PWN Finance Cairo Security Review Report

January 27, 2025

Version 1.0

Email	Telephone
audits@extropy.io	+44 1865338228

Data Classification	Client Confidential
Client Name	PWN
Document Title	PWN Finance Cairo Security Review Report
Document Version	1.0
Author	Extropy Audit Team

Contents

1 Executive Summary **1**

2 Audit summary **2**
2.1 Audit scope . . . 2
2.2 Issues Summary . . . 3
2.3 Methodology . . . 4
2.4 Approach . . . 4
2.5 Audit Notes . . . 4

3 Findings **6**
3.1 [HIGH] Merkle data conversion skips bytes wrongly . . . 6
3.2 [MEDIUM] Proper management of accounts with privileges . . . 6
3.3 [MEDIUM] Usage of rebase tokens may alter the normal functioning of the protocol . . . 7
3.4 [LOW] Possibly empty reference contracts in simple loan . . . 8
3.5 [LOW] LOAN token receiver may be not able to handle tokens . . . 8
3.6 [LOW] Prevent protocol functionality by accepting loans immediately . . . 9
3.7 [LOW] Possibility to pollute a loan with dummy extension proposals . . . 9
3.8 [LOW] Missing checks on ‘credit_amount‘ and ‘available_credit_limit‘ . . . 10
3.9 [LOW] ‘proposal_data.len()‘ is only checked in fungible and dutch proposal types . . . 11
3.10 [LOW] ‘MAX_ACCRUING_INTEREST_APR‘ doesn’t match the Solidity constant . . . 11
3.11 [LOW] Possibility to pollute the protocol with dummy proposals . . . 12
3.12 [LOW] Usage of OZ version with known issues . . . 12
3.13 [LOW] Span decomposition is missing checks in serialization . . . 12
3.14 [INFO] Optimize gas usage for loan minting . . . 13
3.15 [INFO] Duplicate code in loan token URI . . . 14
3.16 [INFO] Typos in code . . . 14
3.17 [INFO] Enumerate loan status options . . . 15
3.18 [INFO] Functions that could be replaced by a multicall . . . 15
3.19 [INFO] Documentation is inconsistent with the code . . . 15
3.20 [INFO] Unneeded access tag parameter . . . 16
3.21 [INFO] Missing unsupported category checks . . . 17
3.22 [INFO] Functions that do not change the contract state are not marked as view . . . 17
3.23 [INFO] ‘IERC721_METADATA_ID‘ interface not registered . . . 17
3.24 [INFO] Use default address . . . 18
3.25 [INFO] Duplicate Event Emission in PWNHub ‘set_tags‘ Function . . . 19
3.26 [INFO] Accruing interest calculation uses a magic number . . . 19
3.27 [INFO] Unneeded use of initializers . . . 20
3.28 [INFO] Math multiplication may panic . . . 20
3.29 [INFO] ‘abi_encoded_packed()‘ never used in the codebase . . . 20
3.30 [INFO] Add comments on how hashes are formed . . . 21
3.31 [INFO] Lock used library versions . . . 22
3.32 [INFO] Inconsistent documentation related to ERC721 token ownership . . . 22

4 Test Coverage **23**
4.1 Solidity tests . . . 23

A Disclaimers **39**
A.1 Client Confidentiality 39
A.2 Proprietary Information 39

1 | Executive Summary

Extropy was contracted to conduct an initial code review and vulnerability assessment of the PWN protocol written in Cairo for the Starknet ecosystem.

The Solidity version of the protocol was already audited by Extropy in May 2024. There are small differences between the Cairo codebases due to the lack of the try-catch construct in the Cairo version currently in use highlighted in the section "2.5 - Audit Notes".

PWN Finance is a decentralized lending protocol that allows users to create new loan proposals: the proposer can either offer credit or request it by creating a proposal which is then subsequently accepted by another user called the acceptor (who is responsible to check the related proposal terms before accepting). Depending on the type of proposal (whether it is an offer or not) proposer and acceptor are also identified as lender and borrower.

There are four types of proposals: `Simple`, which defines a single specific collateral loan; `List`, which defines a list of acceptable collateral ids or a whole collection; `Fungible`, which is not tied to a specific collateral or credit amount: it's specified during the proposal acceptance; `Dutch`, where the collateral amount is fixed while the credit amount change over time while the dutch auction runs.

When a proposal is accepted, a Loan is created and the lender receives an ERC721 token that represents its credit. A loan can be repaid by the borrower paying the interests, if any. If a running loan is not payed back before the timestamp specified by the proposer, or an extension is requested on time for it, it goes into a defaulted state and the borrower loses his collateral.

Overall the protocol is well designed, the code is of high quality and there is evidence of extensive test coverage, using unit tests and integration tests

Section 3 details the findings, where possible we have given recommendations for their resolution.

2 | Audit summary

This audit, conducted from December 2nd 2024 to January 3rd 2025, employed a comprehensive approach using manual review method. Our examination aimed to ensure the robustness and security of the PWN Finance protocol on the Starknet ecosystem.

- The code is taken from the pwn-starknet repository.
- The audit was performed on commit 02b82af8eef699e9999dc4890dc979f75ea930ec.

2.1 | Audit scope

The following contracts were audited:

Contract	LoC
config/interface.cairo	29
config/pwn_config.cairo	406
hub/pwn_hub_tags.cairo	3
hub/pwn_hub.cairo	171
interfaces/fingerprint_computer.cairo	9
interfaces/pool_adapter.cairo	19
interfaces/erc5646.cairo	6
loan/lib/fee_calculator.cairo	29
loan/lib/math.cairo	26
loan/lib/serialization.cairo	116
loan/lib/merkle_proof.cairo	218
loan/terms/simple/loans/error.cairo	75
loan/terms/simple/loans/interface.cairo	36
loan/terms/simple/loans/types.cairo	130
loan/terms/simple/loans/pwn_simple_loan.cairo	1070
loan/terms/simple/proposal/simple_loan_fungible_proposal.cairo	513
loan/terms/simple/proposal/simple_loan_proposal.cairo	343
loan/terms/simple/proposal/simple_loan_list_proposal.cairo	463
loan/terms/simple/proposal/simple_loan_simple_proposal.cairo	368
loan/terms/simple/proposal/simple_loan_dutch_auction_proposal.cairo	597
loan/token/pwn_loan.cairo	275
loan/vault/pwn_vault.cairo	232
multitoken/category_registry.cairo	175
multitoken/library.cairo	533
nonce/revoked_nonce.cairo	324
lib.cairo	76
Total	**6242**

2.2 Issues Summary

ID	Finding	Status
3.1	[HIGH] Merkle data conversion skips bytes wrongly	Resolved
3.2	[MEDIUM] Proper management of accounts with privileges	Acknowledged
3.3	[MEDIUM] Usage of rebase tokens may alter the normal functioning of the protocol	Acknowledged
3.4	[LOW] Possibly empty reference contracts in simple loan	Acknowledged
3.5	[LOW] LOAN token receiver may be not able to handle tokens	Acknowledged
3.6	[LOW] Prevent protocol functionality by accepting loans immediately	Acknowledged
3.7	[LOW] Possibility to pollute a loan with dummy extension proposals	Resolved
3.8	[LOW] Missing checks on 'credit_amount' and 'available_credit_limit'	Acknowledged
3.9	[LOW] 'proposal_data.len()' is only checked in fungible and dutch proposal types	Partially resolved
3.10	[LOW] 'MAX_ACCRUING_INTEREST_APR' doesn't match the Solidity constant	Resolved
3.11	[LOW] Possibility to pollute the protocol with dummy proposals	Acknowledged
3.12	[LOW] Usage of OZ version with known issues	Acknowledged
3.13	[LOW] Span decomposition is missing checks in serialization	Resolved
3.14	[INFO] Optimize gas usage for loan minting	Resolved
3.15	[INFO] Duplicate code in loan token URI	Resolved
3.16	[INFO] Typos in code	Resolved
3.17	[INFO] Enumerate loan status options	Acknowledged
3.18	[INFO] Functions that could be replaced by a multicall	Resolved
3.19	[INFO] Documentation is inconsistent with the code	Resolved
3.20	[INFO] Unneeded access tag parameter	Acknowledged
3.21	[INFO] Missing unsupported category checks	Resolved
3.22	[INFO] Functions that do not change the contract state are not marked as view	Resolved
3.23	[INFO] 'IERC721_METADATA_ID' interface not registered	Resolved
3.24	[INFO] Use default address	Resolved
3.25	[INFO] Duplicate Event Emission in PWNHub 'set_tags' Function	Resolved
3.26	[INFO] Accruing interest calculation uses a magic number	Resolved
3.27	[INFO] Unneeded use of initializers	Partially resolved
3.28	[INFO] Math multiplication may panic	Resolved
3.29	[INFO] 'abi_encoded_packed()' never used in the codebase	Resolved
3.30	[INFO] Add comments on how hashes are formed	Resolved
3.31	[INFO] Lock used library versions	Resolved
3.32	[INFO] Inconsistent documentation related to ERC721 token ownership	Resolved

2.3 | Methodology

2.3.1 | Risk Rating

The risk rating given for issues follows the standard approach of the OWASP Foundation. We combine two factors :

- Likelihood of exploit
- Impact of Exploit

The Categories we use are *Critical*, *High*, *Medium*, *Low* and *Informational* These categories may not align with the categories used by other companies.
The informational category is used to contain suggestions for optimisation (where this is not seen as causing significant impact), or for alternative design or best practices.

2.4 | Approach

The project was assessed mainly by code inspection, with auditors working independently or together, looking for possible exploits. Tests were written were possible to validate the issues found. Manual code inspection techniques were primarily used due to the lack of Static analysis tools for the Starknet ecosystem.

2.5 | Audit Notes

As the current audited codebase is the Starknet version of the Solidity codebase of the protocol audited by Extropy in May 2024, small differences between those two codebases can be highlighted, mainly due to the lack of the try-catch construct in Cairo. Those will be enabled in the future when the try-catch will be added to the Cairo syntax:

- The refinancing functionality. Currently when a user tries to refinance a loan using ‘create_loan()‘ in ‘pwn_simple_loan‘ specifying in input a ‘caller_spec.refinancing_loan_id‘ different than zero, an error will be thrown instead of the ”original” Solidity flow. As a future note, we highlight also that the order of execution of the operations done when refinancing loans may differ from the original codebase: indeed the function ‘_update_repaid_loan()‘ is executed in a different point of ‘create_loan()‘ compared with the solidity version of the protcol and this flow was not analyzed in this audit.
- The auto claim functionality is not available at the moment. Currently the payed back credit goes first to the Vault and only then the owner of the related Loan Token is able to claim it.

Some additional notes:

- Since several contracts in the codebase are upgradeable, the same variable names must be kept during upgrades over time since their hash is used to compute the contract storage slots. Changing just one of them can have dangerous consequences on the related contract and on the whole protocol after the upgrade.
- There is no more multiproposal mechanism: on solidity one can create in one transaction several loan proposal, store their data in a merkle tree and sign the root. This is not present in the current Cairo codebase since multicall can be used
- Within the solidity codebase there are two ways of making proposals: on chain (just storing the proposal hash) or off chain (where you create a proposal off-chain and then provide on chain the data and the signature related to what the proposer proposed off chain). On Starknet instead there

is only the ”on-chain mechanism” of making proposals, so it may be considered to removing nonce functionality related to proposal invalidation, and replace that with simply deleting proposals or marking them deleted. Please note that nonces are also used to distinguish otherwise identical proposals and for extension proposals. In the end not all nonces could be ditched but refactoring the nonce mechanism somewhat on the Cairo codebase can make sense.

- The protocol relies on users to know what kind of tokens they use (for credit and collateral) and whether they're reliable (e.g. collateral may lose value over time) since prices are not relevant for the protocol.

3 | Findings

3.1 | [HIGH] Merkle data conversion skips bytes wrongly

- **Location(s):** merkle_proof.cairo#45
- **Description:** The function `u256_to_be_bytes()` includes logic to truncate zero bytes from the start, to pack the data. However, the logic at line 45 actually removes *any* zero bytes - also from the middle and end.

```
    while bytes
        .len() > 0 {
            let byte = bytes.pop_front().expect('u256_to_be_bytes');

            if byte != 0 {
                significant_bytes.append(byte);
            };
        };
```

  An example: byte vectors [12, 00, 34, 56, 00, 00] and [00, 12, 34, 00, 56] both result in [12, 34, 56].

  This problem can be abused in `simple_loan_list_proposal.accept_proposal()` when a Merkle proof is utilized to check that the offered collateral ID is part of the list of accepted collateral IDs.: the `merkle_inclusion_proof` can be faked. Therefore, an attacker can utilize a worthless collateral ID for a loan, let the loan default and just lose his worthless token.

