RD
AUDITORS

StakeEasy Finance, Smart Contract, Code Review and Security Analysis Report

Customer: StakeEasy Finance
Prepared on: 1st May 2022
Platform: Secret Network
Language: RUST

rdauditors.com

Table of Contents

Disclaimer

This document may contain confidential information about its systems and intellectual property of the customer as well as information about potential vulnerabilities and methods of their exploitation.

The report containing confidential information can be used internally by the customer or it can be disclosed publicly after all vulnerabilities are fixed - upon the decision of the customer.

Document

Name	Smart Contract Code Review and Security Analysis Report of StakeEasy Finance
Platform	Secret/RUST
Folder	Staking_contract
File 1	Admin.rs
MD5 hash	47CA4DBB2B66EC524F364D228A986BA4
SHA256 hash	7967DA4581C2658D36ED176A706C04FA532EDBC45005B33928B47A64EE439E6F
File 2	Airdrop.rs
MD5 hash	3B90D4840E081349C5CB7D184E6880FC
SHA256 hash	EE5B929B5CF2DBA3FAD9D3CEDFBE0580325CCFCBC783120015DE232E37908BB4
File 3	Claim.rs

MD5 hash	01F1306A2BE18E5876039FDD64C1F6C6
SHA256 hash	F3B1DA71A3182CFF6297B4412EFB6452119E8A4C5A5B3B40C1504945C453E5DA
File 4	Contract.rs
MD5 hash	1E44090DD8030D42286C4E91FA81DB1F
SHA256 hash	BA5595331F8553027C447AA590D798EFB37EE0961EC8A5355D979DF282527755
File 5	Deposit.rs
MD5 hash	8CB59BBDEAE2CCFF287639EF6A5D0444
SHA256 hash	C224C6526FD72A417386338034550F1826963ED233D5B95363DA489006AFDCBA
File 6	Lib.rs
MD5 hash	0FD23731EBBEE469E180D3E9DBAD59BE
SHA256 hash	40BC6B99573BC68F2346239BEE17ED354F94D765DFE0B811D69C1FCF134721CF
File 7	Msg.rs

MD5 hash	0FD23731EBBEE469E180D3E9DBAD59BE
SHA256 hash	40BC6B99573BC68F2346239BEE17ED354F94D765DFE0B811D69C1FCF134721CF
File 8	Queries.rs
MD5 hash	E02197418C751FD36D25A1673FDB3F2D
SHA256 hash	91E849683CAE52C70C7D5EE03CCF9BEA384E9CB1FBB22807495BA5FC8AD329AB
File 9	Staking.rs
MD5 hash	29CAF18834832B89DE3BA3996863289B
SHA256 hash	3AE43FAE0A70B61E8738CE9045F90DC12DE786F30BE760A46C02CA6CE0DBDC7F
File 10	State.rs
MD5 hash	D9782A9DDACB8A093F632B166DC9CC0C
SHA256 hash	65188356579B8E0F43E4B5794B1457D08AE1341C7180B79CA76CABACC289BACC

File 11	Token.rs
MD5 hash	49993BA14EEE7C44EB04FDC8025C2BBE
SHA256 hash	E9E578E09F88E893F6C4C31F2CC066DC9A37DB2D5392606523C465A0C7798A8A
File 12	Utils.rs
MD5 hash	80EE45562DDA81B358E247DC78FA9642
SHA256 hash	27EA1BAA303A0563C7B0D9B6CA0E498B09FA839DA1C06CD8B8E7E5DF399F49BB
File 13	Voting.rs
MD5 hash	DDFFA5C26E19D21C80B7A89901EBDEAB
SHA256 hash	79681D861BF132DEFAB30EFA5A015531B153F3AB9D03A6F39B7EB4860574F3A1
File 14	Window.rs
MD5 hash	52105EADCDD3C8BD657FFE2F5FD3CDCA

