Firm Money Audit Report

Prepared by Cyfrin

Version 2.0

Lead Auditors

SBSecurity (Blckhv, Slavcheww)

Arnie

March 9, 2026

Contents

1 About Cyfrin

Cyfrin is a Web3 security company dedicated to bringing industry-leading protection and education to our partners and their projects. Our goal is to create a safe, reliable, and transparent environment for everyone in Web3 and DeFi. Learn more about us at cyfrin.io.

2 Disclaimer

The Cyfrin team makes every effort to find as many vulnerabilities in the code as possible in the given time but holds no responsibility for the findings in this document. A security audit by the team does not endorse the underlying business or product. The audit was time-boxed and the review of the code was solely on the security aspects of the solidity implementation of the contracts.

3 Risk Classification

	Impact: High	**Impact: Medium**	**Impact: Low**
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

4 Protocol Summary

Firm Money is a friendly fork of LiquityV2 for the StatusL2 chain. Its stablecoin `$USF` is a 1.00 USD pegged stablecoin minted against ETH and Status Network core tokens like SNT and future native assets. FIRM is integrated deeply into the L2's economic engine:

- unlocks native asset capital productivity
- USF liquidity anchors core pairs on Orvex, the native DEX
- using FIRM earns Karma, the L2 reputation and governance token
- community governance can boost FIRM yields with L2 native yield

5 Audit Scope

The audit scope was limited to changes made to the following files:

```
contracts/script/DeployLiquity2.s.sol
contracts/src/AddressesRegistry.sol
contracts/src/BorrowerOperations.sol
contracts/src/CollateralRegistry.sol
contracts/src/Dependencies/Constants.sol
contracts/src/PriceFeeds/LINEAPriceFeed.sol
contracts/src/PriceFeeds/SGUSDPriceFeed.sol
contracts/src/PriceFeeds/SNTPriceFeed.sol
contracts/src/TroveManager.sol
```

6 Executive Summary

Over the course of 5 days, the Cyfrin team conducted an audit on the Firm Money smart contracts provided by Firm. In this period, a total of 8 issues were found.

The findings consist of 5 Low severity issues with the remainder being informational.

Insufficient Liquidator incentive risk

The collateral gas compensation cap is set to a universal 0.1 ether (0.1e18 token units) for all collateral branches at Liquidators of sGUSD, SNT, and LINEA troves receive effectively zero compensation even when the SP offset path is used, reducing liquidation incentives on these branches regardless of SP state.

During drastic market conditions that drain the SP, liquidators receive zero compensation. The protocol relies solely on trove owners liquidating to prevent bad debt redistribution onto their own positions, an indirect and unreliable incentive during the most critical moments.

Debt Limit

Debt limit is allowed to be bypassed by the accrual of interest and fees, this is intentional in order to always allow accrual to take place for live positions.

When setting Debt Limit, it is only possible to increment the debt limit by 2x the previous or up to the initial debt limit. Under rare circumstances the `updateDebtLimit` function may be DOSed if currentDebtLimit is ever large enough to overflow when it is multiplied by 2.

Centralization Risks

Once the governor is set in `CollateralRegistry` there exists no function to update this value and a compromised or leaked key could be troublesome.

The governor can arbitrarily set the debt limit in order to block any new debt issuance.

There exists “Max 2x per increase” of the debt limit, but this can be repeated over multiple transactions to effectively bypass this restriction.

