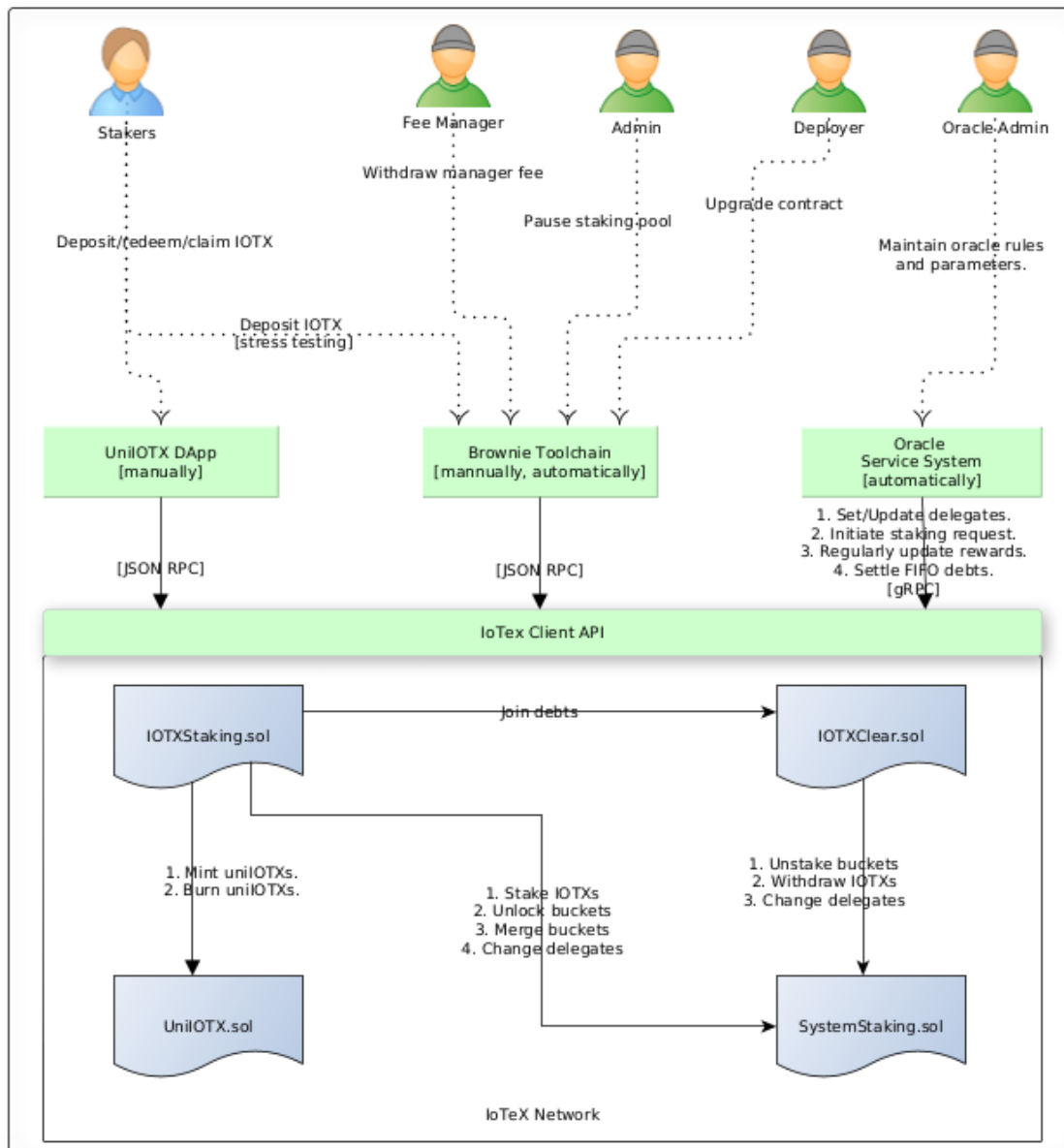


Scheme for Testing Bedrock Liquid Staking (IoTeX)

• 1 Introduction



Test Scope of the System of Bedrock Liquid Staking (IoTeX)

To ensure the successful launch of our product, the Bedrockx Liquid Staking system on the IoTeX Mainnet, it is crucial for both the Rockx team and the IoTeX team to collaborate and conduct comprehensive tests. This joint effort will guarantee the quality of our project during the production phase.