SHA256 hash	433C2B1EE8F09854E590E28A3B781C238CB3A854362DD5BEAE4686C8F36F7B89
File 15	Withdraw.rs
MD5 hash	ED48F389AFAA6B41E06B578D69766099
SHA256 hash	882CEA4D9BE146550B3AFB35C3DE64E81D2749F6E5ACBB921F32A707617211BE
Folder	Staking token
File 16	Contract.rs
MD5 hash	02F1C4B94921DC082716C01BB8774269
SHA256 hash	1CDC8122F869CD0563EC78EFFE9BAF1EF5D9F6143DA55CF15260809AF07857A7
File 17	Msg.rs
MD5 hash	128E707CAD5C35B68F03F63030639982
SHA256 hash	3C8C90C56E56B56254A403BF2E874E5F280810B3EBB81D56DDA05D835D0FF0B8
File 18	Rand.rs

MD5 hash	38F54D0B644C8F1377B6D99C48E66B1E
SHA256 hash	BDC7008182A137E50493A92F0168AD581C35FAD6A2862A0F5506CB4F8C751AC3
File 19	Receiver.rs
MD5 hash	78235A86A05EBAC69ABF807AA737C1F5
SHA256 hash	5B096287675F570BF36B1F601386A1571D43BAFEEC9CF9C0A37DB684B5FDED8F
File 20	State.rs
MD5 hash	5D3967B89A8C1FA3F7F5BF000C4DBBB5
SHA256 hash	C083F1C029E31CBEE1C8473E416EE755CE66E16D30964DD27402F47441BE9375
File 21	Utils.rs
MD5 hash	CC3B8AF7D2378231798585E22B594188
SHA256 hash	4EF3F1BD39E276CCBFD03FE67391F3F283B5963DAD39DFE63D510B52F4BB7D09
File 22	Viewing_key.rs

MD5 hash	4CF9695EEE62127B1FA00ED6F5E1C7B0
SHA256 hash	44AFA787C2752FA1D76B99573112B37858AF8D548C8F2B91EC40449EE47E0478
Folder	Voting_contract
File 23	Admin.rs
MD5 hash	D7C9B9728B54A7F602B1BB7B904F56BC
SHA256 hash	CB8DB722B7687ADDE56C93E7295D766ED2F2BE9E0388C5A729E3B484B29E17F5
File 24	Contract.rs
MD5 hash	20C88D79178D7382D86D1DD9850C43E9
SHA256 hash	EA48AE03F92C86AB24113A0EB6450EC4B28A2F446EFA2DB76E89351EE254A175
File 25	Lib.rs
MD5 hash	77DB8CFE4642803AB9F2AFAE4637913A

SHA256 hash	7946E2FC7838EF1174535D2066AE59346F35A3F796C8F100E046A5F2746D0BF9
File 26	Msg.rs
MD5 hash	DAE4B5F92168FDE99828E5D0FBCDE2DE
SHA256 hash	750BE7A1148364EEC86B43A2D6BFC030D3D896C904348E5EF19634195F1351C1
File 27	Voting.rs
MD5 hash	CD5F23DF77FFA615FF4E1C702FFD35A3
SHA256 hash	7AD8C0E9593FCE17BE90858EA1D9BD9C586EB1A1E98A963FC8D4AB7F12DFDE21
File 28	State.rs
MD5 hash	FE754A7573CF23F233C082D515EC03E2
SHA256 hash	37DCF2360EB533D52EF543D005CA4378BE2370A448F6C25C5BEF44979F8C3122
Date	1/5/2022

Introduction

RD Auditors (Consultant) were contracted by StakeEasy Finance (Customer) to conduct a Smart Contracts Code Review and Security Analysis. This report represents the findings of the security assessment of the customer`s smart contracts and its code review conducted between 14th April - 1st May 2022.

This contract consists of twenty eight files.