- **Recommendation:** Consider fixing the implementation so only leading zeroes are removed in `u256_to_be_bytes()`
- **Status:** Resolved.
- Updates:
  - [Extropy, 27/01/2025]: Fixed issue in commit 7b8d1d[1]

3.2 | [MEDIUM] Proper management of accounts with privileges

- **Location(s):** -
- **Description:** Several contracts give extra power to owner address or to addresses with a certain tag (such as `NONCE_MANAGER`).
- **Recommendation:** Make sure that accesses to the accounts with privileges are secure. Consider using a multisig[2] contract for sensitive roles.
- **Status:** Acknowledged.
- Updates:
  - [Extropy, 27/01/2025]: Acknowledged the issue and stated "The owner of the protocol will be a multisig, increasing the security of the protocol by a higher owner threshold necessary for any hub tag updates."

[1]https://github.com/NethermindEth/pwn-starknet/pull/58/commits/7b8d1d9c532e1716b0b5cc0428bc136f96d44bfe
[2]https://github.com/OpenZeppelin/cairo-contracts/blob/main/packages/governance/src/multisig/multisig.cairo

3.3 | [MEDIUM] Usage of rebase tokens may alter the normal functioning of the protocol

- **Location(s):** -
- **Description:** Protocol allows to use rebase tokens that may have bad impact on the normal flow.

  A rebase token is a type of cryptocurrency token whose total supply is dynamically adjusted based on certain criteria such as the token's price or market conditions. The goal of a rebase token is to maintain its price stability relative to a target price. When the token's price deviates from the target, the protocol automatically adjusts the token's supply through "rebases" to bring the price back in line with the target.

  It's possible to have:

  - **Positive Rebase**: If the token price goes above the desired reference price and the price stability protocol aims to lower the token price, it may execute a positive rebase. In this case, it may be necessary to "mint" new supply to increase the total token supply and lower the price. Specifically each user token balance will be increased proportionally.
  - **Negative Rebase**: If the token price goes below the desired reference price and the price stability protocol aims to increase the token price, it may execute a negative rebase. In this case, it may be necessary to "burn" existing supply to reduce the total token supply and raise the price. Specifically each user token balance will be reduced proportionally.

  Due to positive and negative rebase the following scenarios are possible.

  Scenario A:

  1. Loan is created where Alice uses 100 $stETH rebase tokens as a collateral.
  2. Due to positive rebase after some time the balance of $stETH in the vault grows to 106 tokens.
  3. Alice repays the loan. She expects to get 106 $stETH back. But she gets only 100 tokens. 6 tokens are stuck in the vault forever.

  Scenario B:

  1. Loan is created where Alice uses 100 $stETH rebase tokens as a collateral.
  2. Due to a negative rebase after some time the balance of $stETH in the vault decreases to 94 tokens.
  3. Alice repays the loan. The collateral repayment fails as full amount of collateral can't be transferred to the borrower. Alice can't claim the collateral either.

  Scenario C:

  1. Loan is created where Alice uses 100 $stETH rebase tokens as a collateral.
  2. Second loan is created where Bob uses 100 $stETH rebase tokens as a collateral. The vault has now 200 tokens.
  3. Due to a negative rebase after some time the balance of $aSTETH in the vault decreases to 188 tokens.
  4. Bob repays his loan. Bob gets back its full collateral of 100 $stETH .
  5. Alice repays her loan. She should get 100 $stETH tokens. But there is only 88 $stETH tokens in the vault. Alice's collateral repayment fails. Alice can't claim the collateral either. Tokens belonging to Alice has been transferred to Bob

- **Recommendation:** Consider separating vault for each loan and return the full contract balance on collateral repayment.
- **Status:** Acknowledged.
- **Updates:**
  - `[PWN Finance, 22/05/2024]`: The client acknowledged the issue and stated: "We are aware of this issue but decided to keep it unresolved. We don't allow any identified rebalancing tokens to be used on the platform. Rebalancing tokens are known for poor DeFi integration and the need for wrappers/bundlers. Anyone who uses the protocol directly should be well aware of this behavior. We can "solve" the issue of negative rebalance by transferring the missing balance to the vault directly in case of an honest mistake."

3.4 | [LOW] Possibly empty reference contracts in simple loan

- **Location(s):** pwn_simple_loan.cairo#747-771
- **Description:** The addresses stored in `pwn_simple_loan` contract storage may be zero if `initializer()` is not called as the first function after deployment: this may cause panics later in the code.
- **Recommendation:** Consider verifying that the addresses are provided through calling `initializer()` just after deployment.
- **Status:** Acknowledged.
- Updates:
  - [Extropy, 27/01/2025]: Client acknowledged the issue and stated that is part of the deployment process.

3.5 | [LOW] LOAN token receiver may be not able to handle tokens

- **Location(s):** pwn_loan.cairo#159
- **Description:** Within `pwn_loan.cairo` the function `mint()` is supposed to mint new `LOAN` tokens to the loan owner to represent its ownership.

```
fn mint(ref self: ContractState, owner: ContractAddress) -> felt252 {
    let caller = get_caller_address();
    only_active_loan(ref self, caller);
    ...
    self.erc721.mint(owner, loan_id.into());
    ...
}
```

  However `erc721.mint()` from Open Zeppelin is used which does not check if the recipient is able to receive the ERC721 token, namely if it supports IERC721Receiver interface.
- **Recommendation:** Consider using `safe_mint()` instead available here[3].
- **Status:** Acknowledged.

[3]https://github.com/OpenZeppelin/cairo-contracts/blob/8e660c7b91641afee967a7ed86dd2061d471a861/src/token/erc721/erc721.cairo#L583

- Updates:
  - [Extropy, 27/01/2025]: Acknowledged the issue to be consistent with the Solidity codebase Also, the client stated "The assumption is that if the lender could call the create loan or create a proposal, the address could also receive ERC721 tokens"

3.6 | [LOW] Prevent protocol functionality by accepting loans immediately

- **Location(s):** -
- **Description:** When a loan is offered anyone can accept it. The loan can be repaid anytime within the loan's timeframe. The repayment amount depends on:

  1. The original principal amount
  2. How much time has passed from accepting the loan
  3. What is the loan's APR
  4. What is the loan's fixed interest amount

  An attacker can:

  1. Wait whenever someone posts a loan offer with no fixed interest amount
  2. Accept the loan instantly
  3. Repay the loan instantly

  This way the attacker only pays back the principal amount and only loses gas fees. Repaying the loan within a minute will make the usage of (high) APR irrelevant, since repayment amount is calculated only once a minute.

  The attacker can repeat this process for any offered loan (with no fixed interest amount), crippling the protocol. The users offering loans either have to reissue the loan and repay the listing fee or to abandon the protocol.

  This attack only works for offered loans, not requested loans. Furthermore, loans can have setting to lock the address of who can accept the loan, but in reality this is probably not used much.

  The cost of executing this attack is relatively low due to cheap gas costs in Starknet, and the price will only go down due to upgrades to the network.
- **Recommendation:** Consider adding a minimum fixed interest amount to force attackers to pay for the loans.
- **Status:** Acknowledged.
- Updates:
  - [Extropy, 27/01/2025]: Acknowledged the issue and stated to address this issue in the future.

3.7 | [LOW] Possibility to pollute a loan with dummy extension proposals

- **Location(s):** pwn_simple_loan.cairo#400
- **Description:** Suggesting an extension to a loan can be made by anyone: the proposer does not have to be related to the loan itself in any way. It is therefore possible for an attacker to pollute the contracts by making lots of extension proposals. This makes it much harder to find "real" proposals among the dummy ones.

  Please note that a third user making extension proposals for a loan does not mean that they can be later accepted.

- **Recommendation:** Consider forcing the extension proposer to be either the borrower or the lender. This does not fully mitigate the issue, since a participant can also be malicious, but at least limits the scope.
- **Status:** Resolved.
- Updates:
  - [Extropy, 27/01/2025]: Fixed issue in commit 1b2c41[4]

3.8 | [LOW] Missing checks on ‘credit_amount‘ and ‘available_credit_limit‘

- **Location(s):** simple_loan_proposal.cairo#307-314
- **Description:** Each Proposal type contains `credit_amount` and `available_credit_limit` parameters that can be set arbitrarily by the proposer and are used to keep track of how many credit is left in a proposal when is accepted over time by multiple users. Particularly:
  - `proposal.credit_amount` is defined as ”The amount of credit being offered or requested”
  - `proposal.available_credit_limit` is defined as ”The available credit limit for the proposal”

  A problem arise: no checks are made on how `credit_amount` and `available_credit_limit` values relate to each other.

  Indeed:
  - If a proposal is created with `available_credit_limit` lower than `credit_amount`, it cannot be accepted by anyone due to the error `AVAILABLE_CREDIT_LIMIT_EXCEEDED` thrown in `simple_loan_proposal._accept_proposal()`
  - if a proposal is created with `available_credit_limit` greater than `credit_amount`, since `credit_amount` is embedded in the proposal itself, acceptors cannot set it arbitrarily. This means that if `available_credit_limit = 100 ETH` and `credit_amount = 40 ETH`, then the first user accepting the proposal consumes the first 40 ETH, the second the other 40 ETH and then no one else can accept the remaining 20 ETH, since it is obliged to accept the proposal with the `credit_amount` set by the proposer.
- **Recommendation:** Consider placing checks on `credit_amount` and `available_credit_limit` values to ensure that `available_credit_limit` is always greater than or equal to `credit_amount`. Furthermore, if greater, it should be a multiple of `credit_amount`
- **Status:** Acknowledged.
- Updates:
  - [Extropy, 27/01/2025]: Client acknowledged the issue saying this is responsibility of the proposal creator.

[4]https://github.com/NethermindEth/pwn-starknet/pull/57/commits/1b2c413a7d1d4f4bad7329dcf192cce34290822b

3.9 | [LOW] ‘proposal_data.len()‘ is only checked in fungible and dutch proposal types

- **Location(s):** simple_loan_list_proposal.cairo#209, simple_loan_simple_proposal.cairo#174
- **Description:** When `accept_proposal()` is called within `simple_loan_fungible_proposal` and `simple_loan_dutch_auction_proposal` the length of the input `proposal_data` is compared against `FUNGIBLE_PROPOSAL_DATA_LEN` and `DUTCH_PROPOSAL_DATA_LEN` constants respectively.

  For example in `simple_loan_dutch_auction_proposal.cairo`:

  ```
  fn accept_proposal(
      ref self: ContractState,
      acceptor: starknet::ContractAddress,
      refinancing_loan_id: felt252,
      proposal_data: Array<felt252>,
  ) -> (felt252, Terms) {
      if proposal_data.len() != DUTCH_PROPOSAL_DATA_LEN {
          Err::INVALID_PROPOSAL_DATA_LEN(proposal_data.len());
      }
      ...
  }
  ```

  However this check is missing the other proposal types `simple_loan_list_proposal` and `simple_loan_simple_prop`
- **Recommendation:** Consider adding the same check for `simple_loan_list_proposal` and `simple_loan_simple_pro` in their respective `accept_proposal()` functions.
- **Status:** Partially resolved.
- Updates:
  - [Extropy, 27/01/2025]: Fixed `simple_loan_simple_proposal.cairo` in commit 782b97[5] while `simple_loan_list_proposal.cairo` was not fixed.

3.10 | [LOW] ‘MAX_ACCRUING_INTEREST_APR‘ doesn’t match the Solidity constant

- **Location(s):** pwn_simple_loan.cairo#90
- **Description:** Within `pwn_simple_loan.cairo` the constant `MAX_ACCRUING_INTEREST_APR` is defined as it follows

  ```
  pub const MAX_ACCRUING_INTEREST_APR: u32 = 160000;
  ```

  However the same constant is defined differently in the solidity codebase[6]:

  ```
  uint40 public constant MAX_ACCRUING_INTEREST_APR = 16e6; // 160,000 APR (with 2 decimals)
  ```
- **Recommendation:** Consider changing `MAX_ACCRUING_INTEREST_APR` definition in the cairo code-base to match the solidity one.
- **Status:** Resolved.
- Updates:
  - [Extropy, 27/01/2025]: Fixed issue in commit 92a8f7[7]

[5] https://github.com/NethermindEth/pwn-starknet/pull/59/commits/782b974bf3336b0ddc8df66ac560d82a4bd232e0
[6] https://github.com/PWNDAO/pwn_contracts/blob/70e0e76e0d334cef1a5d3dec84d6f031defb6240/src/loan/terms/simple/loan/PWNSimpleLoan.sol#L40
[7] https://github.com/NethermindEth/pwn-starknet/pull/57/commits/92a8f71ba492685368ccbbc66f61cb0a0558f65b

3.11 | [LOW] Possibility to pollute the protocol with dummy proposals

- **Location(s):** simple_loan_dutch_auction_proposal.cairo#229, simple_loan_fungible_proposal.cairo#204, simple_loan_list_ proposal.cairo#182, simple_loan_simple_proposal.cairo#154
- **Description:** Making a proposal is for free and there is no restriction on how many proposals can be created. Therefore it's possible for an attacker to create lots of dummy proposals (for example with a huge initial fee so that nobody accepts the proposals), polluting the protocol and making it hard to find genuine proposals.