This document combines the use of both End-to-End Tests and Stress Tests to achieve software quality. These two testing approaches will address software correctness and robustness, respectively.

End-to-End Test: Through End-to-End testing, we will evaluate the system's overall functionality and adherence to business rules, ensuring it operates correctly and meets user expectations.

Stress Test: Stress testing involves subjecting the system to excessive load in order to determine its performance limits and identify any potential breaking points or failure.

• 2 Preparation

There are several important things to prepare in order to conduct the testing suite designed by this document. These include, but are not limited to, system initialization, testing environment configuration, accounts and funds, etc.

• 2.1 Accounts Funded

Accounts and Funds							
Account	Controller	Funder	Required Fund (IOTX)	Required Fund (USD)	Total Required Fund (IOTX)	Total Required Fund (USD)	Expected Final Principal State After Test
Staker1	IoTeX Team	ToTeX Foundation	1,000,000	17,280	15,000,000	259,200	Redeemed and claimed back to the original accounts
Staker2			1,000,000	17,280			
Staker3			10,000,000	172,800			
Staker4			1,000,000	17,280			
Staker5			1,000,000	17,280			
Staker6			1,000,000	17,280			
Oracle	Rockx Team	Rockx Financial Department	500	8.64	620	10.8	Used dynamically as a gas fee for the maintenance of the staking pool.
Fee manager			120	2			
Admin							
Deployer							
Note: 1. 1 IOTX = 0.01728 USD, as of Oct 10, 2023. 2. The fund estimation for stakers is calculated excluding gas fees.							

A prerequisite for any operation on the IoTeX blockchain is having accounts with sufficient balance. The accounts and required funds that we plan to utilize in the test are listed above.

It is worth noting that the Rockx Team requires support from the IoTeX team, including IOTX funding and assistance with testing operations.

• 2.2 System Setup

Deployment: The UniIOTX DApp, smart contracts, and Oracle service should be deployed on the IoTeX Mainnet.

Initial State:

1. The DApp must accurately interact with the appropriate smart contracts and display precise data, particularly for APY.
2. Smart contracts should have accurate mutual dependencies and proper access control.
3. The Oracle service needs to be properly configured and capable of effectively interacting with the relevant smart contracts when required.
4. The Web3 API endpoint (<https://babel-api.mainnet.iotex.one>) and the gRPC API endpoint (api.iotex.one:443) should be accessible stably.

• 2.3 Brownie Setup

To conduct some of the test suites, we need to rely on the Brownie toolchain. Therefore, it is necessary to set up the Brownie environment beforehand.

Please visit our smart contract repository at <https://github.com/RockX-SG/uniiotx> for a detailed tutorial on Brownie configuration.

To align with this document, the IoTEx Team needs to use the following account name and endpoint for configuring accounts and network settings:

Account Names: Staker2, Staker4, Staker5, Staker6

Web3 API Endpoint: <https://babel-api.mainnet.iotex.one>

You can clone this repository onto your local computer using the following command: `git clone`

`git@github.com:RockX-SG/uniiotx.git` . This will allow you to run the Brownie commands provided in the documentation.

