Architecture Of The Propbase Staking Contract
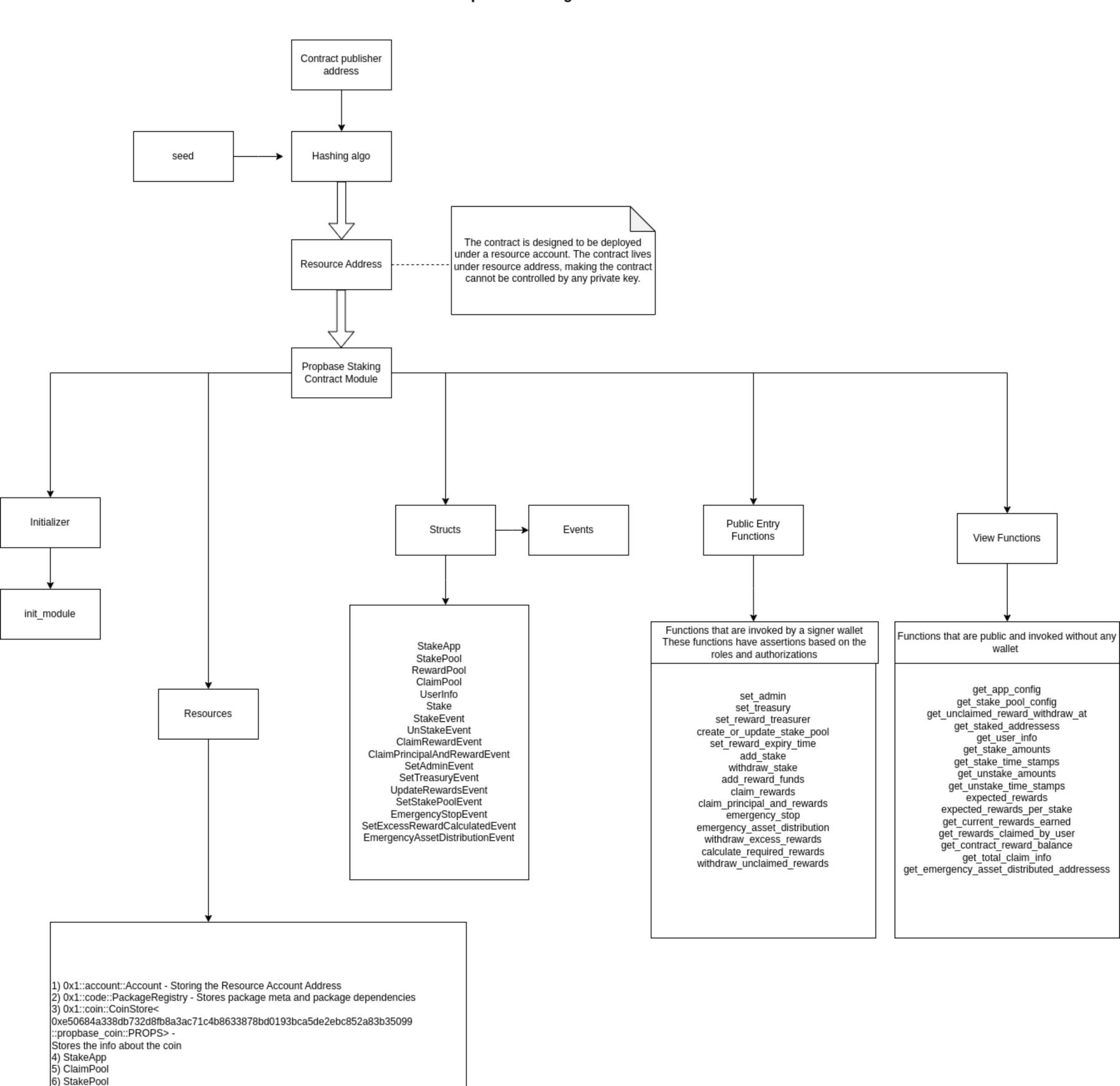
Contract publisher address

seed → Hashing algo

Resource Address

The contract is designed to be deployed under a resource account. The contract lives under resource address, making the contract cannot be controlled by any private key.

Propbase Staking Contract Module

Initializer

init_module

Resources

Structs → Events

Structs
StakeApp
StakePool
RewardPool
ClaimPool
UserInfo
Stake
StakeEvent
UnStakeEvent
ClaimRewardEvent
ClaimPrincipalAndRewardEvent
SetAdminEvent
SetTreasuryEvent
UpdateRewardsEvent
SetStakePoolEvent
EmergencyStopEvent
SetExcessRewardCalculatedEvent
EmergencyAssetDistributionEvent

Public Entry Functions

Functions that are invoked by a signer wallet These functions have assertions based on the roles and authorizations

set_admin
set_treasury
set_reward_treasurer
create_or_update_stake_pool
set_reward_expiry_time
add_stake
withdraw_stake
add_reward_funds
claim_rewards
claim_principal_and_rewards
emergency_stop
emergency_asset_distribution
withdraw_excess_rewards
calculate_required_rewards
withdraw_unclaimed_rewards

View Functions

Functions that are public and invoked without any wallet

get_app_config
get_stake_pool_config
get_unclaimed_reward_withdraw_at
get_staked_addressess
get_user_info
get_stake_amounts
get_stake_time_stamps
get_unstake_amounts
get_unstake_time_stamps
expected_rewards
expected_rewards_per_stake
get_current_rewards_earned
get_rewards_claimed_by_user
get_contract_reward_balance
get_total_claim_info
get_emergency_asset_distributed_addressess

Resources

1) 0x1::account::Account - Storing the Resource Account Address
2) 0x1::code::PackageRegistry - Stores package meta and package dependencies
3) 0x1::coin::CoinStore<
0xe50684a338db732d8fb8a3ac71c4b8633878bd0193bca5de2ebc852a83b35099
::propbase_coin::PROPS> -
Stores the info about the coin
4) StakeApp
5) ClaimPool
6) StakePool
7) RewardPool