Audit Report

November 30, 2024

Big Blymp - Obymare

Contents

1 - Summary

This report provides a comprehensive audit of Obymare, a USD-pegged algorithmic stablecoin protocol.

The investigation spanned several potential vulnerabilities, including scenarios where attackers might exploit the validator to lock up or steal funds.

The audit is conducted without warranties or guarantees of the quality or security of the code. It's important to note that this report only covers identified issues, and we do not claim to have detected all potential vulnerabilities.

1.a - Overview

The Obymare protocol allows users to mint OBYUSD by opening a debt position and locking an ADA collateral into it. Users can then interact with their positions by increasing or reducing their debt, minting or burning OBYUSD respectively. They can also pay other users debt in exchange for their collateral. In the case of unhealthy positions, users can liquidate them with an additional compensation in OBYII tokens, the protocol's utility token.

The protocol consists of two types of UTxOs: Positions and Treasury.

To obtain OBYUSD debt, users open a Position by locking ADA as collateral while paying OBYIII as protocol fees (proportional to the taken debt), which go to the Treasury. The collateral must be above the Minimum Collateral Ratio (MCR) that depends on the current OBYUSD/ADA exchange rate. Users can then adjust the collateral and debt of their Position, as long as it complies with the MCR, and can close it by paying off the entire debt.

Users can also interact with Positions they do not own by liquidating them. This is akin to paying off someone else's debt, either partially or in full. For unhealthy positions i.e. those below the Critical Collateral Ratio (CCR), the liquidator can request assistance from the Treasury, which involves receiving OBYIII in exchange for paying off the Position's debt as an incentive.

A special UTxO, the Exchange Rate Pointer, provides the hash for the Exchange Rate withdraw script that validates the OBYUSD and OBYII prices given by the oracles. A privileged Admin actor controls the initialization and update of this UTxO.

1.b - Process

Our audit process involved a thorough examination of Obymare validators. Areas vulnerable to potential security threats were closely scrutinized, including those where attackers could exploit the validator's functions to disrupt the platform and its users. This included evaluating potential risks such as unauthorized asset addition, hidden market creation, and disruptions to interoperability with other Plutus scripts. This also included the common vulnerabilities such as double satisfaction and minting policy vulnerabilities.

The audit took place over a period of several weeks, and it involved the evaluation of the protocol's mathematical model to verify that the implemented equations matched the expected behavior.

Findings and feedback from the audit were communicated regularly to the Obymare team through Discord. Diagrams illustrating the necessary transaction structure for proper interaction with the protocol are attached as part of this report. The Obymare team addressed these issues in an efficient and timely manner, enhancing the overall security of the platform.

2 - Specification

2.a - UTxOs

2.a.a - Position

- Address: Main validator script hash
- Value:
  - N ADA
  - 1 user validity token
- Datum: `Position(Debt)` where `Debt` is `Int`

2.a.b - Treasury

- Address: Main validator script hash
- Value:
  - M OBYIII
- Datum: `Treasury`

2.a.c - Exchange Rate Pointer (ERP)

- Address: ERP validator script hash
- Value:
  - 1 ERP validity token
- Datum: `ErpDat`: `Credential`

2.b - Assets

2.b.a - OBYUSD

It is a USD stablecoin minted and paid to a user when he opens a position.

- Policy ID: hash of Main validator
- Token name: `tokens.obyusd()`

2.b.b - OBYIII

Obymare protocol token. When a user opens a position, he must pay OBYIII in concept of protocol fees which are stored in Treasury UTxOs.

- Policy ID: defined with parameter `obyiii_hash` in Main validator
- Token name: `tokens.obyiii()`

2.b.c - User validity

Identifies a user position. Uniquely twinned with a user auth token via the tag.

- Policy ID: hash of Main validator
- Token name: `tokens.user_validity_token(tag)`

2.b.d - User auth

The entity that holds this token is considered as owner of a particular position.

- Policy ID: hash of Main validator
- Token name: `tokens.user_auth_token(tag)`

2.b.e - ERP validity

Identifies the ERP UTxO.

- Policy ID: hash of ERP validator
- Token name: `tokens.erp_validity_token()`

2.b.f - ERP auth

Meant to be held by the Admin. Allows for updating the exchange rate source or closing the ERP.