  Please note that it's challenging to determine which proposals are real proposals and which are spam.
- **Recommendation:** Consider adding a retainer for making a proposal. Returned when proposal is accepted. Or adding a fixed fee for each proposal.
- **Status:** Acknowledged.
- Updates:
  - [Extropy, 27/01/2025]: Acknowledged issue and acceptedf this potential attack vector.

3.12 | [LOW] Usage of OZ version with known issues

- **Location(s):** Scarb.toml#8
- **Description:** The OpenZeppelin version in use (`0.14.0`) is an old version:

```
openzeppelin = { git = "https://github.com/OpenZeppelin/cairo-contracts.git",
                                            tag = "v0.14.0" }
```

  Note also that version 0.16.0[8] fixed a bug[9] in the OwnableTwoStep implementation, which is being used in pwn_hub.cairo.
- **Recommendation:** Consider upgrading OZ to version 0.19.0 (or newer).
- **Status:** Acknowledged.
- Updates:
  - [Extropy, 27/01/2025]: Acknowledged the issue and stated "Since OZ v0.14.0 is the highest supported version for Cairo 2.6.4, the only viable solution would be a full Cairo upgrade, which is non-trivial and requires thorough testing and adaptation. Until then, sticking with OZ v0.14.0 is the only practical approach while ensuring best practices in contract usage. "

3.13 | [LOW] Span decomposition is missing checks in serialization

- **Location(s):** serialization.cairo#54-73
- **Description:** The function `serde_decompose()` splits a Span into two Spans, according to lengths defined inside the original Span.

  The function is missing at least the following checks:

  1. What if the original Span is empty (function panics)

[8] https://github.com/OpenZeppelin/cairo-contracts/releases/tag/v0.16.0
[9] https://github.com/OpenZeppelin/cairo-contracts/pull/1119

2. What if the "left" length is too big or too small

```
// Example
input = [3, 1, 1, 2, 1, 1]
// left length is 3 but there are two elements.
// right length will be read as one of the left elements
```

3. What if the "right" length is too big or too small

```
// Example
input = [2, 1, 1, 3, 1, 1]
// right length is 3 but there are two elements
```

4. What if the "left" or "right" lengths are not where they are supposed to be.)

```
// Example
input = [1, 2, 1, 1, 1, 3, 1]
// left length is 2 but is not placed at input[0]
// right length is 3 but is not placed at input[left_len +1] = input [3]
```

- **Recommendation:** Consider adding the required checks to the function. Consider what should happen if some of the checks fail.
- **Status:** Resolved.
- **Updates:**
  - [Extropy, 27/01/2025]: Fixed issue in commit 4bc7cf[10]

3.14 | [INFO] Optimize gas usage for loan minting

- **Location(s):** pwn_loan.cairo#154-155
- **Description:** The `mint()` function reads the `last_loan_id` from storage twice. It can be optimized to only read it once.

```
self.last_loan_id.write(self.last_loan_id.read() + 1);
let loan_id: felt252 = self.last_loan_id.read();
```

- **Recommendation:** Consider reading the last loan id first to the `loan_id` variable.
- **Status:** Resolved.
- **Updates:**
  - [Extropy, 27/01/2025]: Fixed issue in commit 026cab[11]

[10] https://github.com/NethermindEth/pwn-starknet/pull/60/commits/4bc7cfa23d24b3171923ef83dfc9bc9d54a8c7fa
[11] https://github.com/NethermindEth/pwn-starknet/pull/57/commits/026cabee26eb7762ff867c3351c0f5302cf67042

3.15 | [INFO] Duplicate code in loan token URI

- **Location(s):** pwn_loan.cairo#213-220, pwn_loan.cairo#234+241
- **Description:** The function `token_uri()` is duplicated with another naming convention `tokenUri()`. Implementation is duplicated as well unnecessarily.

```
fn token_uri(self: @ContractState, loan_id: felt252) -> ByteArray {
    self.erc721._require_owned(loan_id.into());

    IPwnLoadMetadataProviderDispatcher {
        contract_address: self.loan_contract.read(loan_id)
    }
        .loan_metadata_uri()
}

fn tokenUri(self: @ContractState, loan_id: felt252) -> ByteArray {
    self.erc721._require_owned(loan_id.into());

    IPwnLoadMetadataProviderDispatcher {
        contract_address: self.loan_contract.read(loan_id)
    }
        .loan_metadata_uri()
}
```

- **Recommendation:** Consider calling `token_uri` function from `tokenUri` function to remove duplication.
- **Status:** Resolved.
- Updates:
  - [Extropy, 27/01/2025]: Fixed issue in commit 9ed1c1[12]

3.16 | [INFO] Typos in code

- **Location(s):** pwn_simple_loan.cairo#555, pwn_simple_loan.cairo#971, pwn_simple_loan.cairo#974
- **Description:** There exist various small typos in the code of `pwn_simple_loan.cairo`:
  - At line 555 "calladata" is written instead of "calldata"
  - At line 971 "accuring_minutes" is written instead of "accruing_minutes"
  - At line 974 "accured_interest" is written instead of "accrued_interest"
- **Recommendation:** Consider fixing the mentioned typos to improve code readability. Please also note that the input named "calladata" of `get_lender_spec_hash()` seems to be an error made when translating the codebase from Solidity to Cairo: instead of `lender_spec` the input name used in the Cairo codebase was the Solidity Data Location calldata.
- **Status:** Resolved.
- Updates:
  - [Extropy, 27/01/2025]: Fixed issue in commit 703246[13]

[12] https://github.com/NethermindEth/pwn-starknet/pull/57/commits/9ed1c1037f56a41333620051ac3da29b05f07aa7
[13] https://github.com/NethermindEth/pwn-starknet/pull/57/commits/70324640d12b6cb6c72d44f25086e33924e4160d

3.17 | [INFO] Enumerate loan status options

- **Location(s):** types.cairo#59 , pwn_simple_loan.cairo#373-382
- **Description:** Arbitrary numbers are used for `Loan.status`:

```
/// Represents a loan with its status and terms.
#[derive(Copy, Drop, Serde, Default, starknet::Store)]
pub struct Loan {
    /// The status of the loan.
    pub status: u8,
    ...
}
```

- **Recommendation:** Consider adding an enum that explains the options for loan status and utilizing that
- **Status:** Acknowledged.
- Updates:
  - [Extropy, 27/01/2025]: Acknowledged issue to be consistent with the Solidity codebase.

3.18 | [INFO] Functions that could be replaced by a multicall

- **Location(s):** pwn_hub.cairo#130 , revoked_nonce.cairo#205
- **Description:** Within the codebase it happens to have two functions for the same functionality: the first sets some data based on one struct, given as parameter. The second function takes a vector of structs as parameter, and sets the same data for all of them. For example `set_tag()` and `set_tags()` in `pwn_hub.cairo`

  This pattern is common in Solidity side, but is possibly not needed in Cairo. In Starknet, one can use account contract's multicall functionality[14] instead. The multicall is typically utilized by wallet software (like Argent or Braavos), and is not very handy for contract-to-contract interactions. This is why it may be desirable to still keep dual functions sometimes.
- **Recommendation:** Consider whether some duplicate functionalities could be removed.
- **Status:** Resolved.
- Updates:
  - [Extropy, 27/01/2025]: Fixed issue in commit 4ec43b[15]

3.19 | [INFO] Documentation is inconsistent with the code

- **Location(s):** serialization.cairo#3-4, serialization.cairo#40-41
- **Description:** Within `serialization.sol` the function `serde_concat()` is supposed to concatenate two `Span<felt252>` into one single `Array<felt252>` inserting in it in order:
  - the length of the first `Span<felt252>` element

[14] https://github.com/OpenZeppelin/cairo-contracts/blob/release-v0.14.0/src/account/account.cairo#L92
[15] https://github.com/NethermindEth/pwn-starknet/pull/57/commits/4ec43b24d5ba6ffbf033017adf0a8bb8b4f4b60b

- the elements of the first `Span<felt252>` element
- the length of the second `Span<felt252>` element
- the elements of the second `Span<felt252>` element

Example:

- Span 1 -¿ [1,2]
- Span 2 -¿ [3,4,5,6]
- Concatenated array -¿ [2,1,2,4,3,4,5,6]

However the comment at lines 3-4 states that the output array has the two lengths as the first two elements and then the span elements.

```
/// The output format includes the lengths of both slices at the beginning
/// followed by the elements of each slice.
```

The same thing applies to the comments at lines 40-41 related to the function `serde_decompose()`

- **Recommendation:** Consider changing the comments in order to be consistent with the code.
- **Status:** Resolved.
- Updates:
  - [Extropy, 27/01/2025]: Fixed issue in commit 05610a[16]

3.20 | [INFO] Unneeded access tag parameter

- **Location(s):** revoked_nonce.cairo#121
- **Description:** The access tag is given as a parameter for the constructor. However, the parameter doesn't depend on any deployment time information and could be a constant variable inside the contract.

```
#[constructor]
fn constructor(ref self: ContractState, hub: ContractAddress, access_tag: felt252) {
    self.hub.write(IPwnHubDispatcher { contract_address: hub });
    self.access_tag.write(access_tag);
}
```

- **Recommendation:** Consider removing the access tag parameter from the constructor and instead specifying the variable as a constant inside the file.
- **Status:** Acknowledged.
- Updates:
  - [Extropy, 27/01/2025]: Acknowledged issue to be consistent with the Solidity codebase Also, the contract can be redeployed with different tags without any code changes.

[16]https://github.com/NethermindEth/pwn-starknet/pull/57/commits/05610a234d748e3989d827d52480198225179009

3.21 | [INFO] Missing unsupported category checks

- **Location(s):** library.cairo#196-211, library.cairo#299-315, library.cairo#317-329
- **Description:** Within `library.cairo` the functions `approve_asset()`, `_check_category_via_src5()` and `_check_format_()` do not throw the `UNSUPPORTED_CATEGORY` error if the `Asset` in input does not belong to the allowed categories as done in other functions, for example in `balance_of()` as shown below.

```
if *self.category != Category::ERC20
    && *self.category != Category::ERC721
    && *self.category != Category::ERC1155 {
    Err::UNSUPPORTED_CATEGORY(*self.category);
}
```

  `approve_asset()`, `_check_category_via_src5()` and `_check_format_()` use the `match` keyword which is designed to be exhaustive: if the Asset category in input is not one between ERC20, ERC721 or ERC1155 it will panic.

- **Recommendation:** Consider adding the aforementioned check to `approve_asset()`, `_check_category_via_src5()` and `_check_format_()` to reflect the behavior of the solidity codebase (available here[17]) which throws an error for unsupported categories in the respective functions instead of panicking.
- **Status:** Resolved.
- Updates:
  - [Extropy, 27/01/2025]: Fixed issue in commit f826ef[18]

3.22 | [INFO] Functions that do not change the contract state are not marked as view

- **Location(s):** pwn_simple_loan.cairo#944, pwn_simple_loan.cairo#1020
- **Description:** Within `pwn_simple_loan.cairo` the functions `_check_loan_can_be_repaid()` and `_check_valid_asset()` use a reference to the ContractState even if no state modifications happen within those functions
- **Recommendation:** Consider changing the first function argument to `self: @ContractState`
- **Status:** Resolved.
- Updates:
  - [Extropy, 27/01/2025]: Fixed issue in commit 89eca9[19]

3.23 | [INFO] ‘IERC721_METADATA_ID‘ interface not registered

- **Location(s):** pwn_loan.cairo#126-133
- **Description:** Within `pwn_loan.cairo` constructor the name and symbol of ERC721 component are set directly without using the provided `initializer()` function from here[20]. This design choice was probably made because there is no need to pass anything for the third parameter of the `initializer()` function which is the `base_uri`.

[17] https://github.com/PWNDAO/MultiToken/blob/863dcd8b4c60494d1deda231fb95b48073d85659/src/MultiToken.sol
[18] https://github.com/NethermindEth/pwn-starknet/pull/57/commits/f826ef23dc39657d47183c29d87c6b6aa00873eb
[19] https://github.com/NethermindEth/pwn-starknet/pull/57/commits/89eca9286f4156e7f8ade4b1187d826193b456e9
[20] https://github.com/OpenZeppelin/cairo-contracts/blob/8e660c7b91641afee967a7ed86dd2061d471a861/src/token/erc721/erc721.cairo#L478

```
fn constructor(ref self: ContractState, hub: ContractAddress) {
    self.hub.write(IPwnHubDispatcher { contract_address: hub });
    self.erc721.ERC721_name.write("PWN LOAN");
    self.erc721.ERC721_symbol.write("LOAN");

    self.src5.register_interface(IERC721_ID);
    self.src5.register_interface(IERC5646_ID);
}
```

However the `initializer()` function, in addition to set the name, the symbol and the token base URI, also registers the interfaces `IERC721_ID` and `IERC721_METADATA_ID`. Within the current codebase only `IERC721_ID` and `IERC5646_ID` are registered within `pwn_loan` constructor.

Registering the `IERC721_METADATA_ID` means adding support to the additional view functions `name()`, `symbol()` and `token_uri()` as stated on the docs[21]. Even if those functions are implemented manually at the end of `pwn_loan.cairo` file and so they provide the same functionality from a user's perspective, without registering the interface ID external contracts or dApps might not be able to automatically recognize that the contract supports the metadata extension.