Project Scope

The scope of the project is a smart contract. We have scanned these smart contract for commonly known and more specific vulnerabilities, below are those considered (the full list includes but is not limited to):

- Missing signer/owner
- Integer overflow & underflow
- Arbitrary signed program invocation
- Account confusions
- Unverified Parsed Account
- DuplicateMutableAccount
- Account Cosplay
- Malicious Simulation
- Outdated Version Dependency
- Over Payment
- Inconsistent Rounding
- Other Known / Possible vulnerabilities

The lists of known vulnerabilities related to the RUST programming language. We have checked/tested all possible areas including logical conflict and code flow projections.

Executive Summary

According to the assessment, the customer's RUST smart contract is **well-secured.**

Manual and localized checks are done. All issues were performed by our team, which included the analysis of code functionality, manual audit found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the audit overview section. The general overview is presented in the AS-IS section and all issues found are located in the audit overview section.

We found the following;

Total Issues	1
Critical	0
High	0
Medium	0
Low	0
Very Low	1

Code Quality

The libraries within this smart contract are part of a logical algorithm. A library is a different type of smart contract that contains reusable code. Once deployed on the blockchain (only once), it is assigned to a specific address and its properties/methods can be reused many times by other contracts.

The StakeEasy Finance team has provided scenario and unit test scripts, which helped to determine the integrity of the code in an automated way.

Overall, the code is well commented. Commenting provides rich documentation for functions, return variables and more.

Documentation

We were given the StakeEasy Finance code as a zip file.

The hash of that file is mentioned in the table. As mentioned above, It's well commented smart contract code, so anyone can quickly understand the programming flow as well as complex code logic.

Comments are very helpful in understanding the overall architecture of the protocol. It also provides a clear overview of the system components, including helpful details, like the lifetime of the background script.

Use of Dependencies

As per our observation, the project is providing double-dip gain over Defi services with fine control of admin with the help of Kill Switches using SCRT vs dSCRT. The libraries used in this smart contract infrastructure are based on well-known industry standard open source projects and even core code blocks that are written well and systematically.

AS-IS Overview

File And Function Level Report

Folder: Staking_contract
File: Admin.rs
Observation: Passed
Test Report: Passed
Score: Passed
Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	admin_commands	Passed	All Passed	No Issue	Passed

File: Airdrop.rs
Observation: Passed
Test Report: Passed
Score: Passed
Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	try_claim_airdrop	Passed	All Passed	No Issue	Passed
2	try_send_to_airdrop_dao	Passed	All Passed	No Issue	Passed

3	try_update_airdrop_dao_address	Passed	All Passed	No Issue	Passed
4	try_set_token_viewing_key	Passed	All Passed	No Issue	Passed

File: Claim.rs

Observation: Passed

Test Report: Passed

Score: Passed

Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	claim	Passed	All Passed	No Issue	Passed

File: Contract.rs

Observation: Passed

Test Report: Passed

Score: Passed

Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	init	Passed	All Passed	No Issue	Passed
2	handle	Passed	All Passed	No Issue	Passed
3	try_set_validator_contract	Passed	All Passed	No Issue	Passed
4	try_set_shapshot	Passed	All Passed	No Issue	Passed
5	try_redelegate	Passed	All Passed	No Issue	Passed

File: Deposit.rs

Observation: Passed

Test Report: Passed

Score: Passed

Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	try_deposit	Passed	All Passed	No Issue	Passed
2	try_claim_stake	Passed	All Passed	No Issue	Passed
3	calc_deposit	Passed	All Passed	No Issue	Passed
4	calc_fee	Passed	All Passed	No Issue	Passed

File: Lib.rs

Observation: Passed

Test Report: Passed

Score: Passed

Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	init	Passed	All Passed	No Issue	Passed
2	handle	Passed	All Passed	No Issue	Passed
3	query	Passed	All Passed	No Issue	Passed