• 2.4 Brownie Scripts

Under the Rockx-SG/uniiotx directory, there should be three scripts available for logic merging and stress testing. These scripts must be developed and tested by the Rockx Team. The filenames of the scripts are as follows:

Script Filenames: merge.py, stress_deposit.py, stress_redeem.py, and stress_claim.py

• 2.5 MetaMask Wallet

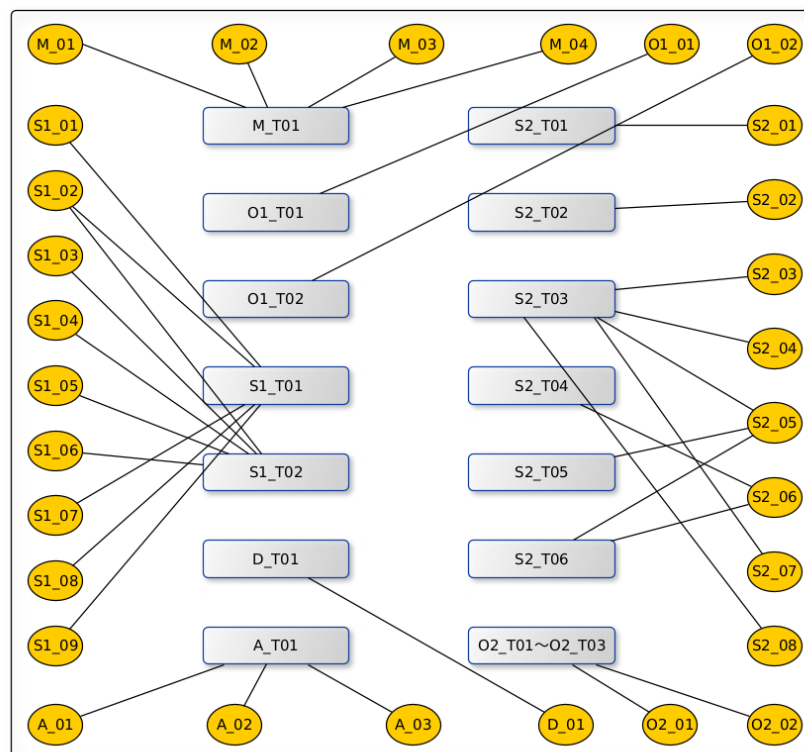
Our UniIOTX DApp, available at <https://app.bedrock.technology/uniiotx>, is designed to integrate with the MetaMask wallet as the default option for end users (stakers) to interact with smart contracts. Therefore, it is necessary for the IoTEx Team to install and configure MetaMask in advance, adding the required accounts. The accounts that need to be used through the DApp/MetaMask are listed below:

Account Names: Staker1, Staker2, Staker3

• 3 Requirements

• 3.1 End-to-End Test

• 3.1.1 Relationship



Correspondence Relation between Test Cases and Test Suites.

This document determines test cases based on business rules and utilizes test suites to execute these test cases. In terms of data modeling, there is a many-to-many relationship between test cases and test suites.

As shown in the picture above, the yellow ellipses represent test cases, while the grey rectangles represent test suites. These test cases and test suites cover almost all the essential functions of our systems.

This picture outlines the correspondence relationship between test cases and test suites. Please continue reading to understand the details of each test case and test suite.