- **Recommendation:** Consider using the `initializer()` function of the ERC721 component leaving empty the `base_uri` field if not needed. Also consider removing the reimplemented view functions as they are provided when the initializer registers the `IERC721_METADATA_ID` interface.
- **Status:** Resolved.
- Updates:
  - [Extropy, 27/01/2025]: Fixed issue in commit 9767b8[22]

3.24 | [INFO] Use default address

- **Location(s):** lib.cairo#71
- **Description:** There is an implementation for `Default::default()` contract address but it's not utilized consistently.

```
impl ContractAddressDefault of Default<starknet::ContractAddress> {
    #[inline(always)]
    fn default() -> starknet::ContractAddress nopanic {
        starknet::contract_address_const::<0>()
    }
}
```

- **Recommendation:** Consider replacing all occurences of `starknet::contract_address_const::<0>()` or `contract_address_const::<0>()` with `Default::default()`
- **Status:** Resolved.
- Updates:
  - [Extropy, 27/01/2025]: Fixed issue in commit a108b4[23]

[21] https://docs.openzeppelin.com/contracts-cairo/0.14.0/api/erc721#IERC721Metadata
[22] https://github.com/NethermindEth/pwn-starknet/pull/57/commits/9767b844a458eddc37c3bda45c9298dcf5b17ef8
[23] https://github.com/NethermindEth/pwn-starknet/pull/57/commits/a108b4e50a76a31a9041d945e88d725c1856efc5

3.25 | [INFO] Duplicate Event Emission in PWNHub ‘set_tags‘ Function

- **Location(s):** pwn_hub.cairo#149
- **Description:** Within `pwn_hub.cairo` the function `set_tags()` the current implementation redundantly emits the TagSet event twice for each tag assignment:

  1. Within the `set_tag` function, which is called for each address-tag pair.
  2. Explicitly within the `set_tags` function, after the call to `set_tag`.

  This results in duplicate event emissions, which lead to the following issues:

  - Gas Inefficiency: Redundant event emissions increase the gas cost for each execution, particularly when processing a large number of addresses and tags.
  - Log Bloating: The event logs are unnecessarily bloated, which can make event indexing and monitoring more complex.
  - Potential Confusion: Monitoring systems relying on events may misinterpret duplicate events as separate operations, causing ambiguity in data analysis.

- **Recommendation:** Since the `set_tag` functions already emits the `TagSet` event, consider removing the redundant event emission in order to prevent duplicate events from being logged.
- **Status:** Resolved.
- Updates:
  - [Extropy, 27/01/2025]: Fixed issue in commit 4ec43b[24] by removing `set_tags()`

3.26 | [INFO] Accruing interest calculation uses a magic number

- **Location(s):** pwn_simple_loan.cairo#93
- **Description:** The value of `ACCRUING_INTEREST_APR_DENOMINATOR` looks like a magic number.

```
pub const ACCRUING_INTEREST_APR_DENOMINATOR: u64 = 5256000000;
```

  In the Solidity version[25] instead its origin it’s clear:

```
uint256 public constant ACCRUING_INTEREST_APR_DENOMINATOR =
            ACCRUING_INTEREST_APR_DECIMALS * MINUTES_IN_YEAR * 100;
```

- **Recommendation:** Consider formulating the number explicitly, to make its origin clearer:

```
pub const ACCRUING_INTEREST_APR_DENOMINATOR: u64 = 100 * MINUTES_IN_YEAR * 100;
```

  Unfortunately, the value `ACCRUING_INTEREST_APR_DECIMALS` can’t be used directly, but it could be indicated in a comment.

- **Status:** Resolved.
- Updates:
  - [Extropy, 27/01/2025]: Fixed issue in commit 877c1d[26]

[24] https://github.com/NethermindEth/pwn-starknet/pull/57/commits/4ec43b24d5ba6ffbf033017adf0a8bb8b4f4b60b
[25] https://github.com/PWNDAO/pwn_contracts/blob/70e0e76e0d334cef1a5d3dec84d6f031defb6240/src/loan/terms/simple/loan/PWNSimpleLoan.sol#L44
[26] https://github.com/NethermindEth/pwn-starknet/pull/57/commits/877c1d6f23fce8572c70cafea9a997796b4b9499

3.27 | [INFO] Unneeded use of initializers

- **Location(s):** pwn_config.cairo#160, pwn_simple_loan.cairo#747
- **Description:** Both of the listed contracts utilize an initializer function while a constructor should be used instead.
- **Recommendation:** In the `pwn_config` contract, consider changing the `initialize()` function to be a constructor and remove dependencies of OZ's Initializable component. For `pwn_simple_loan` contract, consider removing the `initializer()` function after moving its contents to the constructor.
- **Status:** Partially Resolved.
- Updates:
  - [Extropy, 27/01/2025]: Partially fixed issue in commit c56936[27]. Location pwn_config.cairo#160 should still be fixed.
  - [PWN, 27/01/2025]: "config will be deployed as a proxy"

3.28 | [INFO] Math multiplication may panic

- **Location(s):** math.cairo#25
- **Description:** The cast from `u512` into `u256` may panic. Indeed `try_into().unwrap()` may panic if `c` has several lower orders of magnitude than `a*b`. This can be caused by calculation that use the `mul_div()` function.

```
pub fn mul_div(a: u256, b: u256, c: u256) -> u256 {
    if c == 0 {
        panic!("mul_div division by zero");
    }
    let (q, _) = u512_safe_div_rem_by_u256(u256_wide_mul(a, b), c.try_into().unwrap());
    q.try_into().unwrap()
}
```

- **Recommendation:** Consider panicking gracefully if the number is not in the range of a `u256`.
- **Status:** Resolved.
- Updates:
  - [Extropy, 27/01/2025]: Fixed issue in commit 83515a[28]

3.29 | [INFO] 'abi_encoded_packed()' never used in the codebase

- **Location(s):** merkle_proof.cairo#5
- **Description:** The function `abi_encoded_packed()` is never used in the codebase:

[27] https://github.com/NethermindEth/pwn-starknet/pull/57/commits/c569367b68c240a03a8778ec4464a891fc8e9671
[28] https://github.com/NethermindEth/pwn-starknet/pull/61/commits/83515a50a2a0063e4056f2edc0fc3038b81373eb

```
pub fn abi_encoded_packed(data: Array<u256>) -> Array<u8> {
    let mut result: Array<u8> = array![];
    let mut i = 0;
    let len = data.len();
    while i < len {
        result = result.concat(@u256_to_be_bytes(*data.at(i)));
        i += 1;
    };
    result
}
```

- **Recommendation:** Consider removing dead code.
- **Status:** Resolved.
- **Updates:**
  - `[Extropy, 27/01/2025]`: Fixed issue in commit 0654ee[29]

3.30 | [INFO] Add comments on how hashes are formed

- **Location(s):** pwn_hub_tags.cairo#1-3, erc5646.cairo#1, library.cairo#52-57
- **Description:** The used hashes seem arbitrary, which leaves an open question about how they are formed.

```
pub const ACTIVE_LOAN: felt252 =
  0x0256ea094d7a53440eef11fa42b63159fbf703b4ee579494a6ae85afc5603594;
pub const LOAN_PROPOSAL: felt252 =
  0xba7a416221f318a8087fd62f9ff407488b7f5501e79caf9b0666c2df326b9c;
pub const NONCE_MANAGER: felt252 =
  0x1b33e4d1c538d376dd219215a123562fbb87b8c85fa2aa4ebbd8810c2454d9;

pub const IERC5646_ID: felt252 =
  0x012ee61ceedb7b8ff3da67d4e5d24d13d2a1ef35fdcd3a10b9138823f62342ba;

const ERC20_INTERFACE_ID: felt252 =
  0x3d21dcd478803698af065a01681e1f1801a5b80c367ecb5561fbf10b416756e;
const ERC721_INTERFACE_ID: felt252 =
  0x2c8b9553a387f54d6021766166528967ac9bb9393acf1c47678b9eea63dda07;
const ERC1155_INTERFACE_ID: felt252 =
  0xcd38fd6bb8f64dd3988ff3ae65d0cd040c95aaad81b509a1f4f0b3e40adf88;
```

- **Recommendation:** Consider adding comments on how the hashes are formed so it's clear that they are sensible.
- **Status:** Resolved.
- **Updates:**
  - `[Extropy, 27/01/2025]`: Fixed issue in commit d78f0f[30]

[29] https://github.com/NethermindEth/pwn-starknet/pull/57/commits/0654ee0ed55f44a9a278fc67e31649ba882627b9
[30] https://github.com/NethermindEth/pwn-starknet/pull/62/commits/d78f0f202eccbe44ed3d34793822c71f66f5c3d6

3.31 | [INFO] Lock used library versions

- **Location(s):** Scarb.toml#9-19
- **Description:** The used Alexandria libraries utilize the latest versions. This may cause problems if the versions change during development, without developers noticing, and there is a new introduced bug in Alexandria.
- **Recommendation:** Consider locking the used versions.
- **Status:** Resolved.
- **Updates:**
  - `[Extropy, 27/01/2025]`: Fixed issue in commit 38f3a3[31]

3.32 | [INFO] Inconsistent documentation related to ERC721 token ownership

- **Location(s):** pwn_loan.cairo#213, pwn_loan.cairo#234
- **Description:** The documentation claims that if the token is not owned by the caller, a TOKEN_NOT_OWNED error is given. This is incorrect because:

  1. Error is thrown only if the token is not owned by anyone. The caller doesn't matter
  2. The thrown error is actually "ERC721: invalid token ID".

  Related OZ code available here[32].
- **Recommendation:** Consider changing the documentation to state the right error message and not claim that it's related to the caller.
- **Status:** Resolved.
- **Updates:**
  - `[Extropy, 27/01/2025]`: Fixed issue in commit 803444[33]

[31] https://github.com/NethermindEth/pwn-starknet/pull/63/commits/38f3a390e1ba369016598c615a6441254862630b
[32] https://github.com/OpenZeppelin/cairo-contracts/blob/release-v0.14.0/src/token/erc721/erc721.cairo#L78
[33] https://github.com/NethermindEth/pwn-starknet/pull/57/commits/803444708f74c01c3c3857921fa6ed5520626949

4 | Test Coverage

4.1 | Solidity tests

The Cairo (Starknet) project has a comprehensive suite of unit tests, with a focus on a loan and multi-token system. The project's test suite is in excellent health, with no failing tests. The high number of passing tests indicates thorough testing practices and provides a high level of confidence in the correctness and security of the code.
Also, the detailed gas usage information is helpful for identifying potential areas for optimization, while the ignored tests should be addressed to further enhance the test coverage.
However it can be noted that some of the unit test files have thousands of lines, which makes them very difficult to analyze and read. Structuring the tests better, for example to sub-folders, utilizing common setup functionalities if needed, can be considered.

4.1.1 | Key Areas Tested

MultiToken Functionality:

- Tests cover ERC20, ERC721, and ERC1155 token standards.
- Checks for correct category and format validation.
- 'MultiToken' library tests ensure proper handling of different token types and interactions with external contracts.
- Tests using 'check_category_via_src5' ensure proper detection of NFT contracts.

Loan System:

- Loan Creation: Extensive tests for creating loans, including various parameters, edge cases, and error conditions. Fuzz tests are used to test a wide range of inputs.
- Loan Repayment: Tests cover loan repayment, including scenarios with different loan owners, default conditions, and edge cases.
- Loan Claiming: Tests for claiming both repaid and defaulted loans, including transferring of assets and edge cases.
- Loan Extension: Tests for extending loans, including proposal creation, different caller scenarios, and error conditions.
- Loan Metadata: Tests for setting, updating, and retrieving loan metadata.
- Loan Status: Verification of different loan states (e.g., running, defaulted, repaid).
- State Fingerprint: Correct calculation and updating of the state fingerprint.
- Refinancing: The tests include some checks regarding refinancing.

Proposal Types:

- Tests for different proposal types, including Simple, Fungible, Dutch Auction, and List proposals.
- These tests cover proposal creation, data encoding/decoding, credit usage, nonce revocation, and various validation checks.

Fee Calculation:

- Tests for fee calculation, including edge cases like zero amounts, small amounts, and non-zero fees.
- Tests for setting and updating the fee collector address.

Nonce Revocation:

- Extensive fuzz tests for nonce revocation, including different nonce spaces and owner scenarios.
- Tests for checking nonce usability and revocation status.

Merkle Proof:

- Tests for hashing and verifying Merkle proofs.

Serialization:

- Tests for serialization and deserialization of data structures.

Configuration:

- Tests for setting and updating configuration parameters, including fee, fee collector, and loan metadata URI.

Hub Contract:

- Tests for setting and removing tags from addresses in the hub contract.

Vault:

- Basic tests for vault operations like 'push', 'pull', 'supply_to_pool', and 'withdraw_from_pool'.

Integration Tests:

- Limited integration tests for creating loans using various proposal types and claiming repaid and defaulted loans.
- Integrity tests for scenarios where the loan contract is not active.