Summary

Project Name	Firm Money
Repository	firm
Commit	1a4691226fe5...
Fix Commit	48dd5791b22a...
Audit Timeline	Feb 23rd - Feb 27th, 2026
Methods	Manual Review

Issues Found

Critical Risk	0
High Risk	0
Medium Risk	0
Low Risk	5
Informational	3
Gas Optimizations	0
Total Issues	8

Summary of Findings

[L-1] Deployment script is missing new collaterals support	Resolved
[L-2] Debt-limit enforcement is inconsistent with the stated requirement “being at or over the limit means debt can only be repaid or redeemed"	Acknowledged
[L-3] Oracle fallback is inefficient and may choose price that is more stale	Acknowledged
[L-4] Empty SP removes the entire liquidator incentive	Acknowledged
[L-5] Insufficient liquidator incentive on non-ETH branches	Resolved
[I-1] Imported constants remain unused	Resolved
[I-2] `rETH` staleness configuration does not match documentation	Resolved
[I-3] `MetadataNFT` reuses wrong protocol branding	Resolved

7 Findings

7.1 Low Risk

7.1.1 Deployment script is missing new collaterals support

Description: The `DeployLiquity2Script` defines `NUM_BRANCHES = 6` to support ETH, wstETH, rETH, SNT, LINEA, and sGUSD. However, the mainnet deployment path only handles the first three collaterals. Collateral token assignments, price feed deployments, and `TroveManagerParams` for the new branches are incomplete. A mainnet deployment would revert due to uninitialized collateral addresses, a failing `assert` in `_deployPriceFeed`, and the resulting zero-address entries being passed to `CollateralRegistry`.

Impact: Mainnet deployment will fail for branches 3-5 (SNT, LINEA, sGUSD). The `_deployPriceFeed` function hits `assert(_collTokenAddress == RETH_ADDRESS)` and reverts for any collateral beyond rETH. Even if bypassed, `CollateralRegistry` would be initialized with three `address(0)` collateral entries. Additionally, `TroveManagerParams` for the new collaterals are missing the `DEBT_LIMIT` field.

- Collateral assignments missing for indices 3–5:

https://github.com/firm-money/firm/blob/main/contracts/script/DeployLiquity2.s.sol#L654-L664

```
if (block.chainid == 1 && !useTestnetPriceFeeds) {
  vars.collaterals[0] = IERC20Metadata(WETH);
  vars.collaterals[1] = IERC20Metadata(WSTETH_ADDRESS);
  vars.collaterals[2] = IERC20Metadata(RETH_ADDRESS);
  // @audit collaterals[3], [4], [5] never assigned for SNT, LINEA, sGUSD
}
```

- Zero-address collaterals passed to `CollateralRegistry`:

https://github.com/firm-money/firm/blob/main/contracts/script/DeployLiquity2.s.sol#L689

```
r.collateralRegistry = new CollateralRegistry(r.boldToken, vars.collaterals,
vars.troveManagers, deployer);
```

`_deployPriceFeed` only handles three collaterals and reverts otherwise:

https://github.com/firm-money/firm/blob/main/contracts/script/DeployLiquity2.s.sol#L867-L901

```
function _deployPriceFeed(address _collTokenAddress, address _borroweOperationsAddress)
  internal returns (IPriceFeed)
{
  if (block.chainid == 1 && !useTestnetPriceFeeds) {
      if (_collTokenAddress == address(WETH)) {
          return new WETHPriceFeed(...);
      } else if (_collTokenAddress == WSTETH_ADDRESS) {
          return new WSTETHPriceFeed(...);
      }
      // @audit reverts for SNT, LINEA, sGUSD
      assert(_collTokenAddress == RETH_ADDRESS);
      return new RETHPriceFeed(...);
  }
  return new PriceFeedTestnet();
}
```

- `TroveManagerParams` for new collaterals missing `DEBT_LIMIT`:

https://github.com/firm-money/firm/blob/main/contracts/script/DeployLiquity2.s.sol#L394-L421

```
troveManagerParamsArray[3] = TroveManagerParams({
    CCR: CCR_SNT,
    MCR: MCR_SNT,
    SCR: SCR_SNT,
    BCR: BCR_ALL,
    LIQUIDATION_PENALTY_SP: LIQUIDATION_PENALTY_SP_SNT,
```

```
    LIQUIDATION_PENALTY_REDISTRIBUTION: LIQUIDATION_PENALTY_REDISTRIBUTION_SNT
    // @audit missing DEBT_LIMIT (same for indices 4 and 5)
});
```

Recommended Mitigation: Complete the mainnet deployment path: assign the three new collateral token addresses, add corresponding `else if` branches in `_deployPriceFeed` for `SNTPriceFeed`, `LINEAPriceFeed`, and `SGUSDPriceFeed`, and add the `DEBT_LIMIT` field to all three new `TroveManagerParams`.