- Policy ID: hash of ERP validator
- Token name: `tokens.erp_auth_token()`

2.c - Transactions

2.c.a - Users

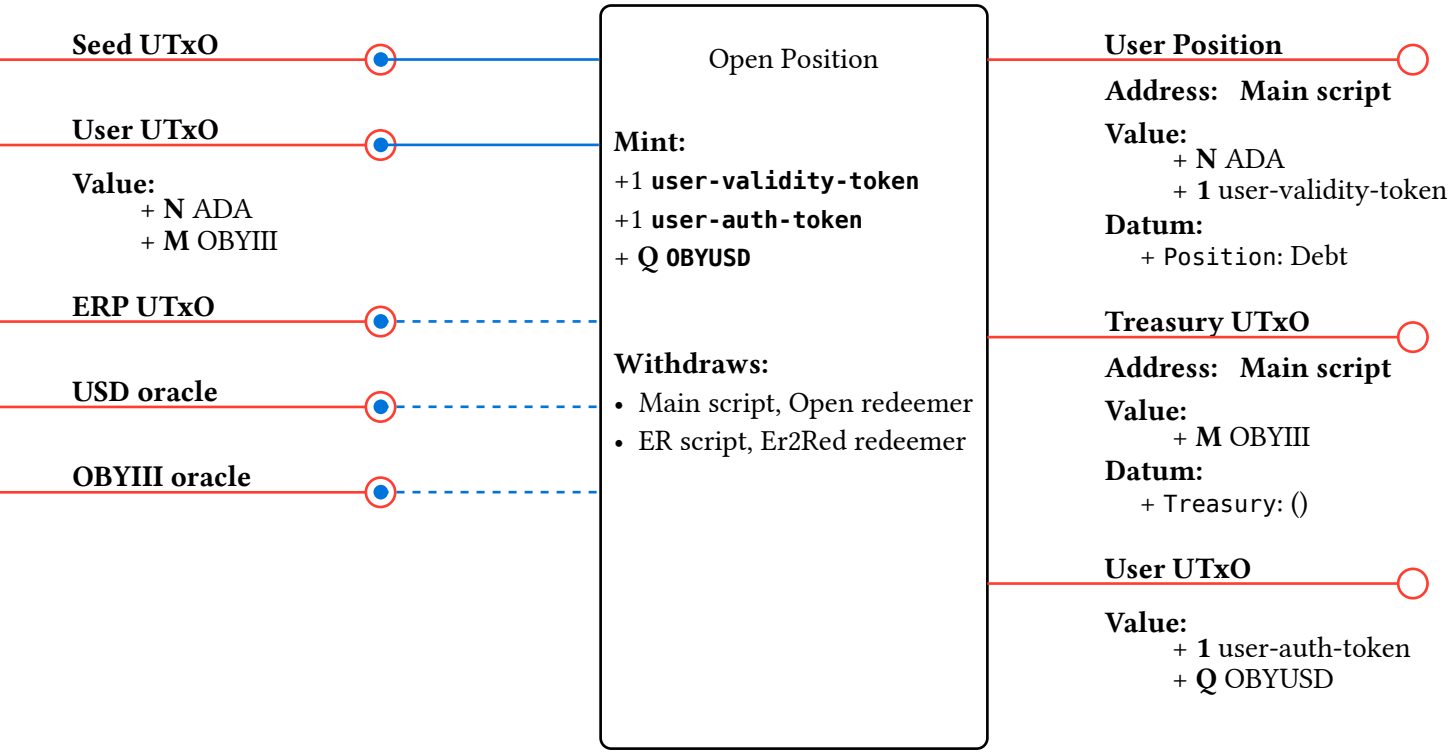
2.c.a.a - Open Position

A user mints OBYUSD by creating a position and locking an ADA collateral >= Minimal Collateral Ratio that is 120% of the minted OBYUSD according to current exchange rate OBYUSD/ADA. In addition, he must pay an OBYII fee of 2% of the collateral according to current exchange rate OBYII/ADA.

The ERP UTxO holds the correct ER script credential in its datum, and the ER script itself enforces the presence of the USD and OBYIII oracles in the reference inputs.

Involved redeemers:

- `Data`, `Mint` purpose: for minting OBYUSD, user auth, and user validity tokens
- `Open`, `WithdrawFrom` purpose: for Main script withdrawal
- `Er2Red`, `WithdrawFrom` purpose: for ER script withdrawal



Note: type Debt = Int

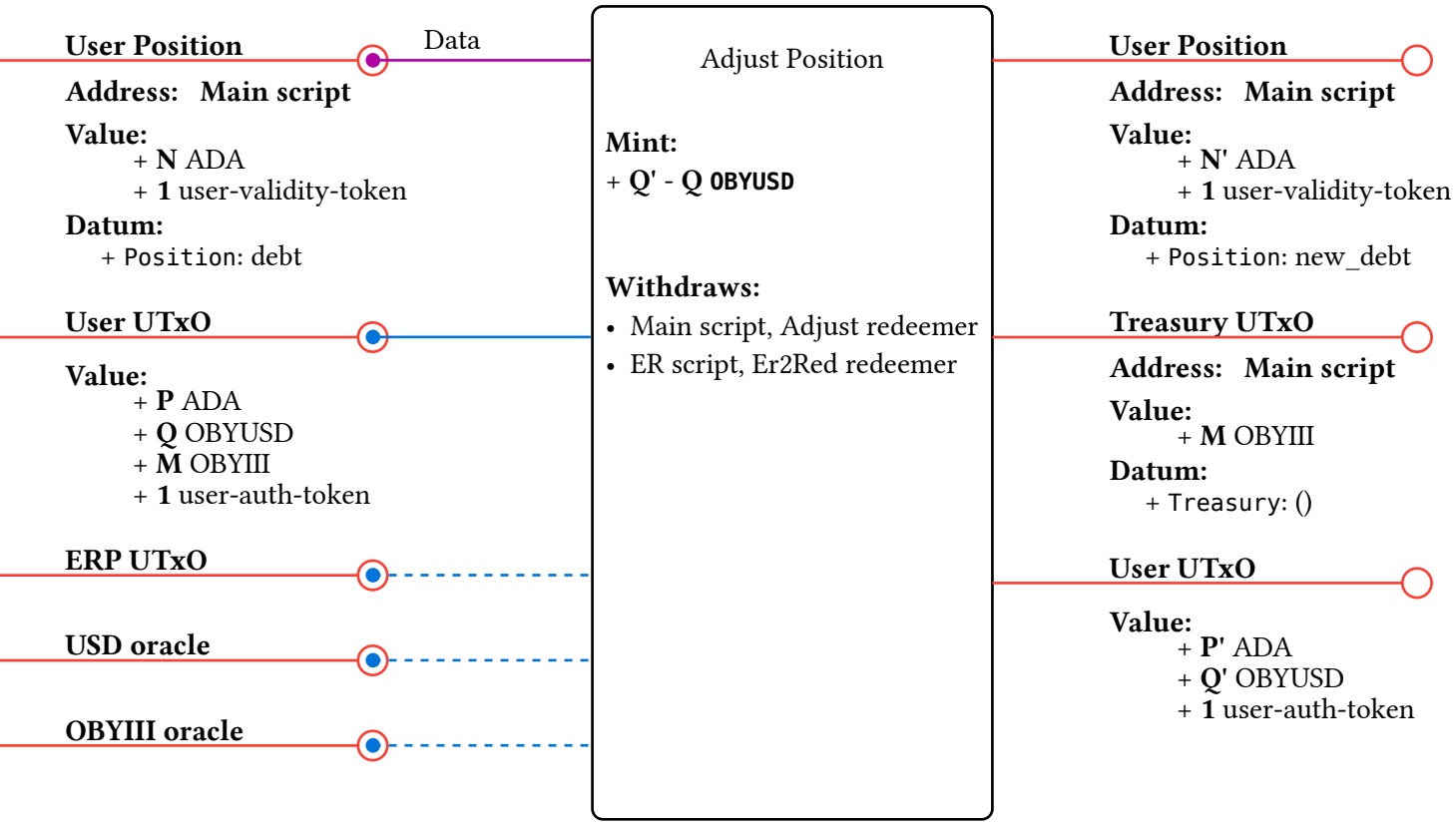
Figure 1: Open Position transaction

2.c.a.b - Adjust Position

The owner of a position adjusts it by changing the debt or the collateral. If the collateral is reduced or the debt is increased, it is checked that the Collateral Ratio is above the minimum. If the ADA collateral is increased, user must pay to a Treasury an OBYII fee of 2% of the added ADA.

Involved redeemers:

- `Data`, `Spend` purpose: for spending the Position UTxO
- `Adjust`, `WithdrawFrom` purpose: for Main script withdrawal
- `Data`, `Mint` purpose: for minting or burning OBYUSD
- `Er2Red`, `WithdrawFrom` purpose: for ER script withdrawal



Note:
N' = P' - P

Figure 2: Adjust Position transaction

2.c.a.c - Close Position

User closes his position by burning his auth token. The corresponding validity token is also burned. If there is still some OBYUSD debt, user must pay it to be burned. All stored collateral is released.