4.1.2 | Gas Usage

The test results include gas usage information, which is valuable for optimizing contract efficiency. Some tests, particularly those involving loops or complex logic, have higher gas consumption.

4.1.3 | Fuzzer Seed

The fuzzer seed is provided, which allows for reproducing the exact set of fuzz tests that were run.

```
snforge test

Collected 358 test(s) from pwn package
[PASS] pwn::multitoken::library::MultiToken::test::test_check_category_via_src5_shou
     ld_return_false_when_erc20_when_src5_supports_erc721 (gas: ~1)
[PASS] pwn::multitoken::library::MultiToken::test::test_check_category_via_src5_shou
     ld_return_true_when_erc721 (gas: ~3)
[PASS] pwn::multitoken::library::MultiToken::test::test_check_category_via_src5_shou
     ld_return_false_when_erc20_when_src5_supports_erc1155 (gas: ~1)
[PASS] pwn::multitoken::library::MultiToken::test::test_check_category_via_src5_shou
     ld_return_false_when_erc20 (gas: ~1)
[PASS] pwn::multitoken::library::MultiToken::test::test_check_category_via_src5_shou
```

```
     ld_return_true_when_erc1155 (gas: ~3)
[PASS] pwn::loan::lib::serialization::tests::test_serde_struct (gas: ~3)
[PASS] pwn::loan::lib::merkle_proof::tests::test_hash (gas: ~1204)
[PASS] pwn::loan::lib::merkle_proof::tests::test_hash2 (gas: ~1197)
[PASS] pwn::loan::lib::serialization::tests::test_serde_concat (gas: ~2)
[PASS] pwn::loan::lib::serialization::tests::test_serde_decompose (gas: ~3)
[PASS] pwn::loan::lib::merkle_proof::tests::test_hash2_2 (gas: ~1215)
[PASS] pwn::loan::lib::merkle_proof::tests::test_verify_proof_mock_proof (gas: ~3629
     )
[PASS] pwn::loan::lib::merkle_proof::tests::test_verify_proof (gas: ~4860)
[PASS] pwn::multitoken::library::MultiToken::test::test_check_category_should_return
     _true_when_category_registered (runs: 256, gas: {max: ~3, min: ~3, mean: ~3.00,
      std deviation: ~0.00})
[PASS] pwn::multitoken::library::MultiToken::test::test_check_category_should_return
     _false_when_different_category_registered (runs: 256, gas: {max: ~3, min: ~3, m
     ean: ~3.00, std deviation: ~0.00})
[PASS] pwn::multitoken::library::MultiToken::test::test_check_category_should_return
     _true_when_category_not_registered_when_check_via_src5_returns_true (runs: 256,
      gas: {max: ~7, min: ~4, mean: ~6.00, std deviation: ~1.40})
[PASS] pwn::multitoken::library::MultiToken::test::test_check_format_should_return_t
     rue_when_erc721_with_zero_amount (runs: 256, gas: {max: ~1, min: ~1, mean: ~1.0
     0, std deviation: ~0.00})
[PASS] pwn::multitoken::library::MultiToken::test::test_check_format_should_return_f
     alse_when_erc20_with_non_zero_id (runs: 256, gas: {max: ~1, min: ~1, mean: ~1.0
     0, std deviation: ~0.00})
[PASS] pwn::multitoken::library::MultiToken::test::test_check_format_should_return_t
     rue_when_erc20_with_zero_id (runs: 256, gas: {max: ~1, min: ~1, mean: ~1.00, st
     d deviation: ~0.00})
[PASS] pwn::multitoken::library::MultiToken::test::test_check_format_should_return_f
     alse_when_erc721_with_non_zero_amount (runs: 256, gas: {max: ~1, min: ~1, mean:
      ~1.00, std deviation: ~0.00})
[PASS] pwn::multitoken::library::MultiToken::test::test_check_format_should_return_t
     rue_when_erc1155 (runs: 256, gas: {max: ~1, min: ~1, mean: ~1.00, std deviation
     : ~0.00})
[IGNORE] tests::integration::simple_loan_integration_test::test_should_repay_loan_wh
     en_not_expired_when_original_lender_is_loan_owner
[IGNORE] tests::integration::protocol_integrity_test::test_should_repay_loan_when_lo
     an_contract_not_active_when_original_lender_is_loan_owner
[IGNORE] tests::unit::simple_loan_test::create_loan::test_fuzz_should_call_proposal_
     contract
[IGNORE] tests::fork::deployed_protocol_test::test_deployed_protocol
[IGNORE] tests::fork::use_cases_test::test_use_case_should_fail_when_20_collateral_p
     assed_with_721_category
[IGNORE] tests::fork::use_cases_test::test_use_case_should_fail_when_20_collateral_p
     assed_with_1155_category
[IGNORE] tests::fork::use_cases_test::test_use_case_should_fail_when_using_erc721_as
     _credit
[IGNORE] tests::fork::use_cases_test::test_should_pass_when_invalid_src5_support
[IGNORE] tests::fork::use_cases_test::test_use_case_should_refinance_running_loan
[PASS] tests::unit::config_test::set_fee_collector::test_should_set_fee_collector_ad
     dress (gas: ~368)
[PASS] tests::unit::config_test::set_loan_metadata_uri::test_should_fail_when_zero_l
     oan_contract (gas: ~367)
[PASS] tests::unit::LOAN_test::test_should_have_correct_name_and_symbol (gas: ~557)
[PASS] tests::unit::LOAN_test::test_should_fail_when_caller_is_not_stored_loan_contr
     act_for_given_loan_id (gas: ~823)
[PASS] tests::unit::config_test::set_fee::test_should_emit_event_fee_updated (gas: ~
     369)
```

```
[PASS] tests::unit::config_test::set_loan_metadata_uri::test_should_fail_when_caller
     _is_not_owner (gas: ~366)
[PASS] tests::unit::config_test::initialize::test_should_fail_when_fee_collector_is_
     zero_address (gas: ~168)
[PASS] tests::unit::LOAN_test::test_should_delete_stored_loan_contract (gas: ~635)
[PASS] tests::unit::LOAN_test::test_should_store_loan_contract_under_loan_id (gas: ~
     818)
[PASS] tests::unit::config_test::set_fee_collector::test_should_fail_when_caller_is_
     not_owner (gas: ~366)
[PASS] tests::unit::LOAN_test::test_should_burn_loan_token (gas: ~636)
[PASS] tests::unit::config_test::set_default_loan_metadata_uri::test_should_emit_eve
     nt_default_loan_metadata_uri_updated (gas: ~498)
[PASS] tests::unit::config_test::loan_metadata_uri::test_should_return_default_loan_
     metadata_uri_when_no_store_value_for_loan_contract (gas: ~500)
[PASS] tests::unit::fee_calculator_test::test_should_return_correct_value_for_zero_f
     ee (gas: ~1)
[PASS] tests::unit::config_test::loan_metadata_uri::test_should_return_loan_metadata
     _uri_when_stored_value_for_loan_contract (gas: ~633)
[PASS] tests::unit::fee_calculator_test::test_should_return_correct_value_for_non_ze
     ro_fee (gas: ~1)
[PASS] tests::unit::fee_calculator_test::test_should_handle_small_amount (gas: ~1)
[PASS] tests::unit::fee_calculator_test::test_should_handle_zero_amount (gas: ~1)
[PASS] tests::unit::hub_test::constructor::test_should_set_hub_owner (gas: ~167)
[PASS] tests::unit::hub_test::set_tag::test_should_fail_when_caller_is_not_owner (ga
     s: ~169)
[PASS] tests::unit::fee_calculator_test::test_fuzz_fee_and_new_loan_amount_are_eq_to
     _original_loan_amount (gas: ~1)
[PASS] tests::unit::hub_test::set_tag::test_should_remove_tag_from_address (gas: ~17
     5)
[PASS] tests::unit::hub_test::set_tag::test_should_add_tag_to_address (gas: ~235)
[PASS] tests::unit::hub_test::set_tag::test_should_emit_event_tag_set (gas: ~235)
[PASS] tests::unit::hub_test::set_tags::test_should_not_fail_when_empty_list (gas: ~
     170)
[PASS] tests::unit::hub_test::set_tags::test_should_fail_when_caller_is_not_owner (g
     as: ~170)
[PASS] tests::unit::hub_test::set_tags::test_should_fail_when_diff_input_lengths (ga
     s: ~170)
[PASS] tests::unit::hub_test::set_tags::test_should_emit_event_tag_set_for_every_set
      (gas: ~308)
[PASS] tests::unit::hub_test::set_tags::test_should_add_tags_to_address (gas: ~307)
[PASS] tests::unit::hub_test::set_tags::test_should_remove_tags_from_address (gas: ~
     187)
[PASS] tests::unit::hub_test::has_tag::test_should_return_false_when_address_does_no
     t_have_tag (gas: ~235)
[PASS] tests::unit::hub_test::has_tag::test_should_return_true_when_address_does_hav
     e_tag (gas: ~167)
[PASS] tests::unit::multitoken_category_registry_test::register_category_value::test
     _should_fail_when_caller_is_not_owner (gas: ~169)
[PASS] tests::unit::multitoken_category_registry_test::constructor::test_should_set_
     contract_owner (gas: ~167)
[PASS] tests::unit::multitoken_category_registry_test::register_category_value::test
     _should_fail_when_category_max_u8_value (gas: ~168)
[PASS] tests::unit::config_test::initialize::test_should_set_values (gas: ~363)
[PASS] tests::unit::LOAN_test::test_should_call_loan_contract_and_return_correct_val
     ue (gas: ~820)
[PASS] tests::unit::LOAN_test::test_should_emit_event_loan_burned (gas: ~638)
[PASS] tests::unit::multitoken_library_test::test_should_fail_when_erc20_when_source
     _is_not_this_when_call_to_non_contract_address (gas: ~3)
```

```
[PASS] tests::unit::multitoken_library_test::test_should_call_safe_transfer_from_whe
     n_erc721 (gas: ~1677)
[PASS] tests::unit::config_test::set_fee_collector::test_should_emit_event_fee_colle
     ctor_updated (gas: ~369)
[PASS] tests::unit::multitoken_library_test::test_should_call_transfer_from_when_erc
     721 (gas: ~1216)
[PASS] tests::unit::multitoken_library_test::test_should_call_safe_transfer_from_whe
     n_erc1155 (gas: ~1746)
[PASS] tests::unit::multitoken_library_test::test_should_return_balance_of_erc20 (ga
     s: ~1209)
[PASS] tests::unit::multitoken_library_test::test_should_set_amount_to_one_when_erc1
     155_with_zero_amount (gas: ~1746)
[PASS] tests::unit::multitoken_library_test::test_should_return_balance_of_erc721 (g
     as: ~1204)
[PASS] tests::unit::multitoken_library_test::test_should_return_balance_of_erc1155 (
     gas: ~1596)
[PASS] tests::unit::multitoken_library_test::test_erc20_transfer_asset_from_should_s
     ucceed_when_approved (gas: ~1232)
[PASS] tests::unit::multitoken_library_test::test_erc20_transfer_asset_from_should_f
     ail_when_not_approved (gas: ~1210)
[PASS] tests::unit::multitoken_library_test::test_erc721_transfer_asset_from_should_
     fail_when_not_approved (gas: ~1206)
[PASS] tests::unit::multitoken_library_test::test_erc1155_transfer_asset_from_should
     _fail_when_not_approved (gas: ~1597)
[PASS] tests::unit::multitoken_library_test::test_erc721_transfer_asset_from_should_
     succeed_when_approved (gas: ~1679)
[PASS] tests::unit::multitoken_library_test::test_is_valid_with_registry_should_retu
     rn_true_when_category_and_format_check_return_true (gas: ~8)
[PASS] tests::unit::multitoken_library_test::test_should_fail_when_different_categor
     y (gas: ~1)
[PASS] tests::unit::multitoken_library_test::test_is_valid_without_registry_should_r
     eturn_true_when_category_and_format_check_return_true (gas: ~8)
[PASS] tests::unit::multitoken_library_test::test_erc1155_transfer_asset_from_should
     _succeed_when_approved (gas: ~1752)
[PASS] tests::unit::multitoken_library_test::test_should_fail_when_different_address
      (gas: ~1)
[PASS] tests::unit::multitoken_library_test::test_should_fail_when_different_id (gas
     : ~1)
[PASS] tests::unit::multitoken_library_test::test_should_pass_when_different_amount
     (gas: ~1)
[PASS] tests::unit::LOAN_test::test_should_fail_when_caller_is_not_active_loan_contr
     act (gas: ~611)
[PASS] tests::unit::config_test::initialize::test_should_fail_when_called_second_tim
     e (gas: ~367)
[PASS] tests::unit::config_test::initialize::test_should_fail_when_owner_is_zero_add
     ress (gas: ~102)
[PASS] tests::unit::config_test::set_fee::test_should_fail_when_new_value_bigger_tha
     n_max_fee (gas: ~369)
[PASS] tests::unit::config_test::set_fee::test_should_set_fee_value (gas: ~368)
[PASS] tests::unit::config_test::set_default_loan_metadata_uri::test_should_store_de
     fault_loan_metadata_uri (gas: ~497)
[PASS] tests::unit::config_test::set_loan_metadata_uri::test_should_store_loan_metad
     ata_uri_to_loan_contract (gas: ~500)
[PASS] tests::unit::LOAN_test::test_should_mint_loan_token (gas: ~820)
[PASS] tests::unit::config_test::set_fee::test_should_fail_when_caller_is_not_owner
     (gas: ~366)
[PASS] tests::unit::config_test::set_fee_collector::test_should_fail_when_setting_ze
     ro_address (gas: ~367)
```

```
[PASS] tests::unit::multitoken_category_registry_test::register_category_value::test
     _fuzz_should_emit_CategoryRegistered (runs: 256, gas: {max: ~235, min: ~1, mean
     : ~233.00, std deviation: ~20.60})
[PASS] tests::unit::revoked_nonce_test::revoke_nonce_with_owner::test_fuzz_should_fa
     il_when_caller_does_not_have_access_tag (runs: 256, gas: {max: ~549, min: ~518,
      mean: ~548.00, std deviation: ~2.07})