Firm Money: Fixed in commits 3a2e368, 59e4a8a.

Cyfrin: Verified.

7.1.2 Debt-limit enforcement is inconsistent with the stated requirement “being at or over the limit means debt can only be repaid or redeemed"

Description: Debt-limit enforcement is inconsistent with the stated requirement “being at or over the limit means debt can only be repaid or redeemed.”

`BorrowerOperations` enforces debt limit in:

1. `openTrove` BorrowerOperations.sol:364
2. `adjustTrove` only when `debtIncrease > 0` BorrowerOperations.sol:665

However, debt can still increase through upfront-fee paths where `debtIncrease == 0`:

1. `_applyUpfrontFee` adds fee debt BorrowerOperations.sol:1181
2. Called from `adjustTroveInterestRate` BorrowerOperations.sol:543
3. Called from `setInterestBatchManager` BorrowerOperations.sol:1013
4. Called from `removeFromBatch` BorrowerOperations.sol:1129
5. Similar fee path in `setBatchManagerAnnualInterestRate` BorrowerOperations.sol:948

Impact: When branch debt is at/above `debtLimit`, users can still increase debt through fee-incurring adjustment operations, violating the intended branch cap behavior. This weakens governance risk controls and allows debt drift above limit via non-borrow operations.

Proof of Concept:

1. Set a branch debt limit close to current debt (or equal to current debt).
2. Use an existing active trove.
3. Trigger a premature interest-rate adjustment (within cooldown) via `adjustTroveInterestRate`.
4. `_applyUpfrontFee` computes a positive `upfrontFee` and adds it to trove debt.
5. No `_requireDebtLimitNotExceeded` check is executed on this path, so transaction succeeds even though branch debt increases above the limit.

Same pattern applies to `setInterestBatchManager`, `removeFromBatch`, and `setBatchManagerAnnualInterestRate`.

Recommended Mitigation: Enforce debt limit on **all positive net debt deltas**, not only explicit `debtIncrease`.

Practical fix:

1. Introduce a helper that checks projected branch debt with `delta = debtIncrease + upfrontFee - debtDecrease`.
2. Call it in every path that can add debt (including `_applyUpfrontFee` callers and batch fee-adjustment flows).
3. If behavior is intentionally “borrow-cap only,” update docs/specs to state that explicitly and remove “only repaid or redeemed” wording.

Firm Money: Acknowledged. This is intended behavior, and an understood limitation listed in the hackmd. You dont want to stop interest from accruing, that creates more danger that could lead to bad debt.

7.1.3 Oracle fallback is inefficient and may choose price that is more stale

Description: `MainnetPriceFeedBase` rejects any Chainlink response older than `stalenessThreshold` and immediately falls back to `lastGoodPrice`. However, `lastGoodPrice` has no associated timestamp, so the system cannot compare freshness between:

1. the rejected (slightly stale) oracle response, and
2. the stored `lastGoodPrice`.

As a result, the protocol can replace a newer but stale oracle value with an even older `lastGoodPrice`, worsening effective staleness.

Impact: Pricing during and after shutdown may rely on data older than the most recent oracle observation, increasing price distortion risk. Shutdown pricing may become less representative than necessary, even under available oracle data.

Proof of Concept: Assume:

- `stalenessThreshold = 24 hours`
- Current time = `T`
- Stored `lastGoodPrice = $2,000`, last updated at `T - 48h`
- Chainlink returns `answer = $1,800`, `updatedAt = T - 25h`

Execution:

1. `_isValidChainlinkPrice` checks `block.timestamp - updatedAt < stalenessThreshold` and returns `false` (`25h >= 24h`).
2. Feed is treated as failed and switches to `lastGoodPrice`.
3. Returned price becomes `$2,000` (48h old), instead of `$1,800` (25h old).