• 3.1.2 Test Cases

End-to-End Test Cases			
Account	Process/ Scenario	Case	Description
Staker1, Staker2	Depositing IOTX	S1_01	The corresponding amount of uniIOTX should be minted appropriately.
		S1_02	Each type of bucket (10000/100000/1000000 IOTX) should be active.
		S1_03	Ten 10000-buckets should be merged into a 100000-bucket.
		S1_04	Ten 100000-buckets should be merged into a 1000000-bucket.
		S1_05	The greater the amount of IOTX staked, the more rewards will yield.
		S1_06	Pending rewards should be synchronized for upcoming depositing requests.
		S1_07	The value of current reserve should increase accordingly.
		S1_08	The exchange ratio and current reserve should increase with reward yields.
		S1_09	The current exchange ratio should remain invariant for every request.
	Redeeming IOTX	S2_01	Users can redeem IOTX only in units of 1000000.
		S2_02	Redemption should not proceed in case of insufficient uniIOTX allowance.
		S2_03	The corresponding amount of uniIOTX should be burned appropriately.
		S2_04	Debts should be carefully documented and arranged for repayment.
		S2_05	Users should continue to earn rewards which can be claimed at any time.
		S2_06	Users can claim the principal only after the debt has been repaid.
		S2_07	The value of current reserve should decrease accordingly.
		S2_08	The current exchange ratio should remain invariant for every request.
Fee manager	Withdrawing manager fee	M_01	The manager's withdrawn fee should be reinvested.
		M_02	An equivalent amount of uniIOTX should be minted appropriately.
		M_03	The value of current reserve should increase accordingly.
		M_04	The current exchange ratio should remain invariant for every request.
Oracle	Updating rewards	O1_01	The yield rewards should be synchronized and divided on a daily basis.
	Sending staking request	O1_02	The staking request should be sent when there is over 10,000 pending IOTX.
Oracle, Staker3	Setting global delegate	O2_01	The global delegate should be replaced if a better candidate is identified.
	Updating delegate	O2_02	The delegate should be updated once it fails to guarantee the reward yield
Admin	Pausing staking pool	A_01	Users should be prohibited from making deposits when it is paused.
		A_02	Users should be prevented from redeeming when it is paused.
		A_03	Users should be prevented from claiming when it is paused.
Deployer	Upgrading contract	D_01	The contract upgrade should not introduce any additional risks. It must uphold the integrity, consistency, and access control.

The table above organizes key business rules based on business processes/scenarios into test cases, which regulate the behavior of our systems.

Please note:

1. Currently, our UniIOTX DApp does not support the withdrawal of the manager fee. Additionally, we do not have a backend service in place to facilitate this action.
2. Currently, our Oracle service does not have the capability to automatically trigger staking requests when there are over 10,000 pending IOTX.
3. To change the delegate rank, a significant deposit of IOTX is necessary. In such cases, the Oracle service oversees the change and initiates the delegate update in our staking pool.

• 3.1.3 Test Suits

End-to-End Test Suits				
Suit	Account	Description	Network	Method
S1_T01	Staker1	Deposit 1,000,000 IOTX at once.	Mainnet	UniIOTX DApp
S1_T02	Staker2	Submit 19 deposit requests, which includes ten deposits of 10,000 IOTX and nine deposits of 100,000 IOTX. The accumulated deposit requests will ultimately lead to two merging operations in the staking pool. To initiate multiple deposit requests automatically, please execute the following command: `brownie run scripts/test/ merge.py --network=iotex-mainnet`	Mainnet	Brownie Scripts
S2_T01	Staker1	Submit one redemption request which is expected to fail for fewer than 1,000,000 IOTX.	Mainnet Fork	Brownie Console
S2_T02		Submit one redemption request which is expected to fail without sufficient uniIOTX allowance.		
S2_T03	Staker1, Staker2	Submit one redemption request of 1,000,000 IOTX for both Staker1 and Staker2, respectively	Mainnet	UniIOTX DApp
S2_T04	Staker1	Submit one claim request which is expected to fail for 1,000,000 IOTX before the debt repayment is done.	Mainnet Fork	Brownie Console
S2_T05	Staker1,	Submit one claim request for the current rewards for both Staker1 and Staker2, respectively	Mainnet	UniIOTX DApp
S2_T06	Staker2	Submit one claim request for the principal and the remaining rewards for both Staker1 and Staker2, respectively.		
M_T01	Fee Manager	Periodically withdraw the manager's fee.		Brownie Console
O1_T01	Oracle	Automatically upate staking rewards on a daily basis.		Oracle Service
O1_T02		Automatically initiate a staking request if the total pending value reaches 10,000 IOTX.		
O2_T01	Staker3, Oracle	Deposit 10,000,000 IOTX at once and Oracle will trigger a delegate update.		UniIOTX DApp, Oracle Service
O2_T02	Staker3	Redeem 10,000,000 IOTX for Staker3		UniIOTX DApp
O2_T03		Claim all the principal and the rewards for Staker3.		
A_T01	Admin, Staker1	Pause the staking pool and submit requests for depositing, redeeming, and claiming, which are expected to fail.	Mainnet Fork	Brownie Console
D_T01	Deployer, Admin	Upgrade the contract and ensure the integrity, consistency, and access control of all contracts.	Mainnet	
Note: 1. The commands should be executed on a CLI terminal from the Rockx-SG/uniIOTX directory.				