File: Queries.rs
Observation: Passed
Test Report: Passed
Score: Passed
Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	query_info	Passed	All Passed	No Issue	Passed
2	query_dev_fee	Passed	All Passed	No Issue	Passed
3	query_exchange_rate	Passed	All Passed	No Issue	Passed
4	query_pending_claims	Passed	All Passed	No Issue	Passed
5	query_current_window	Passed	All Passed	No Issue	Passed
6	query_active_undelegation	Passed	All Passed	No Issue	Passed
7	query_user_claimable	Passed	All Passed	No Issue	Passed
8	query_top_validators	Passed	All Passed	No Issue	Passed
9	fetch_validator_set_from_contract	Passed	All Passed	No Issue	Passed
10	get_top_validators	Passed	All Passed	No Issue	Passed

File: Staking.rs
Observation: Passed
Test Report: Passed
Score: Passed
Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	exchange_rate	Passed	All Passed	No Issue	Passed
2	_calc_exchange_rate	Passed	All Passed	No Issue	Passed
3	get_total_onchain_balance_with_rewards	Passed	All Passed	No Issue	Passed
4	get_balance	Passed	All Passed	No Issue	Passed
5	get_rewards	Passed	All Passed	No Issue	Passed
6	get_rewards_limited	Passed	All Passed	No Issue	Passed
7	get-rewards_all	Passed	All Passed	No Issue	Passed
8	get_rewards_limited	Passed	All Passed	No Issue	Passed
9	undelegate_msg	Passed	All Passed	No Issue	Passed
10	state_msg	Passed	All Passed	No Issue	Passed
11	withdraw_msg	Passed	All Passed	No Issue	Passed
12	redelegate_msg	Passed	All Passed	No Issue	Passed

File: State.rs
Observation: Passed
Test Report: Passed
Score: Passed
Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	store_address	Passed	All Passed	No Issue	Passed
2	get_address	Passed	All Passed	No Issue	Passed
3	store_frozen_exchange_rate	Passed	All Passed	No Issue	Passed
4	get_frozen_exchange_rate	Passed	All Passed	No Issue	Passed

File: Token.rs
Observation: Passed
Test Report: Passed
Score: Passed
Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	query_total_supply	Passed	All Passed	No Issue	Passed

File: Utils.rs
Observation: Passed
Test Report: Passed
Score: Passed
Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	dec_to_uint	Passed	All Passed	No Issue	Passed
2	calc_withdraw	Passed	All Passed	No Issue	Passed
3	calc_threshold	Passed	All Passed	No Issue	Passed

File: Voting.rs
Observation: Passed
Test Report: Passed
Score: Passed
Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	try_vote	Passed	All Passed	No Issue	Passed

File: Window.rs
Observation: Passed
Test Report: Passed
Score: Passed
Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	advance_window	Passed	All Passed	No Issue	Passed
2	check_window_advance	Passed	All Passed	No Issue	Passed

File: Withdraw.rs
Observation: Passed
Test Report: Passed
Score: Passed
Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	trywithdraw	Passed	All Passed	No Issue	Passed
2	release_tokens	Passed	All Passed	No Issue	Passed

Folder: Staking token
File: Contract.rs
Observation: Passed
Test Report: Passed
Score: Passed
Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	init	Passed	All Passed	No Issue	Passed
2	handle	Passed	All Passed	No Issue	Passed
3	pad_response	Passed	All Passed	No Issue	Passed
4	handle	Passed	All Passed	No Issue	Passed
5	try_vote	Passed	All Passed	No Issue	Passed
6	set_voting_contract	Passed	All Passed	No Issue	Passed
7	try_post_initialize	Passed	All Passed	No Issue	Passed
8	query	Passed	All Passed	No Issue	Passed
9	try_burn	Passed	All Passed	No Issue	Passed
10	try_mint	Passed	All Passed	No Issue	Passed
11	update_Voting_msg	Passed	All Passed	No Issue	Passed
12	authenticated_queries	Passed	All Passed	No Issue	Passed
13	query_minters	Passed	All Passed	No Issue	Passed
14	query_view_vote	Passed	All Passed	No Issue	Passed
15	query_exchange_rate	Passed	All Passed	No Issue	Passed
16	query_token_info	Passed	All Passed	No Issue	Passed
17	query_balance	Passed	All Passed	No Issue	Passed
18	query_transactions	Passed	All Passed	No Issue	Passed
19	query_balance	Passed	All Passed	No Issue	Passed
20	stop_minting_gov	Passed	All Passed	No Issue	Passed
21	stop_being_minted	Passed	All Passed	No Issue	Passed