Result: fallback increased effective staleness by 23 hours and used a likely less accurate price.

Recommended Mitigation:

1. Store `lastGoodPriceTimestamp` whenever `lastGoodPrice` is updated.
2. On stale oracle path, compare recency of oracle response and lastGoodPriceTimestamp.

Firm Money: Acknowledged. The situation could just as easily be that the oracle has been offline for months, like in the current case of the Mustang liquity fork. In which case we should deal with this by setting the debt limit to 0 or in other ways. The oracle price is a trusted limitation and if the oracle if offline for long periods of time we no longer trust it, so its a good assumption to go with the previous last good price that could be trusted. Great find and nice edge case scenario though!

7.1.4 Empty SP removes the entire liquidator incentive

Description: The protocol sets `ETH_GAS_COMPENSATION = 0` (Constants.sol:14).

The remaining collateral compensation is only calculated inside the SP offset path at TroveManager.sol:378-382. When the SP is nearly empty, most of the liquidation goes through redistribution where `collGasCompensation` stays at its default value of 0.

Impact: During drastic market conditions that drain the SP, liquidators receive zero compensation. The protocol relies solely on trove owners liquidating to prevent bad debt redistribution onto their own positions, an indirect and unreliable incentive during the most critical moments.

Recommended Mitigation: Extend the collateral gas compensation calculation to the redistribution path so liquidators always receive a baseline reward regardless of SP state, similar to LiquityV1 liquidation design.

Firm Money: Acknowledged. Since Status is a gasless chain, theres no cost for us to process the liquidation or redistribution. We'll already be running liquidation bots so I dont foresee this being an issue even in extreme circumstances. The UX benefits for not having a gas deposit are huge.

7.1.5 Insufficient liquidator incentive on non-ETH branches

Description: The collateral gas compensation cap is set to a universal 0.1 ether (0.1e18 token units) for all collateral branches at Constants.sol:71:

```
uint256 constant COLL_GAS_COMPENSATION_CAP = 0.1 ether;
```

This value is applied in TroveManager.sol:344:

```
return LiquityMath._min(_coll / COLL_GAS_COMPENSATION_DIVISOR, COLL_GAS_COMPENSATION_CAP);
```

Since each collateral has a vastly different dollar value, 0.1 token units translates to vastly different compensation:

Collateral	0.1 tokens	USD value
WETH	0.1 ETH	~$200
wstETH	0.1 wstETH	~$250
rETH	0.1 rETH	~$220
sGUSD	0.1 sGUSD	$0.09997
SNT	0.1 SNT	$0.001102
LINEA	0.1 LINEA	$0.0003209

The cap is meaningless for non-ETH branches.

Impact: Liquidators of `sGUSD`, `SNT`, and `LINEA` troves receive effectively zero compensation even when the SP offset path is used, reducing liquidation incentives on these branches regardless of SP state.

Recommended Mitigation: Define a per-branch `COLL_GAS_COMPENSATION_CAP` that reflects each collateral's dollar value, similar to how CCR/MCR/SCR and liquidation penalties are already configured per branch.

Firm Money: Fixed in commit 656ef21.

Cyfrin: Verified. Team has extended the TroveManager to support immutable collateral gas compensation, adjustable per the collateral token.

7.2 Informational

7.2.1 Imported constants remain unused

Description: The deployment script hardcodes `DEBT_LIMIT` values for the WETH and wstETH branches as `100_000_000e18` instead of using the `DEBT_LIMIT_ETH` and `DEBT_LIMIT_WSTETH` constants already imported from `Constants.sol`. The SNT, LINEA, and sGUSD branches omit `DEBT_LIMIT` entirely despite `DEBT_LIMIT_SNT`, `DEBT_LIMIT_LINEA`, and `DEBT_LIMIT_SGUSD` being imported.