Involved redeemers:

- `Data`, `Spend` purpose: for spending the Position UTxO
- `Close`, `WithdrawFrom` purpose: for Main script withdrawal
- `Data`, `Mint` purpose: for burning OBYUSD if debt is greater than zero

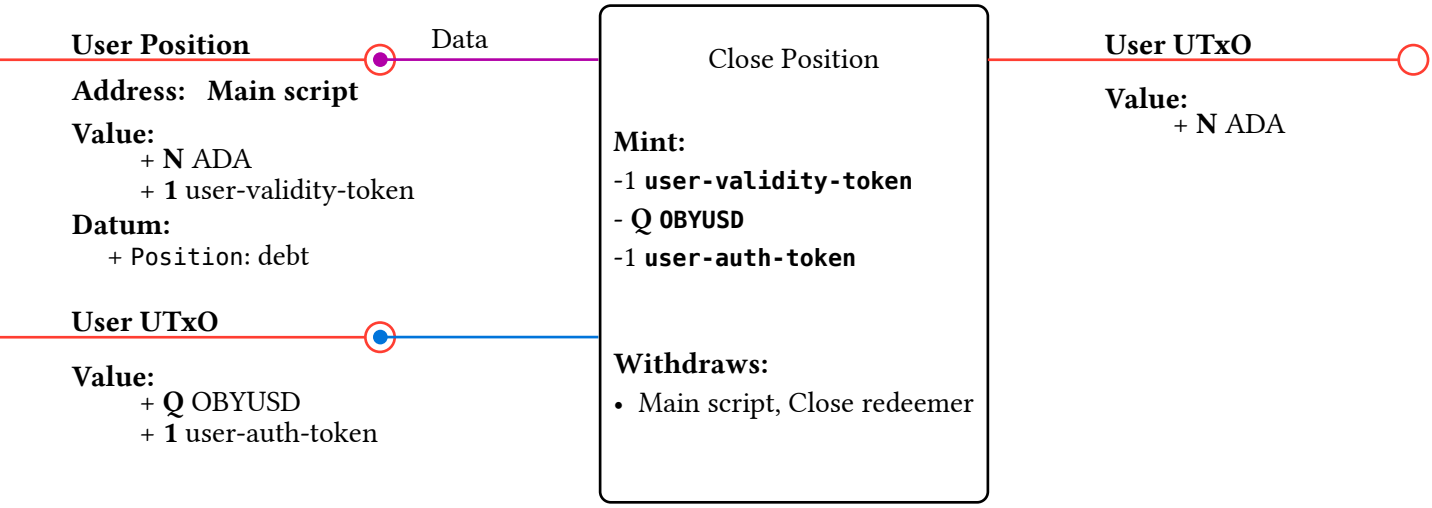


Figure 3: Close Position transaction

2.c.a.d - Redeem Positions

The user burns OBYUSD to paid debt of some positions. At least 1_000 million OBYUSD must be paid. At most one debt can be partially paid, all others must be fully paid. User can take ADA collateral in proportion to paid debt, and depending on current exchange rate. For the user, this transaction acts as a swap of OBYUSD for ADA following current exchange rate and limited by ADA availability in the collaterals. In addition, user must pay to a Treasury an OBYII fee that is in proportion of the paid debt, the fee depending linearly to the Collateral Ratio of each position, from 1% (when CR<=150%) to 3% (when CR>=500%).

Involved redeemers:

- `Data`, `Spend` purpose: one for each Position UTxO spent
- `Redeem`, `WithdrawFrom` purpose: for Main script withdrawal
- `Data`, `Mint` purpose: for burning OBYUSD

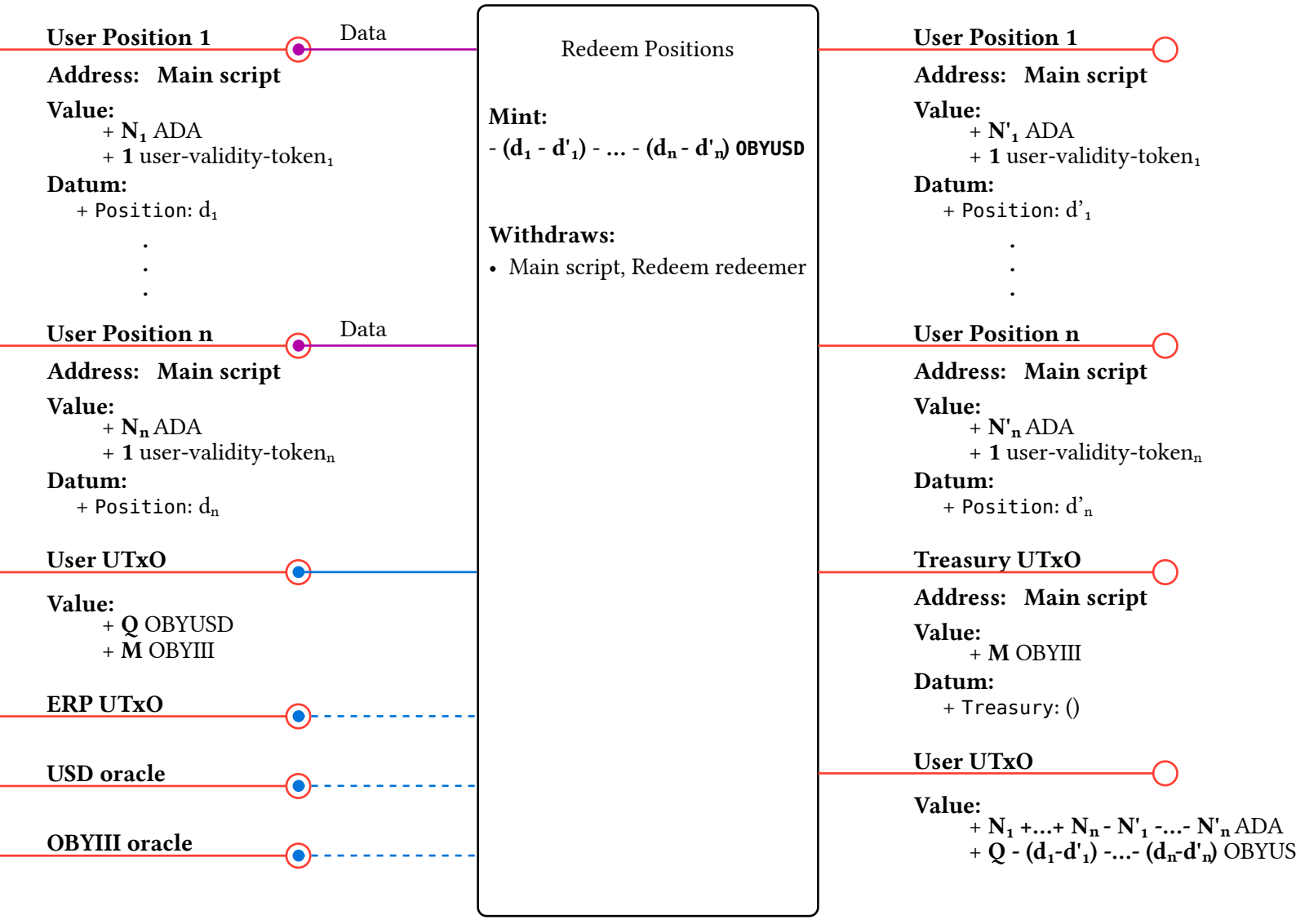


Figure 4: Redeem Positions transaction

2.c.a.e - Liquidate Position

The user liquidates a position that is below the Minimum Collateral Rate. He burns OBYUSD to pay the debt and takes all the collateral.