22	add_admin	Passed	All Passed	No Issue	Passed
23	remove_admin	Passed	All Passed	No Issue	Passed
24	change_admin	Passed	All Passed	No Issue	Passed
25	try_create_key	Passed	All Passed	No Issue	Passed
26	set_Contract_status	Passed	All Passed	No Issue	Passed
27	try_transfer	Passed	All Passed	No Issue	Passed
28	try_add_receiver_api_callback	Passed	All Passed	No Issue	Passed
29	try_send	Passed	All Passed	No Issue	Passed
30	try_register_receive	Passed	All Passed	No Issue	Passed
31	insufficient_allowance	Passed	All Passed	No Issue	Passed
32	Update_voting_balances	Passed	All Passed	No Issue	Passed
33	try_transfer_from_impl	Passed	All Passed	No Issue	Passed
34	try_transfer_from	Passed	All Passed	No Issue	Passed
35	try_send_from	Passed	All Passed	No Issue	Passed
36	try_increase_allowance	Passed	All Passed	No Issue	Passed
37	try_decrease_allowance	Passed	All Passed	No Issue	Passed
38	add_minters	Passed	All Passed	No Issue	Passed
39	remove_minters	Passed	All Passed	No Issue	Passed
40	set_minters	Passed	All Passed	No Issue	Passed
41	Perform_transfer	Passed	All Passed	No Issue	Passed
42	is_admin	Passed	All Passed	No Issue	Passed

43	check_if_admin	Passed	All Passed	No Issue	Passed
44	is_valid_symbol	Passed	All Passed	No Issue	Passed

File: Msg.rs

Observation: Passed

Test Report: Passed

Score: Passed

Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	config	Passed	All Passed	No Issue	Passed
2	public_total_supply	Passed	All Passed	No Issue	Passed
3	get_validation_params	Passed	All Passed	No Issue	Passed
4	states_level_to_u8	Passed	All Passed	No Issue	Passed
5	u8_to_status_level	Passed	All Passed	No Issue	Passed
6	space_pad	Passed	All Passed	No Issue	Passed

File: Rand.rs

Observation: Passed

Test Report: Passed

Score: Passed

Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	sha_256	Passed	All Passed	No Issue	Passed
2	new	Passed	All Passed	No Issue	Passed
3	rand_bytes	Passed	All Passed	No Issue	Passed
4		Passed	All Passed	No Issue	Passed

File: Receiver.rs

Observation: Passed

Test Report: Passed

Score: Passed

Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	new	Passed	All Passed	No Issue	Passed
2	into_binary	Passed	All Passed	No Issue	Passed
3	into_cosmos_msg	Passed	All Passed	No Issue	Passed