Impact: If the debt limit constants in `Constants.sol` are updated, the deployment script will deploy with stale values for WETH/wstETH branches and missing values for the new collateral branches, leading to misconfigured `AddressesRegistry` contracts.

https://github.com/firm-money/firm/blob/main/contracts/script/DeployLiquity2.s.sol#L25-L26

```
import {
  ...
  DEBT_LIMIT_ETH, DEBT_LIMIT_WSTETH, DEBT_LIMIT_RETH,
  DEBT_LIMIT_SNT, DEBT_LIMIT_LINEA, DEBT_LIMIT_SGUSD
} from "src/Dependencies/Constants.sol";`
```

https://github.com/firm-money/firm/blob/main/contracts/script/DeployLiquity2.s.sol#L368-L421

```
// WETH
troveManagerParamsArray[0] = TroveManagerParams({
    ...
    DEBT_LIMIT: 100_000_000e18 // @audit should use DEBT_LIMIT_ETH
});

// wstETH
troveManagerParamsArray[1] = TroveManagerParams({
    ...
    DEBT_LIMIT: 100_000_000e18 // @audit should use DEBT_LIMIT_WSTETH
});

// SNT
troveManagerParamsArray[3] = TroveManagerParams({
    ...
    // @audit missing DEBT_LIMIT: DEBT_LIMIT_SNT
});
```

Recommended Mitigation: Use the imported constants for all branches:

```
- DEBT_LIMIT: 100_000_000e18 // $100M
+ DEBT_LIMIT: DEBT_LIMIT_ETH
```

And add the missing `DEBT_LIMIT` field for SNT, LINEA, and sGUSD using `DEBT_LIMIT_SNT`, `DEBT_LIMIT_LINEA`, and `DEBT_LIMIT_SGUSD` respectively.

Firm Money: Fixed in commit 59e4a8a.

Cyfrin: Verified.

7.2.2 `rETH` staleness configuration does not match documentation

Description: `ORACLE_CONFIG.md` states that all feeds use a 25-hour staleness threshold, but deployment code configures rETH differently.

- Doc claim: ORACLE_CONFIG.md:30
- Actual deployment value: `RETH_ETH_STALENESS_THRESHOLD = 48 hours` at DeployLiquity2.s.sol:119
- That 48h value is passed to `RETHPriceFeed` at DeployLiquity2.s.sol:892

This is a documentation/config inconsistency.

Impact: Operational and audit assumptions can be wrong. Parties relying on docs may expect rETH oracle stale-data handling at 25h, while deployed behavior tolerates up to 48h for the `RETH-ETH` leg. This can affect incident response expectations and risk modeling for oracle freshness.

Recommended Mitigation: Align docs and code to a single source of truth.

Firm Money: Fixed in PR13.

Cyfrin: Verified.

7.2.3 `MetadataNFT` reuses wrong protocol branding

Description: The `MetadataNFT::uri` function contains hardcoded Liquity V2 branding in the NFT metadata. The title reads "Liquity V2 - " and the description references "Liquity V2 is a collateralized debt platform" and "to their own Ethereum address". This branding is inaccurate for the Firm Money deployment on Status Network L2.

https://github.com/firm-money/firm/blob/main/contracts/src/NFTMetadata/MetadataNFT.sol#L36-L49

```
return json.formattedMetadata(
    string.concat("Liquity V2 - ", IERC20Metadata(_troveData._collToken).name()),
    string.concat(
        "Liquity V2 is a collateralized debt platform. Users can lock up ",
        IERC20Metadata(_troveData._collToken).symbol(),
        " to issue stablecoin tokens (BOLD) to their own Ethereum address. ..."
    ),
    renderSVGImage(_troveData),
    attr
);
```

Impact: Trove NFTs will display incorrect protocol branding and network references. This affects user-facing metadata on NFT marketplaces and wallets.

Recommended Mitigation: Update the branding to reflect the Firm Money protocol and Status Network.

Firm Money: Fixed in commit a04706b.

Cyfrin: Verified.