Involved redeemers:

- `Data`, `Spend` purpose: for spending the Position UTxO
- `Lqd`, `WithdrawFrom` purpose: for Main script withdrawal
- `Data`, `Mint` purpose: for burning the Position's user validity token

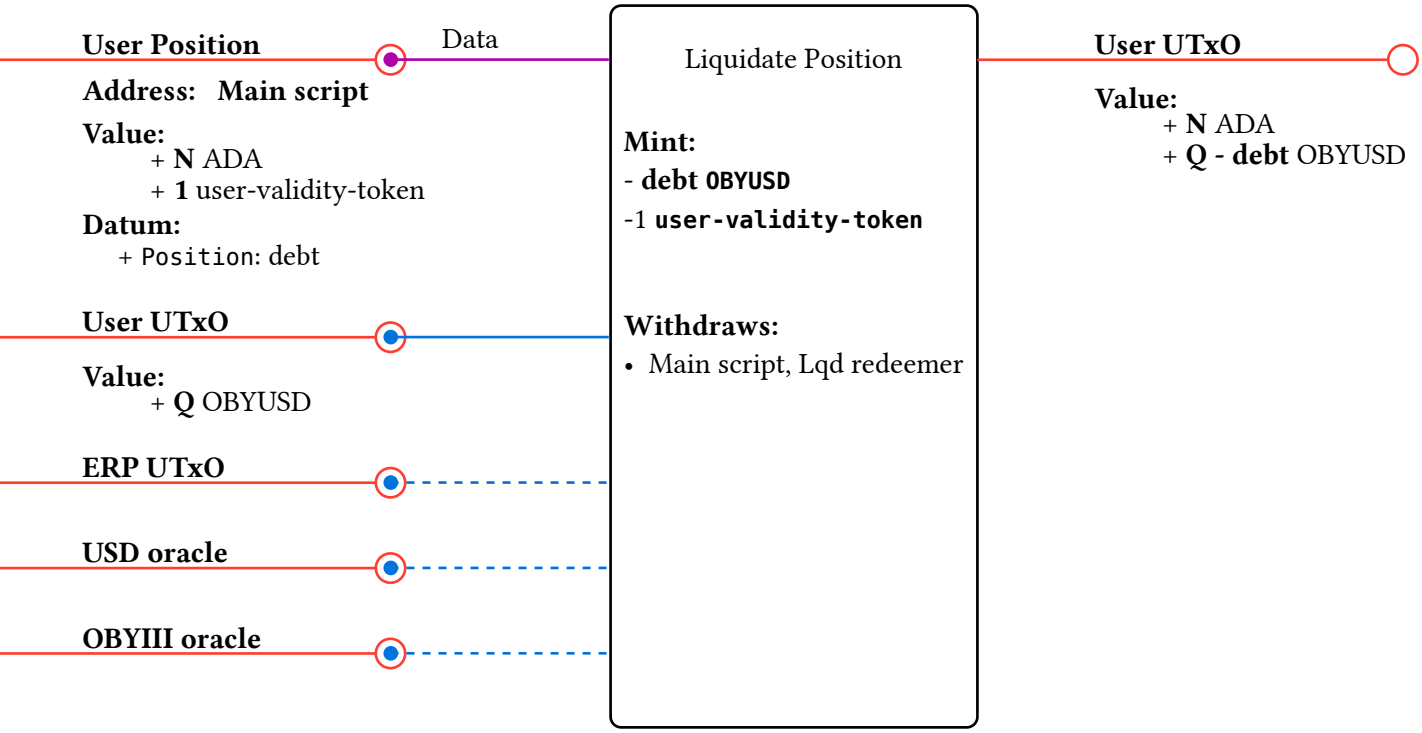


Figure 5: Liquidate Position transaction

2.c.a.f - Liquidate Position with Treasury assistance

The user liquidates a position that is below the Critical Collateral Rate. He burns OBYUSD to pay the debt and takes all the collateral. As compensation, he takes from treasuries an amount of OBYII tokens an equivalent of the for the missing collateral (according to current exchange rate OBYII/ADA).

Involved redeemers:

- `Data`, `Spend` purpose: for spending the Position UTxO
- `Data`, `Spend` purpose: one for each Treasury UTxO spent
- `Lwc`, `WithdrawFrom` purpose: for Main script withdrawal
- `Data`, `Mint` purpose: for burning the Position's user validity token and burning OBYUSDs