[PASS] tests::unit::config_test::set_loan_metadata_uri::test_should_emit_event_loan_
     metadata_uri_updated (gas: ~498)
[PASS] tests::unit::multitoken_category_registry_test::unregister_category_value::te
     st_should_fail_when_caller_is_not_owner (gas: ~169)
[PASS] tests::unit::revoked_nonce_test::revoke_nonce::test_fuzz_should_fail_when_non
     ce_already_revoked (runs: 256, gas: {max: ~652, min: ~528, mean: ~650.00, std d
     eviation: ~8.59})
[PASS] tests::unit::revoked_nonce_test::revoke_nonces::test_fuzz_should_fail_when_an
     y_nonce_already_revoked (runs: 256, gas: {max: ~652, min: ~529, mean: ~651.00,
     std deviation: ~8.52})
[PASS] tests::unit::revoked_nonce_test::revoke_nonce::test_fuzz_should_emit_nonce_re
     voked (runs: 256, gas: {max: ~529, min: ~465, mean: ~528.00, std deviation: ~4.
     06})
[PASS] tests::unit::LOAN_test::test_should_increase_last_loan_id (gas: ~819)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_return_used
     _credit (gas: ~1186)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_revoke_nonc
     e (gas: ~1196)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_o
     ffer_nonce_not_usable (gas: ~1242)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_return_prop
     osal_hash (gas: ~1126)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_c
     aller_is_not_proposer (gas: ~1137)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_c
     aller_is_not_allowed_acceptor (gas: ~1245)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_emit_propos
     al_made (gas: ~1203)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_revoke_offe
     r_when_available_credit_limit_equal_to_zero (gas: ~1305)
[PASS] tests::unit::revoked_nonce_test::revoke_nonce::test_fuzz_should_store_nonce_a
     s_revoked (runs: 256, gas: {max: ~530, min: ~466, mean: ~529.00, std deviation:
      ~4.06})
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_u
     sed_credit_exceeds_available_credit_limit (gas: ~1230)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_increase_us
     ed_credit_when_used_credit_not_exceeds_available_credit_limit (gas: ~1297)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_c
     omputer_registry_returns_computer_when_computer_returns_different_state_fingerp
     rint (gas: ~1426)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_pass_when_c
     omputer_returns_matching_fingerprint (gas: ~1373)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_call_loan_c
     ontract_with_loan_terms (gas: ~1490)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_return_used_cred
     it (gas: ~1186)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_revoke_nonce (ga
     s: ~1196)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_fail_when_caller
     _is_not_proposer (gas: ~1137)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_return_proposal_
     hash (gas: ~1126)
```

```
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_emit_proposal_ma
     de (gas: ~1202)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_make_proposal (g
     as: ~1197)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_return_encoded_p
     roposal_data (gas: ~1133)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_return_decoded_p
     roposal_data (gas: ~1139)
[PASS] tests::unit::revoked_nonce_test::is_nonce_revoked::test_fuzz_should_return_st
     ored_value (runs: 256, gas: {max: ~460, min: ~396, mean: ~429.00, std deviation
     : ~31.98})
[PASS] tests::unit::revoked_nonce_test::revoke_nonces::test_fuzz_should_store_nonces
     _as_revoked (runs: 256, gas: {max: ~666, min: ~602, mean: ~665.00, std deviatio
     n: ~4.06})
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_make_propos
     al (gas: ~1198)
[PASS] tests::unit::revoked_nonce_test::is_nonce_usable::test_fuzz_should_return_fal
     se_when_nonce_space_is_not_equal_to_current_nonce_space (runs: 256, gas: {max:
     ~460, min: ~396, mean: ~459.00, std deviation: ~4.06})
[PASS] tests::unit::revoked_nonce_test::is_nonce_usable::test_fuzz_should_return_fal
     se_when_nonce_is_revoked (runs: 256, gas: {max: ~458, min: ~458, mean: ~458.00,
      std deviation: ~0.00})
[PASS] tests::unit::multitoken_category_registry_test::unregister_category_value::te
     st_fuzz_should_emit_CategoryUnregistered (runs: 256, gas: {max: ~170, min: ~170
     , mean: ~170.00, std deviation: ~0.00})
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_fail_when_refina
     ncing_loan_ids_is_not_equal_when_proposed_refinancing_loan_id_not_zero_when_ref
     inancing_loan_id_not_zero_when_offer (gas: ~1221)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_return_enco
     ded_proposal_data (gas: ~1135)
[PASS] tests::unit::revoked_nonce_test::revoke_nonce_with_nonce_space::test_fuzz_sho
     uld_fail_when_nonce_already_revoked (runs: 256, gas: {max: ~587, min: ~527, mea
     n: ~586.00, std deviation: ~5.27})
[PASS] tests::unit::revoked_nonce_test::is_nonce_usable::test_fuzz_should_return_tru
     e_when_nonce_space_is_equal_to_current_nonce_space_when_nonce_is_not_revoked (r
     uns: 256, gas: {max: ~460, min: ~396, mean: ~459.00, std deviation: ~4.06})
[PASS] tests::unit::revoked_nonce_test::revoke_nonce_with_nonce_space::test_fuzz_sho
     uld_store_nonce_as_revoked (runs: 256, gas: {max: ~465, min: ~465, mean: ~465.0
     0, std deviation: ~0.00})
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_pass_when_refina
     ncing_loan_ids_not_equal_when_proposed_refinancing_loan_id_zero_when_refinancin
     g_loan_id_not_zero_when_offer (gas: ~1296)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_return_deco
     ded_proposal_data (gas: ~1141)
[PASS] tests::unit::revoked_nonce_test::revoke_nonce_space::test_fuzz_should_increme
     nt_current_nonce_space (runs: 256, gas: {max: ~465, min: ~401, mean: ~464.00, s
     td deviation: ~4.06})
[PASS] tests::unit::revoked_nonce_test::revoke_nonce_with_nonce_space::test_fuzz_sho
     uld_emit_nonce_revoked (runs: 256, gas: {max: ~464, min: ~464, mean: ~464.00, s
     td deviation: ~0.00})
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_fail_when_offer_
     nonce_not_usable (gas: ~1237)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_increase_used_cr
     edit_when_used_credit_not_exceeds_available_credit_limit (gas: ~1296)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_fail_when_propos
     al_expired (gas: ~1225)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_fail_when_used_c
     redit_exceeds_available_credit_limit (gas: ~1230)
```

```
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_revoke_offer_whe
    n_available_credit_limit_equal_to_zero (gas: ~1301)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_fail_when_comput
    er_registry_returns_computer_when_computer_returns_different_state_fingerprint
    (gas: ~1420)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_fail_when_caller
    _is_not_allowed_acceptor (gas: ~1240)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_call_loan_contra
    ct_with_loan_terms (gas: ~1480)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_revoke_nonce (gas: ~
    1196)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_return_used_credit (
    gas: ~1186)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_return_proposal_hash
     (gas: ~1126)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_pass_when_comput
    er_returns_matching_fingerprint (gas: ~1369)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_emit_proposal_made (
    gas: ~1202)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_make_proposal (gas:
    ~1197)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_fail_when_caller_is_
    not_proposer (gas: ~1137)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_return_encoded_propo
    sal_data (gas: ~1134)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_accept_any_collatera
    l_id_when_merkle_root_is_zero (gas: ~1296)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_return_decoded_propo
    sal_data (gas: ~1140)
[PASS] tests::unit::revoked_nonce_test::revoke_nonce_space::test_fuzz_should_emit_no
    nce_space_revoked (runs: 256, gas: {max: ~465, min: ~401, mean: ~464.00, std de
    viation: ~4.06})
[PASS] tests::unit::revoked_nonce_test::current_nonce_space::test_fuzz_should_return
    _current_nonce_space (runs: 256, gas: {max: ~460, min: ~396, mean: ~459.00, std
     deviation: ~4.06})
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_fail_when_refina
    ncing_loan_ids_not_equal_when_refinancing_loan_id_not_zero_when_request (gas: ~
    1222)
[PASS] tests::unit::config_test::set_default_loan_metadata_uri::test_should_fail_whe
    n_caller_is_not_owner (gas: ~366)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_pass_when_computer_r
    eturns_matching_fingerprint (gas: ~1369)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_return_used_credit
     (gas: ~1186)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_call_loan_contract_w
    ith_loan_terms (gas: ~1477)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_fail_when_proposal
    _expired (gas: ~1218)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_return_proposal_ha
    sh (gas: ~1126)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_fail_when_offer_no
    nce_not_usable (gas: ~1230)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_revoke_nonce (gas:
     ~1196)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_fail_when_caller_i
    s_not_proposer (gas: ~1137)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_fail_when_caller_i
    s_not_allowed_acceptor (gas: ~1233)
```

```
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_make_proposal (gas
     : ~1197)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_emit_proposal_made
     (gas: ~1202)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_revoke_offer_when_
     available_credit_limit_equal_to_zero (gas: ~1294)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_increase_used_cred
     it_when_used_credit_not_exceeds_available_credit_limit (gas: ~1286)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_return_encoded_pro
     posal_data (gas: ~1127)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_fail_when_computer
     _registry_returns_computer_when_computer_returns_different_state_fingerprint (g
     as: ~1405)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_pass_when_computer
     _returns_matching_fingerprint (gas: ~1362)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_fail_when_used_cre
     dit_exceeds_available_credit_limit (gas: ~1220)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_return_decoded_pro
     posal_data (gas: ~1132)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_fail_when_caller_i
     s_not_proposed_loan_contract (gas: ~1232)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_fail_when_caller_n
     ot_tagged_active_loan (gas: ~1220)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_fail_when_proposer
     _is_same_as_acceptor (gas: ~1221)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_fail_when_proposed
     _refinancing_loan_id_not_zero_when_refinancing_loan_id_zero (gas: ~1214)
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_return_proposal_ha
     sh_and_loan_terms (gas: ~1460)
[PASS] tests::unit::simple_loan_test::get_lender_spec_hash::test_should_return_lende
     r_spec_hash (gas: ~5175)
[PASS] tests::unit::revoked_nonce_test::revoke_nonce_space::test_fuzz_should_return_
     new_nonce_space (runs: 256, gas: {max: ~463, min: ~399, mean: ~462.00, std devi
     ation: ~4.06})
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_fail_when_refinanc
     ing_loan_ids_is_not_equal_when_proposed_refinancing_loan_id_not_zero_when_refin
     ancing_loan_id_not_zero_when_offer (gas: ~1214)
[PASS] tests::unit::simple_loan_test::create_loan::test_fuzz_should_fail_when_propos
     al_contract_not_tagged_loan_proposal (runs: 256, gas: {max: ~5219, min: ~5206,
     mean: ~5218.00, std deviation: ~1.25})
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_pass_when_refinanc
     ing_loan_ids_not_equal_when_proposed_refinancing_loan_id_zero_when_refinancing_