File: State.rs

Observation: Passed

Test Report: Passed

Score: Passed

Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	voting_password	Passed	All Passed	No Issue	Passed
2	is_voting	Passed	All Passed	No Issue	Passed
3	is_being_minted	Passed	All Passed	No Issue	Passed
4	set_bin_data	Passed	All Passed	No Issue	Passed
5	get_bin_data	Passed	All Passed	No Issue	Passed
6	from_storage	Passed	All Passed	No Issue	Passed
7	as_readonly	Passed	All Passed	No Issue	Passed
8	constants	Passed	All Passed	No Issue	Passed
9	total_supply	Passed	All Passed	No Issue	Passed
10	contract_status	Passed	All Passed	No Issue	Passed
11	minters	Passed	All Passed	No Issue	Passed
12	tx_count	Passed	All Passed	No Issue	Passed
13	voting_contract	Passed	All Passed	No Issue	Passed
14	constants	Passed	All Passed	No Issue	Passed
15	set_constants	Passed	All Passed	No Issue	Passed
16	total_supply	Passed	All Passed	No Issue	Passed
17	set_total_supply	Passed	All Passed	No Issue	Passed
18	contract_status	Passed	All Passed	No Issue	Passed
19	gov_token	Passed	All Passed	No Issue	Passed
20	Voting_contract	Passed	All Passed	No Issue	Passed
21	voting_password	Passed	All Passed	No Issue	Passed
22	set_gov_token	Passed	All Passed	No Issue	Passed
23	set_voting_contract	Passed	All Passed	No Issue	Passed

24	set_voting_password	Passed	All Passed	No Issue	Passed
25	set_is_minting_gov	Passed	All Passed	No Issue	Passed
26	set_is_voting	Passed	All Passed	No Issue	Passed
27	is_voting	Passed	All Passed	No Issue	Passed
28	is_minting_gov	Passed	All Passed	No Issue	Passed
29	set_is_being_minted	Passed	All Passed	No Issue	Passed
30	is_being_minted	Passed	All Passed	No Issue	Passed
31	set_contract_status	Passed	All Passed	No Issue	Passed
32	set_minters	Passed	All Passed	No Issue	Passed
33	add_minters	Passed	All Passed	No Issue	Passed
34	remove_minters	Passed	All Passed	No Issue	Passed
35	minters	Passed	All Passed	No Issue	Passed
36	tx_count	Passed	All Passed	No Issue	Passed
37	set_tx_count	Passed	All Passed	No Issue	Passed
38	constants	Passed	All Passed	No Issue	Passed
39	gov_token	Passed	All Passed	No Issue	Passed
40	voting_contract	Passed	All Passed	No Issue	Passed
41	voting_password	Passed	All Passed	No Issue	Passed
42	is_voting	Passed	All Passed	No Issue	Passed
43	is_minting_gov	Passed	All Passed	No Issue	Passed
44	is_being_minted	Passed	All Passed	No Issue	Passed
45	total_supply	Passed	All Passed	No Issue	Passed
46	contract_status	Passed	All Passed	No Issue	Passed
47	minters	Passed	All Passed	No Issue	Passed

48	tx_count	Passed	All Passed	No Issue	Passed
49	from_storage	Passed	All Passed	No Issue	Passed
50	as_readonly	Passed	All Passed	No Issue	Passed
51	account_amount	Passed	All Passed	No Issue	Passed
52	from_storage	Passed	All Passed	No Issue	Passed
53	as_readonly	Passed	All Passed	No Issue	Passed
54	balance	Passed	All Passed	No Issue	Passed
55	set_account_balance	Passed	All Passed	No Issue	Passed
56	account_amount	Passed	All Passed	No Issue	Passed
57	read_allowance	Passed	All Passed	No Issue	Passed
58	write_allowance	Passed	All Passed	No Issue	Passed
59	write_viewing_key	Passed	All Passed	No Issue	Passed
60	read_viewing_key	Passed	All Passed	No Issue	Passed
61	get_receiver_hash	Passed	All Passed	No Issue	Passed
62	set_receiver_hash	Passed	All Passed	No Issue	Passed
63	slice_to_u128	Passed	All Passed	No Issue	Passed
64	slice_to_u8	Passed	All Passed	No Issue	Passed

File: Utils.rs

Observation: Passed

Test Report: Passed

Score: Passed

Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	ct_slice_compare	Passed	All Passed	No Issue	Passed
2	create_hashhed_password	Passed	All Passed	No Issue	Passed