The table above organizes closely related test cases into different test suites, which serve as the testing units for later scheduling.

• 3.2 Stress Test

Stres Test Suits

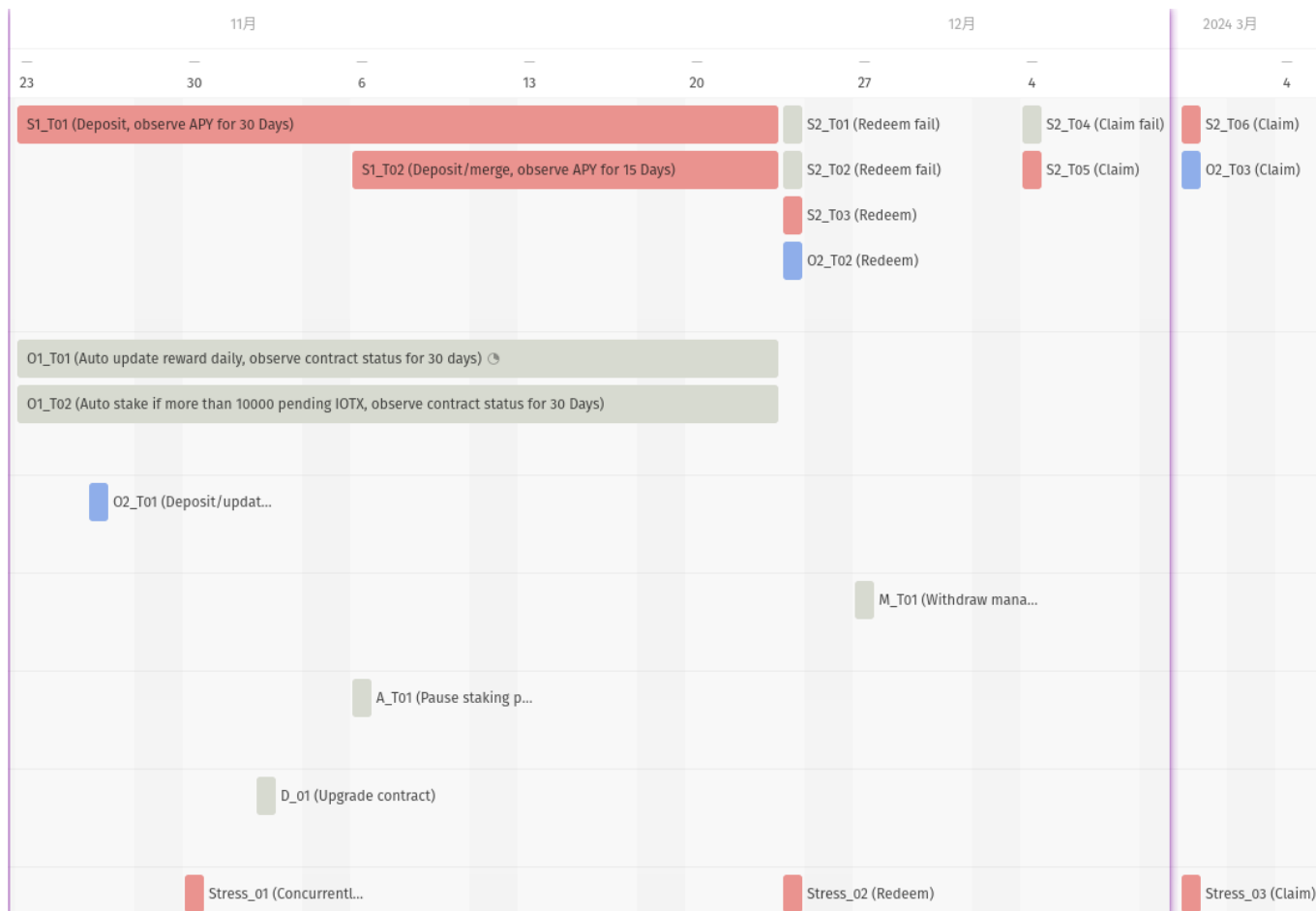
Suit	Account	Description	Network	Method
Stress_01	Staker4, Staker5, Staker6	Three users will issue deposit requests simultaneously and continuously. Each user will submit 100 requests, with each request depositing 10,000 IOTX. The estimated duration of the entire process is approximately 10 minutes. In case any issues arise, the requests should be stopped midway. When an staker initiates IOTXStaking.deposit(), it activates UniIOTX.mint(), SystemStaking.stake(), and SystemStaking.merge(), provided the specified conditions are fulfilled. To initiate multiple deposit requests simultaneously, please execute the following command: `brownie run scripts/test/stress_deposit.py --network=iotex-mainnet`.	Mainnet	Brownie Scripts
Stress_02		Redeem 1,000,000 IOTX for Staker4, Staker5, and Staker6, respectively. When an staker invokes IOTXStaking.redeem(), it sequentially triggers UniIOTX.burnFrom(), SystemStaking.unlock(), SystemStaking.safeTransferFrom(), and IOTXClear.joinDebt() To initiate multiple redeem requests simultaneously, please execute the following command: `brownie run scripts/test/stress_redeem.py --network=iotex-mainnet`.	Mainnet	
Stress_03		Claim all the principal and the rewards for Staker4, Staker5, and Staker6, respectively. When a staker invokes the IOTXClear.claim() function, the specified value of IOTX will be transferred from the IOTXClear contract to the designated recipient. To initiate multiple claim requests simultaneously, please execute the following command: `brownie run scripts/test/stress_claim.py --network=iotex-mainnet`.	Mainnet	
Note: 1. The commands should be executed on a CLI terminal from the Rockx-SG/uniiotx directory.				