     loan_id_not_zero_when_offer (gas: ~1289)
[PASS] tests::unit::simple_loan_test::create_loan::test_should_fail_when_pool_adapte
     r_not_registered_when_pool_source_of_funds (gas: ~6120)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_i
     nvalid_auction_duration (gas: ~1125)
[PASS] tests::unit::revoked_nonce_test::revoke_nonce_with_owner::test_fuzz_should_fa
     il_when_nonce_already_revoked (runs: 256, gas: {max: ~771, min: ~592, mean: ~76
     8.00, std deviation: ~14.51})
[PASS] tests::unit::LOAN_test::test_should_return_loan_id (gas: ~818)
[PASS] tests::unit::simple_loan_test::create_loan::test_should_emit_loan_created (ga
     s: ~6148)
[PASS] tests::unit::LOAN_test::test_should_emit_event_loan_minted (gas: ~821)
[PASS] tests::unit::simple_loan_test::refinance_loan::test_should_fail_when_refinanc
     ing_disabled (gas: ~6957)
[PASS] tests::unit::simple_loan_test::create_loan::test_fuzz_should_return_new_loan_
     id (runs: 256, gas: {max: ~5872, min: ~5872, mean: ~5872.00, std deviation: ~0.
    00})
[PASS] tests::unit::simple_loan_test::repay_loan::test_should_fail_when_loan_does_no
    t_exist (gas: ~5970)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_fail_when_zero_m
    in_collateral_amount (gas: ~1217)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_fail_when_collat
    eral_amount_less_than_min_collateral_amount (gas: ~1221)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_fail_when_propos
    ed_refinancing_loan_id_not_zero_when_refinancing_loan_id_zero (gas: ~1221)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_fuzz_should_return_cred
    it_amount (runs: 256, gas: {max: ~1123, min: ~1123, mean: ~1123.00, std deviati
    on: ~0.00})
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_fail_when_propos
    er_is_same_as_acceptor (gas: ~1228)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_fail_when_caller
    _is_not_proposed_loan_contract (gas: ~1246)
[PASS] tests::unit::simple_loan_fungible_proposal_test::test_should_fail_when_caller
    _not_tagged_active_loan (gas: ~1234)
[PASS] tests::unit::simple_loan_test::repay_loan::test_fuzz_should_fail_when_loan_is
    _not_running (runs: 256, gas: {max: ~6034, min: ~6034, mean: ~6034.00, std devi
    ation: ~0.00})
[PASS] tests::unit::simple_loan_test::repay_loan::test_should_fail_when_loan_is_defa
    ulted (gas: ~6033)
[PASS] tests::unit::simple_loan_test::claim_loan::test_should_emit_loan_claimed_when
    _repaid (gas: ~5180)
[PASS] tests::unit::simple_loan_test::claim_loan::test_should_emit_loan_claimed_when
    _defaulted (gas: ~5187)
[PASS] tests::unit::revoked_nonce_test::revoke_nonce_with_owner::test_fuzz_should_st
    ore_nonce_as_revoked (runs: 256, gas: {max: ~601, min: ~537, mean: ~600.00, std
     deviation: ~4.06})
[PASS] tests::unit::simple_loan_test::claim_loan::test_fuzz_should_fail_when_caller_
    is_not_loan_token_holder (runs: 256, gas: {max: ~6033, min: ~6033, mean: ~6033.
    00, std deviation: ~0.00})
[PASS] tests::unit::simple_loan_test::claim_loan::test_should_fail_when_loan_does_no
    t_exist (gas: ~5973)
[PASS] tests::unit::multitoken_category_registry_test::registered_category_value::te
    st_fuzz_should_return_category_not_registered_when_not_registered (runs: 256, g
    as: {max: ~167, min: ~167, mean: ~167.00, std deviation: ~0.00})
[PASS] tests::unit::simple_loan_test::claim_loan::test_should_fail_when_loan_is_not_
    repaid_nor_expired (gas: ~6038)
[PASS] tests::unit::simple_loan_test::claim_loan::test_should_pass_when_loan_is_defa
    ulted (gas: ~5182)
[PASS] tests::unit::simple_loan_test::claim_loan::test_should_pass_when_loan_is_repa
    id (gas: ~5175)
[PASS] tests::unit::simple_loan_test::claim_loan::test_should_delete_loan_data (gas:
     ~5180)
[PASS] tests::unit::simple_loan_test::claim_loan::test_should_burn_loan_token (gas:
    ~5176)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_pass_when_given_coll
    ateral_id_is_whitelisted (runs: 256, gas: {max: ~5509, min: ~5499, mean: ~5508.
    00, std deviation: ~1.19})
[PASS] tests::unit::revoked_nonce_test::revoke_nonce_with_owner::test_fuzz_should_em
    it_nonce_revoked (runs: 256, gas: {max: ~600, min: ~536, mean: ~599.00, std dev
    iation: ~4.06})
[PASS] tests::unit::revoked_nonce_test::revoke_nonce_with_nonce_space_and_owner::tes
    t_fuzz_should_fail_when_caller_does_not_have_access_tag (runs: 256, gas: {max:
    ~483, min: ~452, mean: ~482.00, std deviation: ~2.07})
[PASS] tests::unit::revoked_nonce_test::revoke_nonce_with_nonce_space_and_owner::tes
```

```
     t_fuzz_should_fail_when_nonce_already_revoked (runs: 256, gas: {max: ~676, min:
      ~616, mean: ~674.00, std deviation: ~5.61})
[PASS] tests::unit::multitoken_library_test::test_fuzz_should_return_erc20 (runs: 25
     6, gas: {max: ~1, min: ~1, mean: ~1.00, std deviation: ~0.00})
[PASS] tests::unit::simple_loan_test::claim_loan::test_fuzz_should_transfer_repaid_a
     mount_to_lender_when_loan_is_repaid (runs: 256, gas: {max: ~5194, min: ~5194, m
     ean: ~5194.00, std deviation: ~0.00})
[PASS] tests::unit::simple_loan_test::claim_loan::test_should_transfer_collateral_to
     _lender_when_loan_is_defaulted (gas: ~5187)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_fail_when_caller_not
     _tagged_active_loan (gas: ~1231)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_fail_when_proposer_i
     s_same_as_acceptor (gas: ~1228)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_fail_when_caller_is_
     not_proposed_loan_contract (gas: ~1243)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_fail_when_proposed_r
     efinancing_loan_id_not_zero_when_refinancing_loan_id_zero (gas: ~1222)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_fail_when_refinancin
     g_loan_ids_is_not_equal_when_proposed_refinancing_loan_id_not_zero_when_refinan
     cing_loan_id_not_zero_when_offer (gas: ~1222)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_fail_when_offer_nonc
     e_not_usable (gas: ~1238)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_fail_when_proposal_e
     xpired (gas: ~1225)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_fail_when_caller_is_
     not_allowed_acceptor (gas: ~1240)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_fail_when_given_coll
     ateral_id_is_not_whitelisted (runs: 256, gas: {max: ~5464, min: ~5430, mean: ~5
     463.00, std deviation: ~2.20})
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_revoke_offer_when_av
     ailable_credit_limit_equal_to_zero (gas: ~1301)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_fail_when_refinancin
     g_loan_ids_not_equal_when_refinancing_loan_id_not_zero_when_request (gas: ~1222
     )
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_pass_when_refinancin
     g_loan_ids_not_equal_when_proposed_refinancing_loan_id_zero_when_refinancing_lo
     an_id_not_zero_when_offer (gas: ~1296)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_increase_used_credit
     _when_used_credit_not_exceeds_available_credit_limit (gas: ~1293)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_fail_when_used_credi
     t_exceeds_available_credit_limit (gas: ~1228)
[PASS] tests::unit::simple_loan_list_proposal_test::test_should_fail_when_computer_r
     egistry_returns_computer_when_computer_returns_different_state_fingerprint (gas
     : ~1417)
[PASS] tests::unit::simple_loan_test::try_claim_repaid_loan::test_should_fail_when_t
     ransfer_fails_when_source_of_funds_equal_to_original_lender (gas: ~5821)
[PASS] tests::unit::simple_loan_test::try_claim_repaid_loan::test_should_fail_when_t
     ransfer_fails_when_source_of_funds_not_equal_to_original_lender (gas: ~5823)
[PASS] tests::unit::simple_loan_test::extend_loan::test_should_fail_when_proposal_ex
     pirated (gas: ~5968)
[PASS] tests::unit::simple_loan_test::try_claim_repaid_loan::test_should_emit_loan_c
     laimed (gas: ~5891)
[PASS] tests::unit::simple_loan_test::try_claim_repaid_loan::test_should_call_supply
     _on_pool_adapter_when_source_of_funds_not_equal_to_original_lender (gas: ~5906)
[PASS] tests::unit::simple_loan_test::extend_loan::test_should_fail_when_offer_nonce
     _not_usable (gas: ~5995)
[PASS] tests::unit::simple_loan_test::repay_loan::test_fuzz_should_update_loan_data_
     when_loan_owner_is_not_original_lender (runs: 256, gas: {max: ~6367, min: ~6303
     , mean: ~6364.00, std deviation: ~13.53})
[PASS] tests::unit::simple_loan_test::try_claim_repaid_loan::test_fuzz_should_not_pr
     oceed_when_loan_not_in_repaid_state (runs: 256, gas: {max: ~6765, min: ~6701, m
     ean: ~6744.00, std deviation: ~29.77})
[PASS] tests::unit::simple_loan_test::extend_loan::test_should_fail_when_caller_is_n
     ot_borrower_nor_loan_owner (gas: ~5939)
[PASS] tests::unit::simple_loan_test::try_claim_repaid_loan::test_should_burn_loan_t
     oken (gas: ~5887)
[PASS] tests::unit::simple_loan_test::try_claim_repaid_loan::test_fuzz_should_not_pr
     oceed_when_original_lender_not_equal_to_loan_owner (runs: 256, gas: {max: ~6760
     , min: ~6760, mean: ~6760.00, std deviation: ~0.00})
[PASS] tests::unit::simple_loan_test::try_claim_repaid_loan::test_should_fail_when_p
     ool_adapter_not_registered_when_source_of_funds_not_equal_to_original_lender (g
     as: ~5891)
[PASS] tests::unit::simple_loan_test::try_claim_repaid_loan::test_should_delete_loan
     _data (gas: ~5898)
[PASS] tests::unit::simple_loan_test::try_claim_repaid_loan::test_should_transfer_to
     _original_lender_when_source_of_funds_equal_to_original_lender (gas: ~5896)
[PASS] tests::unit::simple_loan_test::try_claim_repaid_loan::test_should_transfer_am
     ount_to_pool_adapter_when_source_of_funds_not_equal_to_original_lender (gas: ~5
     906)
[PASS] tests::unit::simple_loan_test::try_claim_repaid_loan::test_should_not_call_tr
     ansfer_when_credit_amount_is_zero (gas: ~5883)
[PASS] tests::unit::simple_loan_test::extend_loan::test_should_fail_when_caller_is_b
     orrower_and_proposer_is_not_loan_owner (gas: ~6022)
[PASS] tests::unit::simple_loan_test::extend_loan::test_should_fail_when_caller_is_l
     oan_owner_and_proposer_is_not_borrower (gas: ~6022)
[PASS] tests::unit::simple_loan_test::extend_loan::test_should_not_transfer_credit_w
     hen_amount_zero (gas: ~6050)
[PASS] tests::unit::simple_loan_test::extend_loan::test_should_fail_when_extension_d
     uration_more_than_max (gas: ~5973)
[PASS] tests::unit::simple_loan_test::extend_loan::test_should_fail_when_invalid_com
     pensation_asset (gas: ~6061)
[PASS] tests::unit::simple_loan_test::extend_loan::test_should_revoke_extension_nonc
     e (gas: ~6047)
[PASS] tests::unit::simple_loan_test::extend_loan::test_should_fail_when_extension_d
     uration_less_than_min (gas: ~5972)
[PASS] tests::unit::simple_loan_test::extend_loan::test_should_not_transfer_credit_w
     hen_address_zero (gas: ~6050)
[PASS] tests::unit::simple_loan_test::extend_loan::test_should_transfer_compensation
     _when_defined (gas: ~5954)
[PASS] tests::unit::multitoken_library_test::test_fuzz_should_return_erc721 (runs: 2
     56, gas: {max: ~1, min: ~1, mean: ~1.00, std deviation: ~0.00})
[PASS] tests::unit::simple_loan_test::create_loan::test_fuzz_should_revoke_callers_n
     once_when_flag_is_true (runs: 256, gas: {max: ~6214, min: ~6214, mean: ~6214.00
     , std deviation: ~0.00})
[PASS] tests::unit::revoked_nonce_test::revoke_nonce_with_nonce_space_and_owner::tes
     t_fuzz_should_store_nonce_as_revoked (runs: 256, gas: {max: ~536, min: ~536, me
     an: ~536.00, std deviation: ~0.00})
[PASS] tests::unit::simple_loan_simple_proposal_test::test_should_fail_when_refinanc
     ing_loan_ids_not_equal_when_refinancing_loan_id_not_zero_when_request (gas: ~12
     14)
[PASS] tests::unit::simple_loan_test::get_loan::test_should_return_correct_status (g
     as: ~6081)
[PASS] tests::unit::simple_loan_test::create_loan::test_fuzz_should_call_withdraw_wh
     en_pool_source_of_funds (runs: 256, gas: {max: ~6229, min: ~6101, mean: ~6226.0
     0, std deviation: ~19.36})
[PASS] tests::unit::simple_loan_test::make_extension_proposal::test_fuzz_should_fail
```

```
    _when_caller_not_proposer (runs: 256, gas: {max: ~5218, min: ~5205, mean: ~5217
