endorsement or indictment of any particular project or team, and the Reports do not guarantee the security of any particular project. This Report does not consider, and should not be interpreted as considering or having any bearing on, the potential economics of a token, token sale or any other product, service or other asset. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. No Report provides any warranty or representation to any Third-Party in any respect, including regarding the bugfree nature of code, the business model or proprietors of any such business model, and the legal compliance of any such business. No third party should rely on the

Reports in any way, including for the purpose of making any decisions to buy or sell any token, product, service or other asset. Specifically, for the avoidance of doubt, this Report does not constitute investment advice, is not intended to be relied upon as investment advice, is not an endorsement of this project or team, and it is not a guarantee as to the absolute security of the project. CD owes no duty to any Third-Party by virtue of publishing these Reports.

PURPOSE OF REPORTS The Reports and the analysis described therein are created solely for Clients and published with their consent. The scope of our review is limited to a review of code and only the code we note as being within the scope of our review within this report. Any Solidity code itself presents unique and unquantifiable risks as the Solidity language itself remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond specified code that could present security risks. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. In some instances, we may perform penetration testing or infrastructure assessments depending on the scope of the particular engagement.

CD makes the Reports available to parties other than the Clients (i.e., “third parties”) – on its website. CD hopes that by making these analyses publicly available, it can help the blockchain ecosystem develop technical best practices in this rapidly evolving area of innovation.

LINKS TO OTHER WEB SITES FROM THIS WEB SITE You may, through hypertext or other computer links, gain access to web sites operated by persons other than ConsenSys and CD. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites’ owners. You agree that ConsenSys and CD are not responsible for the content or operation of such Web sites, and that ConsenSys and CD shall have no liability to you or any other person or entity for the use of third party Web sites. Except as described below, a hyperlink from this web Site to another web site does not imply or mean that ConsenSys and CD endorses the content on that Web site or the operator or operations of that site. You are solely responsible for determining the extent to which you may use any content at any other web sites to which you link from the Reports. ConsenSys and CD assumes no responsibility for the use of third party software on the Web Site and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any outcome generated by such software.

TIMELINESS OF CONTENT The content contained in the Reports is current as of the date appearing on the Report and is subject to change without notice. Unless indicated otherwise, by ConsenSys and CD.