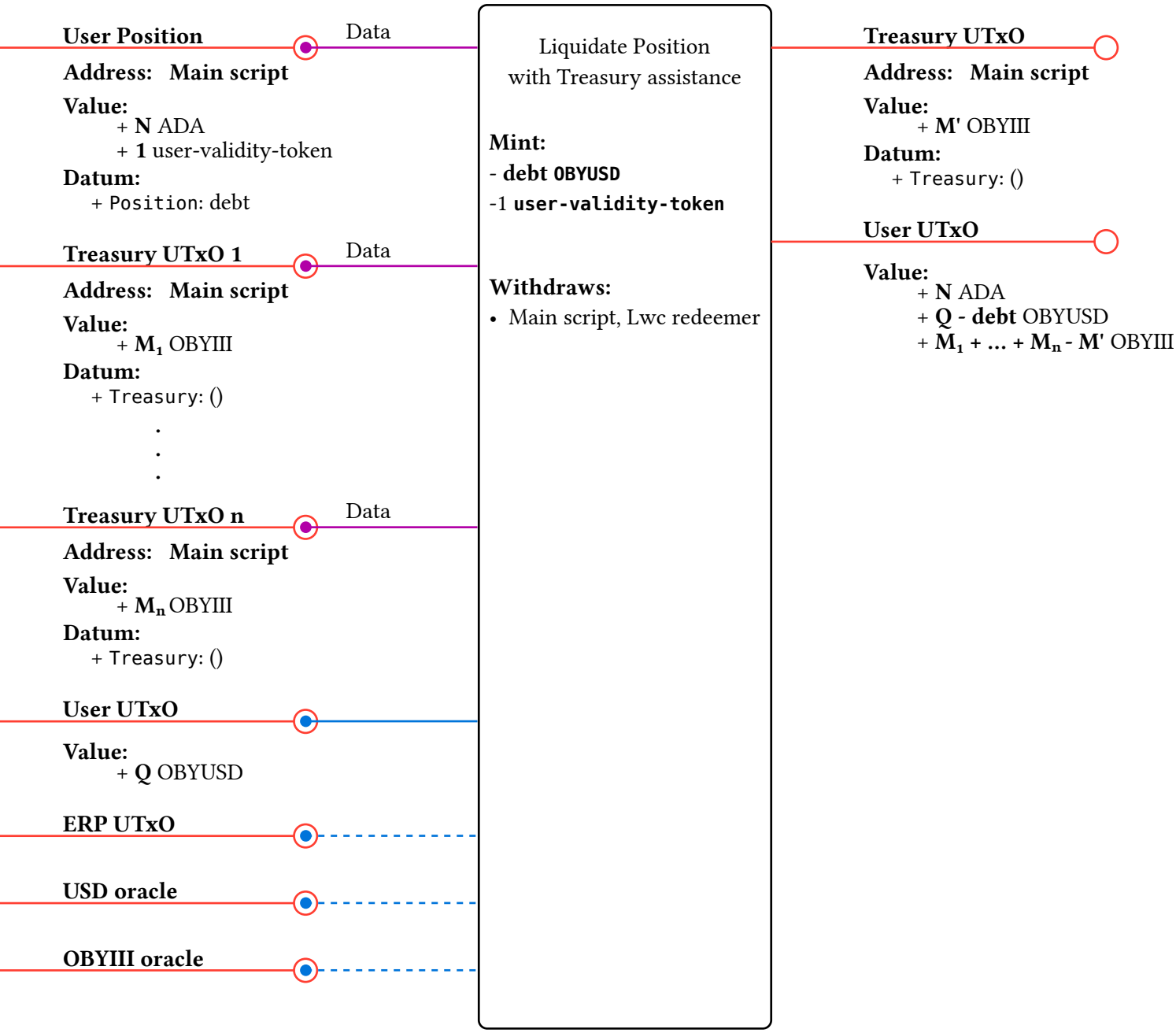


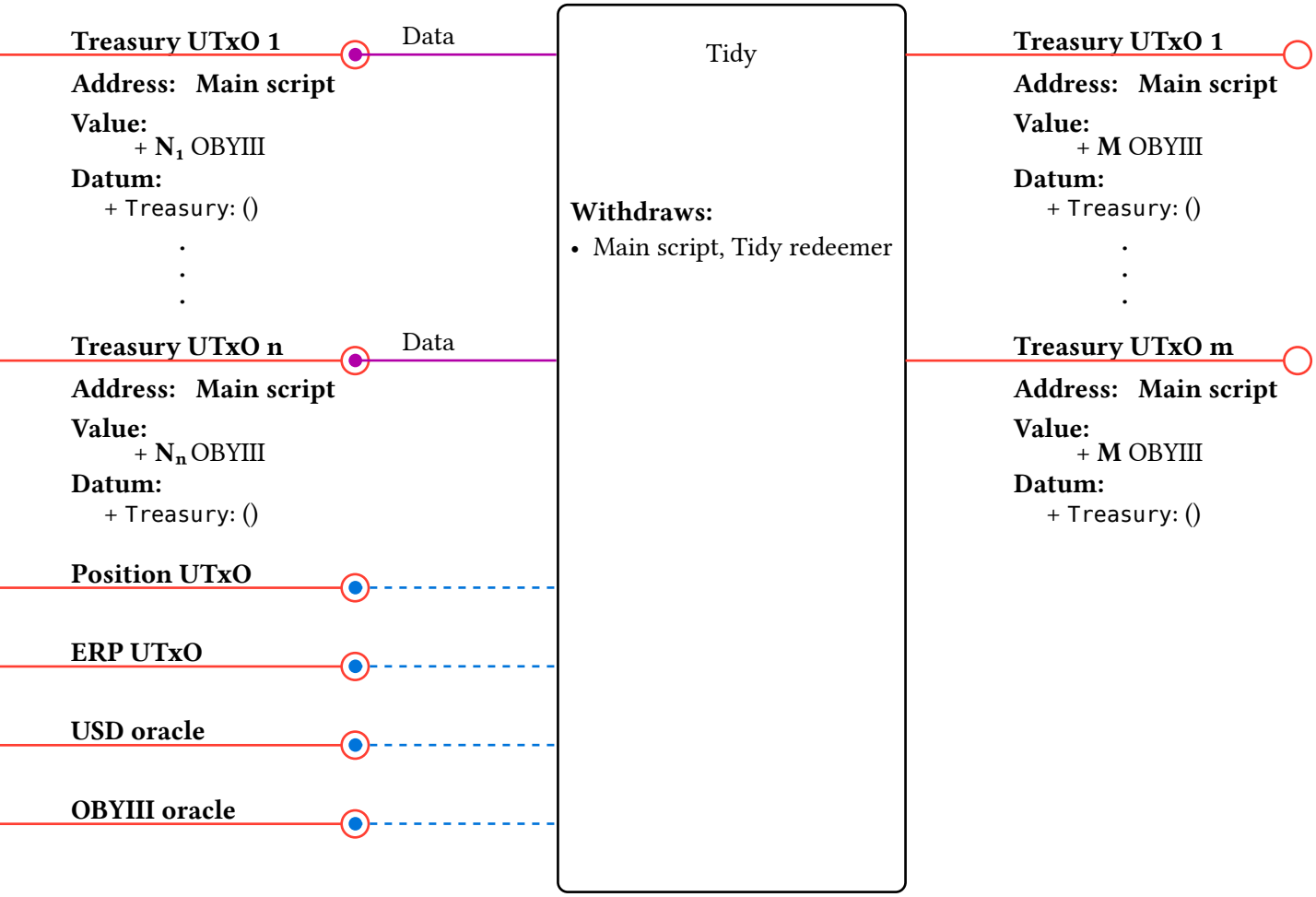
Figure 6: Liquidate Position with Treasury assistance transaction

2.c.a.g - Tidy Treasury

A user collapses Treasury UTxOs, meaning there must be more Treasury inputs than outputs. A Position below the CCR must be included as reference input.

Involved redeemers:

- `Data`, `Spend` purpose: one for each Treasury UTxO spent
- `Tidy`, `WithdrawFrom` purpose: for Main script withdrawal



Note:
n ≥ m
$N_1 + ... + N_n - m < M*m$
Referenced Position Input is below CCR

Figure 7: Tidy transaction

2.c.a.h - Burn Auth token

User burns an auth token. Useful when the related position has been liquidated.