    .00, std deviation: ~1.23})
[PASS] tests::unit::multitoken_library_test::test_fuzz_should_return_erc1155 (runs:
    256, gas: {max: ~1, min: ~1, mean: ~1.00, std deviation: ~0.00})
[PASS] tests::unit::simple_loan_test::get_loan::test_should_return_empty_loan_data_f
    or_non_existing_loan (gas: ~5192)
[PASS] tests::unit::simple_loan_test::get_state_fingerprint::test_should_return_zero
    _if_loan_does_not_exist (gas: ~5180)
[PASS] tests::unit::simple_loan_test::loan_metadata_uri::test_should_return_correct_
    value (gas: ~5184)
[PASS] tests::unit::simple_loan_test::get_state_fingerprint::test_should_update_stat
    e_fingerprint_when_loan_defaulted (gas: ~5911)
[PASS] tests::unit::revoked_nonce_test::revoke_nonces::test_fuzz_should_emit_nonce_r
    evoked (runs: 256, gas: {max: ~670, min: ~606, mean: ~669.00, std deviation: ~4
    .06})
[PASS] tests::unit::vault_test::pwn_vault_withdraw_from_pool_test::test_should_call_
    withdraw_on_pool_adapter (gas: ~1513)
[PASS] tests::unit::vault_test::pwn_vault_push_from_test::test_should_fail_when_inco
    mplete_transaction (gas: ~1502)
[PASS] tests::unit::vault_test::pwn_vault_push_from_test::test_should_call_safe_tran
    sfer_from_from_origin_to_beneficiary (gas: ~1670)
[PASS] tests::unit::vault_test::pwn_vault_push_test::test_should_fail_when_incomplet
    e_transaction (gas: ~1273)
[PASS] tests::unit::vault_test::pwn_vault_withdraw_from_pool_test::test_should_fail_
    when_incomplete_transaction (gas: ~1415)
[PASS] tests::unit::vault_test::pwn_vault_supply_to_pool_test::test_should_transfer_
    asset_to_pool_adapter (gas: ~1435)
[PASS] tests::unit::vault_test::pwn_vault_supply_to_pool_test::test_should_fail_when
    _incomplete_transaction (gas: ~1417)
[PASS] tests::integration::protocol_integrity_test::test_should_fail_to_create_loan_
    when_loan_contract_not_active (gas: ~5994)
[PASS] tests::unit::vault_test::pwn_vault_pull_test::test_should_call_transfer_from_
    from_origin_to_vault (gas: ~1432)
[PASS] tests::unit::vault_test::pwn_vault_push_test::test_should_call_safe_transfer_
    from_from_vault_to_beneficiary (gas: ~1432)
[PASS] tests::unit::vault_test::pwn_vault_pull_test::test_should_fail_when_incomplet
    e_transaction (gas: ~1273)
[PASS] tests::unit::simple_loan_test::get_state_fingerprint::test_fuzz_should_return
    _correct_state_fingerprint (runs: 256, gas: {max: ~6023, min: ~5895, mean: ~602
    2.00, std deviation: ~8.92})
[PASS] tests::integration::simple_loan_integration_test::test_should_create_loan_fro
    m_fungible_proposal (gas: ~7763)
[PASS] tests::integration::protocol_integrity_test::test_should_fail_to_create_loan_
    terms_when_caller_is_not_active_loan (gas: ~6002)
[PASS] tests::integration::simple_loan_integration_test::test_should_create_loan_fro
    m_list_proposal (gas: ~1)
[PASS] tests::integration::protocol_integrity_test::test_should_claim_repaid_loan_wh
    en_loan_contract_not_active (gas: ~6679)
[PASS] tests::integration::protocol_integrity_test::test_should_fail_to_create_loan_
    when_passing_invalid_terms_factory_contract (gas: ~6418)
[PASS] tests::integration::simple_loan_integration_test::test_should_create_loan_fro
    m_simple_proposal (gas: ~7563)
[PASS] tests::integration::simple_loan_integration_test::test_should_create_loan_wit
    h_erc721_collateral (gas: ~7487)
[PASS] tests::integration::simple_loan_integration_test::test_should_create_loan_wit
    h_erc20_collateral (gas: ~7433)
[PASS] tests::integration::simple_loan_integration_test::test_should_create_loan_wit
    h_erc1155_collateral (gas: ~7561)
```

```
[PASS] tests::integration::simple_loan_integration_test::test_should_create_loan_fro
     m_dutch_auction_proposal (gas: ~7772)
[PASS] tests::unit::simple_loan_test::create_loan::test_fuzz_should_not_revoke_calle
     rs_nonce_when_flag_is_false (runs: 256, gas: {max: ~6144, min: ~6144, mean: ~61
     44.00, std deviation: ~0.00})
[PASS] tests::integration::protocol_integrity_test::test_should_repay_loan_when_loan
     _contract_not_active_when_original_lender_is_not_loan_owner (gas: ~7590)
[PASS] tests::integration::simple_loan_integration_test::test_should_claim_defaulted
     _loan (gas: ~6648)
[PASS] tests::unit::simple_loan_test::make_extension_proposal::test_should_emit_exte
     nsion_proposal_made (gas: ~5248)
[PASS] tests::integration::simple_loan_integration_test::test_should_fail_to_repay_l
     oan_when_loan_expired (gas: ~7629)
[PASS] tests::unit::simple_loan_test::extend_loan::test_should_fail_when_loan_does_n
     ot_exist (gas: ~5254)
[PASS] tests::unit::multitoken_library_test::test_should_fail_when_erc20_when_source
     _is_this_when_call_to_non_contract_address (gas: ~3)
[PASS] tests::unit::simple_loan_test::make_extension_proposal::test_should_store_mad
     e_flag (gas: ~5246)
[PASS] tests::unit::simple_loan_test::extend_loan::test_should_fail_when_loan_is_rep
     aid (gas: ~5963)
[PASS] tests::unit::multitoken_library_test::test_should_call_transfer_when_erc20_wh
     en_source_is_this (gas: ~1218)
[PASS] tests::unit::simple_loan_test::extend_loan::test_should_update_loan_data (gas
     : ~6109)
[PASS] tests::unit::multitoken_library_test::test_should_fail_when_erc20_when_source
     _is_this_when_transfer_returns_fale (gas: ~1075)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_c
     urrent_auction_credit_amount_not_in_intended_credit_amount_range_when_offer (ga
     s: ~1229)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_a
     uction_duration_not_in_full_minutes (gas: ~1124)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_i
     nvalid_credit_amount_range (gas: ~1127)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_p
     roposal_expired (gas: ~1125)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_a
     uction_not_in_progress (gas: ~1125)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_return_corr
     ect_edge_values (gas: ~1151)
[PASS] tests::integration::simple_loan_integration_test::test_should_claim_repaid_lo
     an_when_original_lender_is_not_loan_owner (gas: ~6739)
[PASS] tests::unit::multitoken_library_test::test_should_call_transfer_when_erc20_wh
     en_source_is_not_this (gas: ~1226)
[PASS] tests::unit::multitoken_library_test::test_should_fail_when_erc20_when_source
     _is_not_this_when_transfer_returns_false (gas: ~1076)
[PASS] tests::unit::simple_loan_test::create_loan::test_should_not_fail_when_caller_
     lender_when_lender_spec_hash_mismatch (gas: ~6140)
[PASS] tests::unit::simple_loan_test::create_loan::test_should_fail_when_invalid_col
     lateral_asset (gas: ~5219)
[PASS] tests::unit::simple_loan_test::create_loan::test_fuzz_should_fail_when_loan_t
     erms_duration_less_than_min (runs: 256, gas: {max: ~5187, min: ~5187, mean: ~51
     87.00, std deviation: ~0.00})
[PASS] tests::unit::simple_loan_test::create_loan::test_should_fail_when_invalid_cre
     dit_asset (gas: ~5218)
[PASS] tests::unit::simple_loan_test::create_loan::test_fuzz_should_fail_when_loan_t
     erms_interest_apr_out_of_bounds (runs: 256, gas: {max: ~5190, min: ~5189, mean:
      ~5189.00, std deviation: ~0.86})
```

```
[PASS] tests::unit::simple_loan_test::repay_loan::test_fuzz_should_transfer_repaid_a
     mount_to_vault (runs: 256, gas: {max: ~6366, min: ~6302, mean: ~6363.00, std de
     viation: ~13.53})
[PASS] tests::unit::simple_loan_test::repay_loan::test_should_emit_loan_paid_back (g
     as: ~6085)
[PASS] tests::unit::simple_loan_test::repay_loan::test_should_transfer_collateral_to
     _borrower (gas: ~6082)
[PASS] tests::unit::simple_loan_test::loan_repayment_amount::test_should_return_zero
     _when_loan_does_not_exist (gas: ~5180)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_p
     roposer_is_same_as_acceptor (gas: ~1232)
[PASS] tests::unit::simple_loan_test::create_loan::test_should_transfer_collateral_f
     rom_borrower_to_vault (gas: ~6146)
[PASS] tests::unit::simple_loan_test::create_loan::test_should_mint_loan_token (gas:
      ~6144)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_c
     aller_not_tagged_active_loan (gas: ~1240)
[PASS] tests::unit::simple_loan_test::create_loan::test_should_store_loan_data (gas:
      ~6143)
[PASS] tests::unit::simple_loan_test::get_loan::test_fuzz_should_return_loan_token_o
     wner (runs: 256, gas: {max: ~5903, min: ~5775, mean: ~5902.00, std deviation: ~
     8.00})
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_c
     aller_is_not_proposed_loan_contract (gas: ~1252)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_c
     urrent_auction_credit_amount_not_in_intended_credit_amount_range_when_request (
     gas: ~1229)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_r
     efinancing_loan_ids_is_not_equal_when_proposed_refinancing_loan_id_not_zero_whe
     n_refinancing_loan_id_not_zero_when_offer (gas: ~1226)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_r
     efinancing_loan_ids_not_equal_when_refinancing_loan_id_not_zero_when_request (g
     as: ~1229)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_pass_when_r
     efinancing_loan_ids_not_equal_when_proposed_refinancing_loan_id_zero_when_refin
     ancing_loan_id_not_zero_when_offer (gas: ~1300)
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_fuzz_should_return
     _correct_credit_amount_when_request (runs: 256, gas: {max: ~1125, min: ~1125, m
     ean: ~1125.00, std deviation: ~0.00})
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_should_fail_when_p
     roposed_refinancing_loan_id_not_zero_when_refinancing_loan_id_zero (gas: ~1226)
[PASS] tests::unit::simple_loan_test::loan_repayment_amount::test_should_return_accr
     ued_interest (gas: ~5981)
[PASS] tests::unit::simple_loan_test::get_loan::test_fuzz_should_return_repayment_am
     ount (runs: 256, gas: {max: ~5978, min: ~5912, mean: ~5977.00, std deviation: ~
     6.96})
[PASS] tests::unit::simple_loan_test::loan_repayment_amount::test_fuzz_should_return
     _accrued_interest_when_non_zero_accrued_interest (runs: 256, gas: {max: ~6086,
     min: ~5958, mean: ~6082.00, std deviation: ~15.17})
[PASS] tests::unit::simple_loan_dutch_auction_proposal_test::test_fuzz_should_return
     _correct_credit_amount_when_offer (runs: 256, gas: {max: ~1125, min: ~1125, mea
     n: ~1125.00, std deviation: ~0.00})
[PASS] tests::unit::simple_loan_test::loan_repayment_amount::test_fuzz_should_return
     _fixed_interest_when_zero_accrued_interest (runs: 256, gas: {max: ~6019, min: ~
     5891, mean: ~6013.00, std deviation: ~18.81})
[PASS] tests::unit::revoked_nonce_test::revoke_nonce_with_nonce_space_and_owner::tes
     t_fuzz_should_emit_nonce_revoked (runs: 256, gas: {max: ~535, min: ~535, mean:
     ~535.00, std deviation: ~0.00})
```

```
[PASS] tests::unit::simple_loan_test::create_loan::test_fuzz_should_fail_when_caller
     _not_lender_when_lender_spec_hash_mismatch (runs: 256, gas: {max: ~5239, min: ~
     5213, mean: ~5238.00, std deviation: ~1.82})
[PASS] tests::unit::multitoken_library_test::test_fuzz_should_return_erc1155_with_no
     _amount (runs: 256, gas: {max: ~1, min: ~1, mean: ~1.00, std deviation: ~0.00})
[PASS] tests::unit::simple_loan_test::get_loan::test_fuzz_should_return_static_loan_
     data (runs: 256, gas: {max: ~6291, min: ~6163, mean: ~6288.00, std deviation: ~
     12.55})
[PASS] tests::unit::simple_loan_test::create_loan::test_fuzz_should_transfer_credit_
     to_borrower_and_fee_collector (runs: 256, gas: {max: ~6500, min: ~6164, mean: ~
     6478.00, std deviation: ~46.74})


Tests: 349 passed, 0 failed, 0 skipped, 9 ignored, 0 filtered out
Fuzzer seed: 3229876501507863689
```

A | Disclaimers

The audit makes no statements or warranty about utility of the code, safety of the code, suitability of the business model, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purpose

A.1 | Client Confidentiality

This document contains Client Confidential information and may not be copied without written permission.

A.2 | Proprietary Information

The content of this document should be considered proprietary information. Extropy gives permission to copy this report for the purposes of disseminating information within your organisation or any regulatory agency.

extropy