The table above outlines stress test suites designed to estimate the robustness of our system.

• 4 Scheduling

Test Scheduling

Suit	Account	Short Description	IOTX Required	Test Time	Operator
S1_T01	Staker1	Deposit, observe APY for 30 Days	1,000,000	2023/10/23~2013/11/23	IoTeX Team
S1_T02	Staker2	Deposit and merge, observe APY for 15 Days	1,000,000	2023/11/6~2013/11/23	
S2_T01	Staker1	Redeem fail	0	2023/11/24	Rockx Team
S2_T02		Redeem fail	0		
S2_T03	Staker1, Staker2	Redeem	GasFee		IoTeX Team
S2_T04	Staker1	Claim fail	0	2023/12/4	Rockx Team
S2_T05	Staker1, Staker2	Claim	GasFee		IoTeX Team
S2_T06		Claim			
M_T01	Fee Manager	Withdraw manager fee			2023/11/27
O1_T01	Oracle	Auto update reward daily, observe contract status for 30 days	GasFee	2023/10/23~2013/11/23	Rockx Team
O1_T02		Auto stake if over 10000 pending IOTX, observe contract status for 30 Days			Rockx Team
O2_T01	Staker3, Oracle	Deposit and update delegate	10,000,000	2023/10/26~2013/10/27	IoTeX Team, Rockx Team
O2_T02	Staker3	Redeem	GasFee	2023/11/24	IoTeX Team
O2_T03		Claim		2024/2/29	
A_T01	Admin, Staker1	Pause staking pool	0	2023/11/6	Rockx Team
D_T01	Deployer, Admin	Upgrade contract	GasFee	2023/11/2	
Stress_01	Staker4~Staker6	Concurrently deposit IOTX	3,000,000	2023/10/30	IoTeX Team
Stress_02		Redeem	GasFee	2023/11/24	
Stress_03		Claim		2024/2/29	
Note: 1. The fund estimation for stakers is calculated excluding gas fees.					