File: Viewing_key.rs
Observation: Passed
Test Report: Passed
Score: Passed
Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	check_viewing_key	Passed	All Passed	No Issue	Passed
2	new	Passed	All Passed	No Issue	Passed
3	to_hashed	Passed	All Passed	No Issue	Passed
4	as_bytes	Passed	All Passed	No Issue	Passed
5	fmt	Passed	All Passed	No Issue	Passed

Folder: Voting_contract
File: Contract.rs
Observation: Passed
Test Report: Passed
Score: Passed
Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	init	Passed	All Passed	No Issue	Passed
2	handle	Passed	All Passed	No Issue	Passed
3	query	Passed	All Passed	No Issue	Passed
4	query_proposal_state	Passed	All Passed	No Issue	Passed
5	query_inactive_proposal	Passed	All Passed	No Issue	Passed

File: Lib.rs
Observation: Passed
Test Report: Passed
Score: Passed
Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	init	Passed	All Passed	No Issue	Passed
2	handle	Passed	All Passed	No Issue	Passed
3	query	Passed	All Passed	No Issue	Passed

File: State.rs
Observation: Passed
Test Report: Passed
Score: Passed
Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	set_config	Passed	All Passed	No Issue	Passed
2	read_config	Passed	All Passed	No Issue	Passed
3	Contains	Passed	All Passed	No Issue	Passed
4	set_active_Proposal	Passed	All Passed	No Issue	Passed
5	get_active_Proposal	Passed	All Passed	No Issue	Passed
6	get_inactive_Proposal	Passed	All Passed	No Issue	Passed
7	winner	Passed	All Passed	No Issue	Passed
8	change	Passed	All Passed	No Issue	Passed
9	load	Passed	All Passed	No Issue	Passed
10	store	Passed	All Passed	No Issue	Passed
11	set	Passed	All Passed	No Issue	Passed
12	get	Passed	All Passed	No Issue	Passed

File: Voting.rs

Observation: Passed

Test Report: Passed

Score: Passed

Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	ct_slice_compare	Passed	All Passed	No Issue	Passed
2	changge_votes	Passed	All Passed	No Issue	Passed
3	desable_proposal	Passed	All Passed	No Issue	Passed
4	get_proposals	Passed	All Passed	No Issue	Passed

5	active_proposals	Passed	All Passed	No Issue	Passed
6	inactive_proposals	Passed	All Passed	No Issue	Passed
7	set_password	Passed	All Passed	No Issue	Passed
8	query_vote	Passed	All Passed	No Issue	Passed
9	try_Vote	Passed	All Passed	No Issue	Passed
10	tally	Passed	All Passed	No Issue	Passed

File: Admin.rs

Observation: Passed

Test Report: Passed

Score: Passed

Conclusion: Passed

Sl.	Function	Observation	Test Report	Conclusion	Score
1	admin_commands	Passed	All Passed	No Issue	Passed
2	init_vote	Passed	All Passed	No Issue	Passed
3	create_snapshot	Passed	All Passed	No Issue	Passed

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to lost tokens etc.
High	High level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g. public access to crucial functions.
Medium	Medium level vulnerabilities are important to fix; however, they cannot lead to lost tokens.
Low	Low level vulnerabilities are most related to outdated, unused etc. These code snippets cannot have a significant impact on execution.
Lowest Code Style/ Best Practice	Lowest level vulnerabilities, code style violations and information statements cannot affect smart contract execution and can be ignored.

Audit Findings

Critical:

No critical severity vulnerabilities were found.

High:

No high severity vulnerabilities were found.

Medium:

No medium severity vulnerabilities were found.

Low:

No low severity vulnerabilities were found.

Very Low

Some of the functions can be declared without the ‘pub’ keyword, however we did not find any security threats as per the functional logic inside those functions.

Discussion

Staking_contract/src/admin.rs

```
254         HandleMsg::ChangeOwner { new_owner } => {
255             config.admin = new_owner;
256 
257             set_config(&mut deps.storage, &config)?;
258             Ok(HandleResponse::default())
259         }
260 
```

Direct admin change is not recommended, it should be split into two functions transfer_admin_control and accept_admin_control, with two separate calls. This will protect the system from mistakenly transferring to the wrong address/admin.