Involved redeemers:

- `Data`, `Spend` purpose: one for each Treasury UTxO spent
- `Burn`, `WithdrawFrom` purpose: for Main script withdrawal

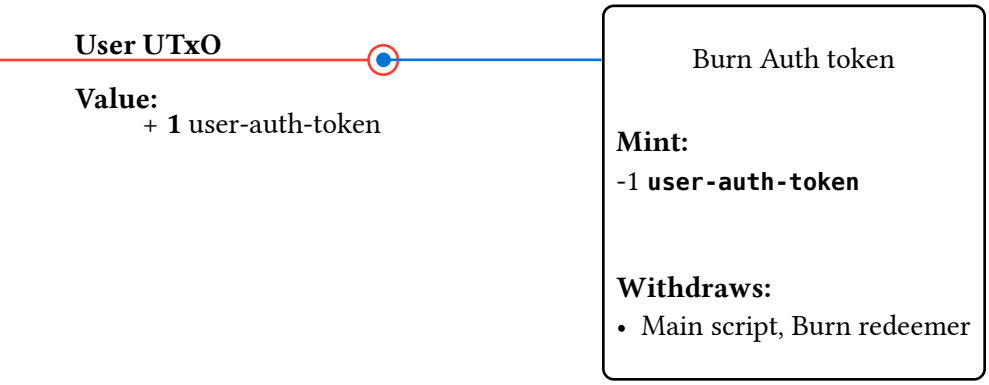


Figure 8: Burn Auth token transaction

2.c.b - Admin

2.c.b.a - Mint OBYIII

The Admin initially mints the total supply of OBYIII. It is free of distributing it as pleased.

Since the minting needs for this functionality are very simple, it could be archieved with either a very simple Plutus script or a Native script.

2.c.b.b - Init ERP

Admin initializes the Exchange Rate Pointer UTxO (ERP) that will hold in its datum the current Exchange Rate (ER) script hash. This UTxO is identified by a validity token and the admin is allowed to perform operations over it because it holds the related auth token.

This operation is ensured to happen only once for each script hash by having a output reference as seed as the validator parameter.

Involved redeemers:

- `Erp2Init`, `Minting` purpose: for minting ERP validity and auth tokens

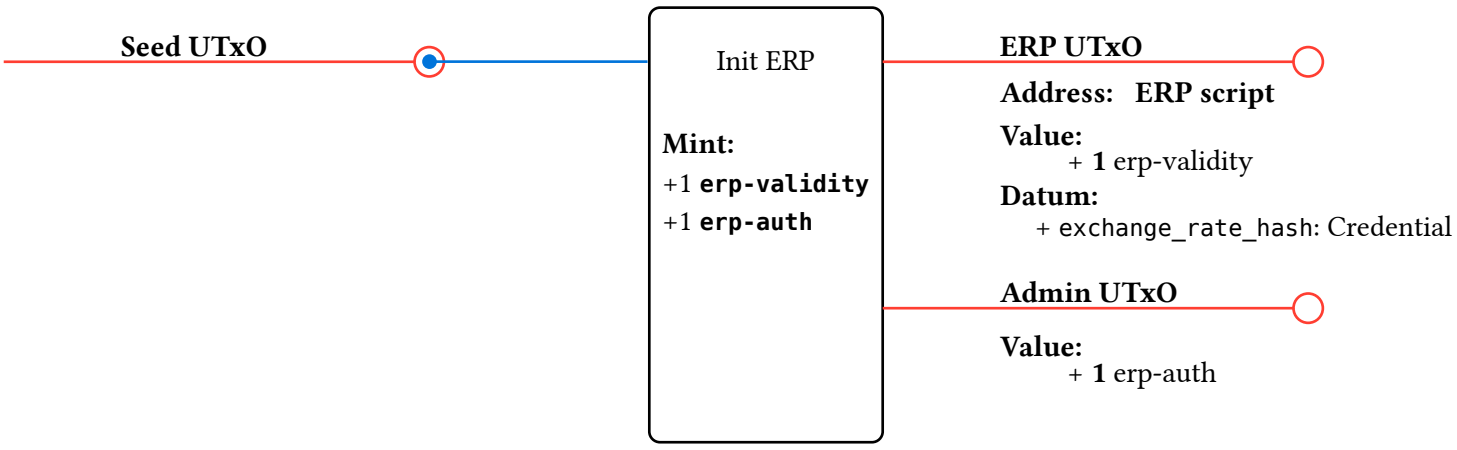


Figure 9: Init ERP transaction

2.c.b.c - Update ERP

Admin updates the ER script hash from the ERP UTxO datum.

Involved redeemers:

- `Erp3Update`: for spending the ERP UTxO

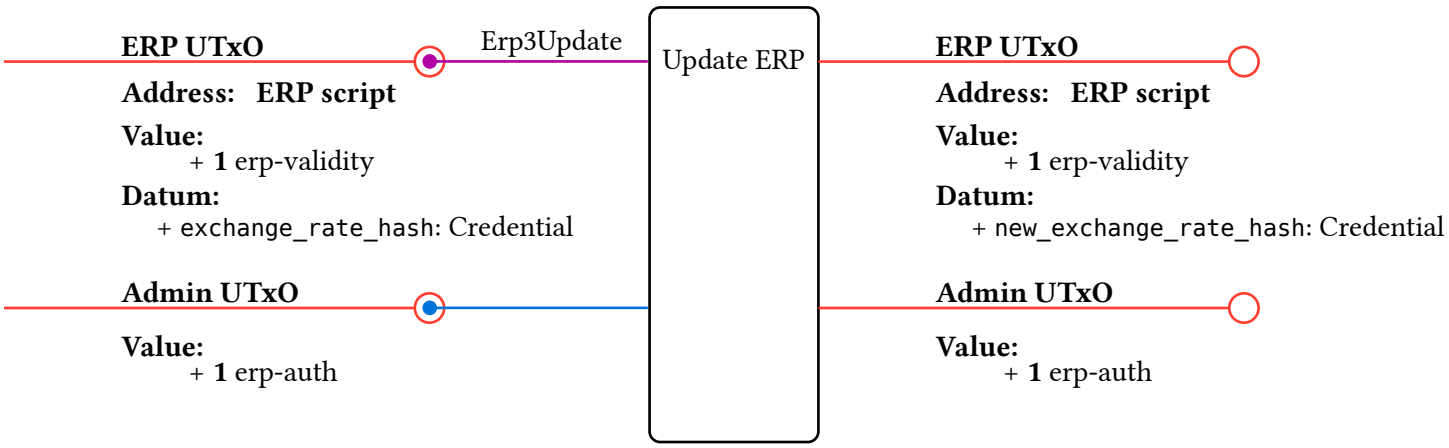


Figure 10: Update ERP transaction

2.c.b.d - Close ERP

Admin closes the ERP UTxO by burning both validity and auth ERP tokens.