To ensure the orderly and systematic execution of all designed test suites, we have created the schedule outlined above. Both the [Rockx Team](#) and the [IoTeX team](#) need to closely collaborate according to this schedule.

We plan to start the test on October 23, 2023. The testing duration will last until February 29, 2024. However, most of the testing jobs can be completed within 45 days, by December 4, 2023, once the test begins.

The main reason for the prolonged duration of the entire test is the 94-day duration of the unstaking phase, as per our business rule.

• 5 Implementation

To facilitate efficient collaboration during the testing scheme, we have created a Google Spreadsheet. You can access it at <https://docs.google.com/spreadsheets/d/1VWB7llmJ2EuJBkqgxT7okx2yuHUNh-GL495zZNHTZyQ/edit?usp=sharing>.


Both teams can rely on this spreadsheet to track testing progress, capture staking statuses, and record any problems.

• 5.1 Progress Tracking

Table for Test Progress Tracking					
Suit	Operator	Start Time	End Time	Short Description	Progress
S1_T01	IoTeX Team	2023/10/23	2013/11/23	Deposit, observe APY for 30 Days	Pending ▼
O1_T01	Rockx Team	2023/10/23	2013/11/23	Auto update reward daily, observe contract status for 30 days	Pending ▼
O1_T02	Rockx Team	2023/10/23	2013/11/23	Auto stake if over 10000 pending IOTX, observe contract status for 30 Days	Pending ▼
O2_T01	IoTeX Team, Rockx Team	2023/10/26	2013/10/27	Deposit and update delegate	Pending ▼
Stress_01	IoTeX Team	2023/10/30	2023/10/30	Concurrently deposit IOTX	Pending ▼
D_T01	Rockx Team	2023/11/2	2023/11/2	Upgrade contract	Pending ▼
S1_T02	IoTeX Team	2023/11/6	2013/11/23	Deposit and merge, observe APY for 15 Days	Pending ▼
A_T01	Rockx Team	2023/11/6	2023/11/6	Pause staking pool	Pending ▼
S2_T01	Rockx Team	2023/11/24	2023/11/24	Redeem fail	Pending ▼
S2_T02	Rockx Team	2023/11/24	2023/11/24	Redeem fail	Pending ▼
S2_T03	IoTeX Team	2023/11/24	2023/11/24	Redeem	Pending ▼
O2_T02	IoTeX Team	2023/11/24	2023/11/24	Redeem	Pending ▼
Stress_02	IoTeX Team	2023/11/24	2023/11/24	Redeem	Pending ▼
M_T01	Rockx Team	2023/11/27	2023/11/27	Withdraw manager fee	Pending ▼
S2_T04	Rockx Team	2023/12/4	2023/12/4	Claim fail	Pending ▼
S2_T05	IoTeX Team	2023/12/4	2023/12/4	Claim	Pending ▼
S2_T06	IoTeX Team	2024/2/29	2024/2/29	Claim	Pending ▼
O2_T03	IoTeX Team	2024/2/29	2024/2/29	Claim	Pending ▼
Stress_03	IoTeX Team	2024/2/29	2024/2/29	Claim	Pending ▼
Note: 1. Please update the Progress pull-down menu to reflect the status as either Pending, Ongoing, Done, or Failed. 2. If a test suite fails, please provide details regarding the corresponding problem. 3. Additionally, remember to record account status and staking statistics throughout the entire test lifecycle.					