Conclusion

We were given smart contract files and have used all possible tests based on the given object. The contract is written systematically, so it is ready to go for production.

Since possible test cases can be unlimited, developer level documentation was provided, which helped with the code flow structure. Hence we performed some extra test cases to check the integrity of code and its performance/outcomes as per project scope and its security assessment, and found no vulnerabilities. Under the scope of unlimited test cases we provide no such guarantee of future outcomes. We have used static analysis tools and manual observations to cover maximum possible test cases to scan everything.

The security state of the reviewed contract is now “well-secured”.

Note For contract User

The Software Requirement Specification and developer level documentation were provided, so our test cases/observations were extended to a wide testing scope.

We do not guarantee any discrepancy raised from any dependent library/macros (external objects).

Even if smart contracts are decentralized in nature, some administrative code blocks are necessary to smoothly operate the services encapsulated inside the projects. We do not guarantee any outcomes caused by administrative actions performed with the deployed program.

Technical auditing does not guarantee the project's ethical side. Please do your due diligence before investing. Our audit report is never an investment advice.

Our Methodology

We like to work with a transparent process and make our reviews a collaborative effort. The goals of our security audits are to improve the quality of systems we review and aim for sufficient remediation to help protect users. The following is the methodology we use in our security audit process.

Manual Code Review

In manually reviewing all of the code, we look for any potential issues with code logic, error handling, protocol and header parsing, cryptographic errors, and random number generators. We also watch for areas where more defensive programming could reduce the risk of future mistakes and speed up future audits. Although our primary focus is on the in-scope code, we examine dependency code and behavior when it is relevant to a particular line of investigation.

Vulnerability Analysis

Our audit techniques included manual code analysis, user interface interaction, and whitebox penetration testing. We look at the project's web site to get a high level understanding of what functionality the software under review provides. We then meet with the developers to gain an appreciation of their vision of the software. We install and use the relevant software, exploring the user interactions and roles. While we do this, we brainstorm threat models and attack surfaces. We read design documentation, review other audit results, search for similar projects, examine source code dependencies, skim open issue tickets, and generally investigate details other than the implementation.

Documenting Results

We follow a conservative, transparent process for analyzing potential security vulnerabilities and seeing them through successful remediation. Whenever a potential issue is discovered, we immediately create an Issue entry for it in this document, even though we have not yet verified the feasibility and impact of the issue. This process is conservative because we document our suspicions early even if they are later shown to not represent exploitable vulnerabilities.

We generally follow a process of first documenting the suspicion with unresolved questions, then confirming the issue through code analysis, live experimentation, or automated tests. Code analysis is the most tentative, and we strive to provide test code, log captures, or screenshots demonstrating our confirmation. After this we analyse the feasibility of an attack in a live system.

Suggested Solutions

We search for immediate mitigations that live deployments can take, and finally we suggest the requirements for remediation engineering for future releases. The mitigation and remediation recommendations should be scrutinised by the developers and deployment engineers, and successful mitigation and remediation is an ongoing collaborative process after we deliver our report, and before the details are made public.

Disclaimers

The smart contracts given for audit have been analyzed in accordance with the best industry practices at the date of this report, in relation to: cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report, (Source Code); the Source Code compilation, deployment and functionality (performing the intended functions).

Because the total number of test cases are unlimited, the audit makes no statements or warranties on the security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only - we recommend proceeding with several independent audits and a public bug bounty program to ensure security of smart contracts.

Technical Disclaimer

Smart contracts are deployed and executed on the blockchain. The platform, its programming language, and other software related to the smart contract can have their own vulnerabilities that can lead to hacks. Thus, the audit can’t guarantee explicit security of the audited smart contracts.

Email: info@rdauditors.com

Website: www.rdauditors.com