Involved redeemers:

- `Erp3Close`: for spending the ERP UTxO
- `Erp2Burn`: for burning both ERP validity and auth tokens

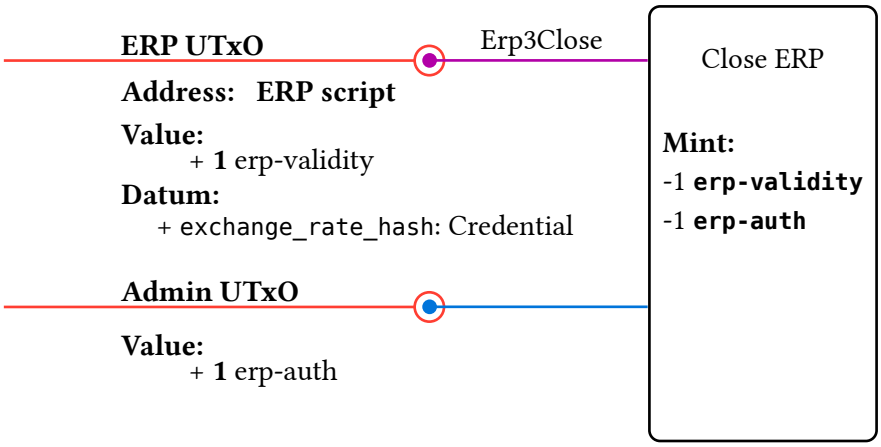


Figure 11: Close ERP transaction

3 - Audited Files

Below is a list of all files audited in this report, any files **not** listed here were **not** audited. The final state of the files for the purposes of this report is considered to be commit `7ee24ca4044363fab104452e9ebd197596c77654`.

Filename
./aik/validators/m.ak
./aik/validators/erp.ak
./aik/validators/er.ak
./aik/validators/er_em.ak
./aik/validators/om.ak
./aik/validators/er_sig.ak
./aik/lib/obymare/tokens.ak
./aik/lib/obymare/utils.ak
./aik/lib/obymare/types.ak
./aik/lib/obymare/constants.ak

4 - Findings

ID	Title	Severity	Status
OBY-001	`Tidy`: Treasury outputs value not validated	Critical	Resolved
OBY-101	`Redeem`: Output position can have negative debt	Major	Resolved
OBY-102	Prevent inclusion of reference scripts	Major	Resolved
OBY-103	Missing conversion in sufficient fee validation	Major	Resolved
OBY-201	Forbid Treasury UTxOs with staking credential	Minor	Resolved
OBY-202	`Tidy`: restict Treasury outputs conditions	Minor	Resolved

5 - OBY-001 `Tidy`: Treasury outputs value not validated

Category	Commit	Severity	Status
Vulnerability	287e40e74c75c2fb94e12875bcaf9db36e387929	Critical	Resolved

5.a - Description

In the Tidy transactions there is no validation for the composition of the entire values paid to the Treasury outputs. This opens up the possibility of an attacker to fill Treasury outputs with trash tokens, which hinder other users of the protocol by letting them the hazards of dealing with them.

5.b - Recommendation

Check that Treasury outputs contain only ADA and OBYIII in their value.

5.c - Resolution

Resolved in commit `3d5d47f`

6 - OBY-101 `Redeem`: Output position can have negative debt

Category	Commit	Severity	Status
Bug	287e40e74c75c2fb94e12875bcaf9db36e387929	Major	Resolved

6.a - Description

In the `Redeem` transaction it is possible to output a position with negative debt by overpaying it. Specifically, this is the position that can be selected to have non-zero debt. Validation only checks that the debt is decreased, so it admits overpayment.

We couldn't find any harm that this feature can do to the protocol or the users, but it deviates significantly from the protocol specification where there is no mention to negative debt positions. As this is an unspecified and probably unexpected behaviour, we strongly recommend its resolution.

6.b - Recommendation

Add a check here to ensure that debt must be greater or equal than zero.

6.c - Resolution

Resolved in PR #8.

7 - OBY-102 Prevent inclusion of reference scripts

Category	Commit	Severity	Status
Vulnerability	287e40e74c75c2fb94e12875bcaf9db36e387929	Major	Resolved

7.a - Description

With the addition of the `minFeeRefScriptsCoinsPerByte` protocol parameter in the upcoming Conway era, including a reference script in any input (whether it's a reference or not) will impact the transaction fees, regardless of whether the script is executed.

Given that the reference script field is not validated in any output of the protocol, there's an attack vector where a malicious party includes a huge reference script in every output of a transaction, costing more fees to the next party interacting with those UTxOs.

7.b - Recommendation

We recommend ensuring that any UTxO belonging to the protocol does not include a reference script.

7.c - Resolution

Resolved in commit `3d5d47f`.

8 - OBY-103 Missing conversion in sufficient fee validation

Category	Commit	Severity	Status
Bug	d6ce554a8e7372ecf8e5d29be1101f2bf97df266	Major	Resolved

8.a - Description

The sufficient fee check is:

```
fr_denom * o_num * fee - fr_num * o_denom * debt >= 0
```

This check can be read as:

```
O * fee >= FR * debt
```

where `O` is the OBYIII/ADA exchange rate, `fee` is in OBYIII and `debt` is in OBYUSD. The left hand side of the inequality is in ADA units, while the right hand side is in OBYUSD units. Therefore the comparison is between incompatible units. The right hand side should be converted to ADA. The correct check should be:

```
O * fee >= FR * U * debt
```

where `U` is the OBYUSD/ADA exchange rate.

8.b - Recommendation

Add `u_num` and `u_denom` as parameters of the function `sufficient_fee` and use them to perform the missing conversion from OBYUSD to ADA.

8.c - Resolution

Resolved in PR #11.

9 - OBY-201 Forbid Treasury UTxOs with staking credential

Category	Commit	Severity	Status
Design	287e40e74c75c2fb94e12875bcaf9db36e387929	Minor	Resolved

9.a - Description

Staking credential of Treasury UTxOs are left to their creators. Moreover, when users perform Tidy operations, they can modify the staking credential of Treasury UTxOs created by someone else. As Treasury UTxOs are not meant to have high amounts of ADAs, their staking is not an interesting feature for the protocol. To discourage unnecessary activity over the Treasury UTxO set, they should be enforced to have an empty staking credential.

9.b - Recommendation

Ensure that in all transactions where Treasury outputs are created the staking credential is set to `None`. This also has an interesting benefit in discoverability: all Treasury UTxOs will have the same address.

9.c - Resolution

Resolved in commit `3d5d47f`.

10 - OBY-202 `Tidy`: restict Treasury outputs conditions

Category	Commit	Severity	Status
Bug	287e40e74c75c2fb94e12875bcaf9db36e387929	Minor	Resolved

10.a - Description

The purpose of `Tidy` is to collapse Treasury UTxOs for Liquidation with assistance on very large positions, and is useful for normalizing the UTxO set too.