• 5.2 Status Capturing

Table for Account State and Staking Statistics Capturing							
Operator	Time	Case	Staker1	Staker2	Staker3	Staker4	FeeManager
IoTeX Team	2023/10/23	Before initiating the deposit request for the S1_T01 test suite.					
		After submitting the deposit request for the S1_T01 test suite.					
	2023/10/26	Before initiating the deposit request for the O2_T01 test suite.					
		After submitting the deposit request for the O2_T01 test suite.					
	2023/10/30	Before initiating any deposit request for the Stress_01 test suite.					
		After submitting all the deposit requests for the Stress_01 test suite.					
	2023/11/6	Before initiating any deposit request for the S1_T02 test suite.					
		After submitting all the deposit requests fors the S1_T02 test suite.					
	2023/11/24	Before initiating any redeem request for the S2_T03 and O2_T02 test suites.					
		After submitting all the redeem requests for the S2_T03 and O2_T02 test suite.					
Rockx Team	2023/11/27	Before initiating the manager fee withdrawal request for the M_T01 test suite.					
IoTeX Team		After submitting the manager fee withdrawal request for the M_T01 test suite.					
	2023/12/4	Before initiating any claim request for the S2_T05 test suite.					
		After submitting all the claim requests for the S2_T05 test suite.					
	2024/2/29	Before initiating any claim request for the S2_T06 and o2_T03 test suites.					
		After submitting all the claim requests for the S2_T06 and o2_T03 test suites.					
	<div>Note:</div> <div>1. Please insert the picture captured from our UnilOTX DApp into the blank cell.</div> <div>2. You can find the DApp at https://app.bedrock.technology/uniiotx.</div> <div>3. The data scope that we need to capture is as follows.</div> <div>4. The captured statues are essential and valuable for our testing, analysis, and judgment.</div>						

 0x3af4...2b5375

Available to stake

42.56981 IOTX

Staked amount ⓘ +

0.0 uniIOTX

≈ 0.0 IOTX

Claimable ⓘ

0.0 IOTX

[⬇ Claim](#)

Staking statistics

TOKEN CONTRACT 0x236f8c0a61dA474dB21B693fB2ea7AAB0c803894 ⓘ

Expected APY ⓘ

0.0%

Exchange ratio

1.0 IOTX = 1.0 uniIOTX ↔

Staking pool fee ⓘ

10%

Current reserve


4.0 IOTX

Total IOTX staked

0.0 IOTX

Stake >

Unstake >

 Add to MetaMask

- **5.3 Problem Recording**

Table for Test Problem Recording		
Suit	Operator	Description
S1_T01	IoTeX Team	
O1_T01	Rockx Team	
O1_T02	Rockx Team	
O2_T01	IoTeX Team, Rockx Team	
Stress_01	IoTeX Team	
D_T01	Rockx Team	
S1_T02	IoTeX Team	
A_T01	Rockx Team	
S2_T01	Rockx Team	
S2_T02	Rockx Team	
S2_T03	IoTeX Team	
O2_T02	IoTeX Team	
Stress_02	IoTeX Team	
M_T01	Rockx Team	
S2_T04	Rockx Team	
S2_T05	IoTeX Team	
S2_T06	IoTeX Team	
O2_T03	IoTeX Team	
Stress_03	IoTeX Team	
Note: 1. Please provide a concise and accurate description of the problem, along with necessary evidence such as screenshots, transaction hash, etc. 2. Feel free to expand the columns or rows of this table sheet in case you need to organize multiple problems for a test suite.		

- **Appendixes**

- **A References**

[Whitepaper](#)
[UniiOTX DApp](#)
[Contract Repository](#)
[Google Spreadsheet](#)

- **B Report**

A testing report should be written or linked here after the completion of testing.