However, the validation is too lax: it allows not only collapsing but dispersion of the Treasury UTxO set, which goes against the protocol specification for the functionality.

10.b - Recommendation

Don't allow Treasury UTxOs dispersion.

10.c - Resolution

Resolved in commit `3d5d47f`.

11 - Appendix

11.a - Terms and Conditions of the Commercial Agreement

11.a.a - Confidentiality

Both parties agree, within a framework of trust, to discretion and confidentiality in handling the business. This report cannot be shared, referred to, altered, or relied upon by any third party without Txpipe LLC, 651 N Broad St, Suite 201, Middletown registered at the county of New Castle, written consent.

The violation of the aforementioned, as stated supra, shall empower TxPipe to pursue all of its rights and claims in accordance with the provisions outlined in Title 6, Subtitle 2, Chapter 20 of the Delaware Code titled "Trade Secrets,", and to also invoke any other applicable law that protects or upholds these rights.

Therefore, in the event of any harm inflicted upon the company's reputation or resulting from the misappropriation of trade secrets, the company hereby reserves the right to initiate legal action against the contractor for the actual losses incurred due to misappropriation, as well as for any unjust enrichment resulting from misappropriation that has not been accounted for in the calculation of actual losses.

11.a.b - Service Extension and Details

This report does not endorse or disapprove any specific project, team, code, technology, asset or similar. It provides no warranty or guarantee about the quality or nature of the technology/code analyzed.

This agreement does not authorize the client Big Blymp to make use of the logo, name, or any other unauthorized reference to Txpipe LLC, except upon express authorization from the company.

TxPipe LLC shall not be liable for any use or damages suffered by the client or third-party agents, nor for any damages caused by them to third parties. The sole purpose of this commercial agreement is the delivery of what has been agreed upon. The company shall be exempt from any matters not expressly covered within the contract, with the client bearing sole responsibility for any uses or damages that may arise.

Any claims against the company under the aforementioned terms shall be dismissed, and the client may be held accountable for damages to reputation or costs resulting from non-compliance with the aforementioned provisions. **This report provides general information and is not intended to constitute financial, investment, tax, legal, regulatory, or any other form of advice.**

Any conflict or controversy arising under this commercial agreement or subsequent agreements shall be resolved in good faith between the parties. If such negotiations do not result in a conventional agreement, the parties agree to submit disputes to the courts of Delaware and to the laws of that jurisdiction under the powers conferred by the Delaware Code, TITLE 6, SUBTITLE I, ARTICLE 1, Part 3 § 1-301. and Title 6, SUBTITLE II, chapter 27 §2708.

11.a.c - Disclaimer

The audit constitutes a comprehensive examination and assessment as of the date of report submission. The company expressly disclaims any certification or endorsement regarding the subsequent performance, effectiveness, or efficiency of the contracted entity, post-report delivery, whether resulting from modification, alteration, malfeasance, or negligence by any third party external to the company.

The company explicitly disclaims any responsibility for reviewing or certifying transactions occurring between the client and third parties, including the purchase or sale of products and services.

This report is strictly provided for ***informational purposes*** and reflects solely the due diligence conducted on the following files and their corresponding hashes using sha256 algorithm:

Filename: ./aik/validators/m.ak **Hash**: b740064f8ce607c3c1f415ec98de44e1a46042d690c0abca985c9b7328fd61d0
Filename: ./aik/validators/erp.ak **Hash**: d5ac99e6fe0d98899cd69ddebf58a11666c4843acce5cc0604b219e857223afc
Filename: ./aik/validators/er.ak **Hash**: 367c316ea17c387caa10a8a4e22a1bcb47a57769dbb43cee614a44b7735d0ce7
Filename: ./aik/validators/er_em.ak **Hash**: 99930abc5d116f1594d496ea5c04e31e5ec5a064b3dfeb49794c93db54bfff8e
Filename: ./aik/validators/om.ak **Hash**: f4ec3e6f342b566d37df12ca3a12a676beac54bb5d712af5121d138c8bdd2daf
Filename: ./aik/validators/er_sig.ak **Hash**: cf707387a399c1abf142ebb1682511c4d013e223a37334ff7efa5a113de78366
Filename: ./aik/lib/obymare/tokens.ak **Hash**: 406b07448d8c6ccd13cd046adbb1723275b3d720f7901031d62a65f94b0b7c21
Filename: ./aik/lib/obymare/utils.ak **Hash**: db859b1242d1ff1f57f22522fc19467ac9e0d6c155e1982ee92aa86c22f8ca40
Filename: ./aik/lib/obymare/types.ak **Hash**: 7f26bbff05c0469d9b384ec2cbd6936449fe07205d890a17a06af16d1339d549
Filename: ./aik/lib/obymare/constants.ak **Hash**: b45950faacf1f00a29fc8bd8c938960fbe7e2e4291b67a91128ec6e09c6b92a6

TxPipe advocates for the implementation of multiple independent audits, a publicly accessible bug bounty program, and continuous security auditing and monitoring. Despite the diligent manual review processes, the potential for errors exists. TxPipe strongly advises seeking multiple independent opinions on critical matters. It is the firm belief of TxPipe that every entity and individual is responsible for conducting their own due diligence and maintaining ongoing security measures.

11.b - Issue Guide

11.b.a - Severity

Severity	Description
Critical	Critical issues highlight exploits, bugs, loss of funds, or other vulnerabilities that prevent the dApp from working as intended. These issues have no workaround.
Major	Major issues highlight exploits, bugs, or other vulnerabilities that cause unexpected transaction failures or may be used to trick general users of the dApp. dApps with Major issues may still be functional.
Minor	Minor issues highlight edge cases where a user can purposefully use the dApp in a non-incentivized way and often lead to a disadvantage for the user.
Info	Info are not issues. These are just pieces of information that are beneficial to the dApp creator. These are not necessarily acted on or have a resolution, they are logged for the completeness of the audit.

11.b.b - Status

Status	Description
Resolved	Issues that have been **fixed** by the **project** team.
Acknowledged	Issues that have been **acknowledged** or **partially fixed** by the **project** team. Projects can decide to not **fix** issues for whatever reason.
Identified	Issues that have been **identified** by the **audit** team. These are waiting for a response from the **project** team.

11.c - Revisions

This report was created using a git based workflow. All changes are tracked in a github repo and the report is produced using typst. The report source is available here. All versions with downloadable PDFs can be found on the releases page.

11.d - About Us

TxPipe is a blockchain technology company responsible for many projects that are now a critical part of the Cardano ecosystem. Our team built Oura, Scrolls, Pallas, Demeter, and we're the original home of Aiken. We're passionate about making tools that make it easier to build on Cardano. We believe that blockchain adoption can be accelerated by improving developer experience. We develop blockchain tools, leveraging the open-source community and its methodologies.

11.d.a - Links

- Website
- Email
- Twitter