AGVE Protocol

SMART CONTRACT AUDIT

01.07.2021

Made in Germany by Chainsulting.de

Table of contents

1. Disclaimer

The audit makes no statements or warrantees about utility of the code, safety of the code, suitability of the business model, investment advice, endorsement of the platform or its products, 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 purposes only.

The information presented in this report is confidential and privileged. If you are reading this report, you agree to keep it confidential, not to copy, disclose or disseminate without the agreement of agave.finance. If you are not the intended receptor of this document, remember that any disclosure, copying or dissemination of it is forbidden.

Major Versions / Date	Description
0.1 (14.04.2021)	Layout
0.5 (16.04.2021)	Verify Claims and Test Deployment
0.6 (17.04.2021)	Testing SWC Checks
0.8 (19.04.2021)	Automated Security Testing Manual Security Testing
0.9 (20.04.2021)	Summary and Recommendation
1.0 (24.04.2021)	Final document
1.1 (01.07.2021)	Added deployed contract

2. About the Project

Website: https://agave.finance

Twitter: https://twitter.com/Agave_lending

Discord: https://discord.gg/SstXTj6xgp

Telegram: https://t.me/Agave1Hive

Blog: https://agavefinance.medium.com

Reddit: https://www.reddit.com/r/AGVE

2.1 Project Overview

Introducing $AGVE, or Agave, a decentralized non-custodial money market protocol where users can participate by borrowing or lending money through the application. Agave is a fork of Aave, built by the 1Hive community and deployed on the xDai chain where network fees are substantially lower. Additionally, the platform is expected to undergo many changes to build on the system that Aave has built, with the intention of making Agave even more integrated than Aave.

This will be accomplished with further integration of other 1Hive projects such as Celeste, marketplace games, and the Honeyswap exchange. Users can deposit tokens into the smart contract protocol, where they receive a ‘aToken’ back in return. This ‘aToken’ will accrue interest for token holders. Additionally, users can enable their deposit to be used as collateral and thus use this to borrow other assets. This allows users to leverage their deposits.

For example, if a user deposits $HNY into the protocol, then they can use that $HNY as collateral to borrow $XDAI to buy more $HNY. This becomes much more lucrative when compared to mainnet Ethereum because of the low fees on xDai. This allows 1Hive to offer more to its users. Currently, the main platform built from 1Hive is Honeyswap, which is similar to Uniswap but build on the xDai network. With the addition of Agave, users will be able to earn a yield on regular Honeyswap liquidity tokens, thus giving them an additional income stream from the fees generated on Honeyswap by providing liquidity.

3. Vulnerability & Risk Level

Risk represents the probability that a certain source-threat will exploit vulnerability, and the impact of that event on the organization or system. Risk Level is computed based on CVSS version 3.0.

Level	Value	Vulnerability	Risk (Required Action)
Critical	9 – 10	A vulnerability that can disrupt the contract functioning in a number of scenarios, or creates a risk that the contract may be broken.	Immediate action to reduce risk level.
High	7 – 8.9	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.	Implementation of corrective actions as soon as possible.
Medium	4 – 6.9	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.	Implementation of corrective actions in a certain period.
Low	2 – 3.9	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.	Implementation of certain corrective actions or accepting the risk.
Informational	0 – 1.9	A vulnerability that have informational character but is not effecting any of the code.	An observation that does not determine a level of risk

4. Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices. To do so, reviewed line-by-line by our team of expert pentesters and smart contract developers, documenting any issues as there were discovered.

4.1 Methodology

The auditing process follows a routine series of steps:

1. Code review that includes the following:
   i. Review of the specifications, sources, and instructions provided to Chainsulting to make sure we understand the size, scope, and functionality of the smart contract.
   ii. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
   iii. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Chainsulting describe.
2. Testing and automated analysis that includes the following:
   i. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
   ii. Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts.

4.2 Used Code from other Frameworks/Smart Contracts (direct imports)

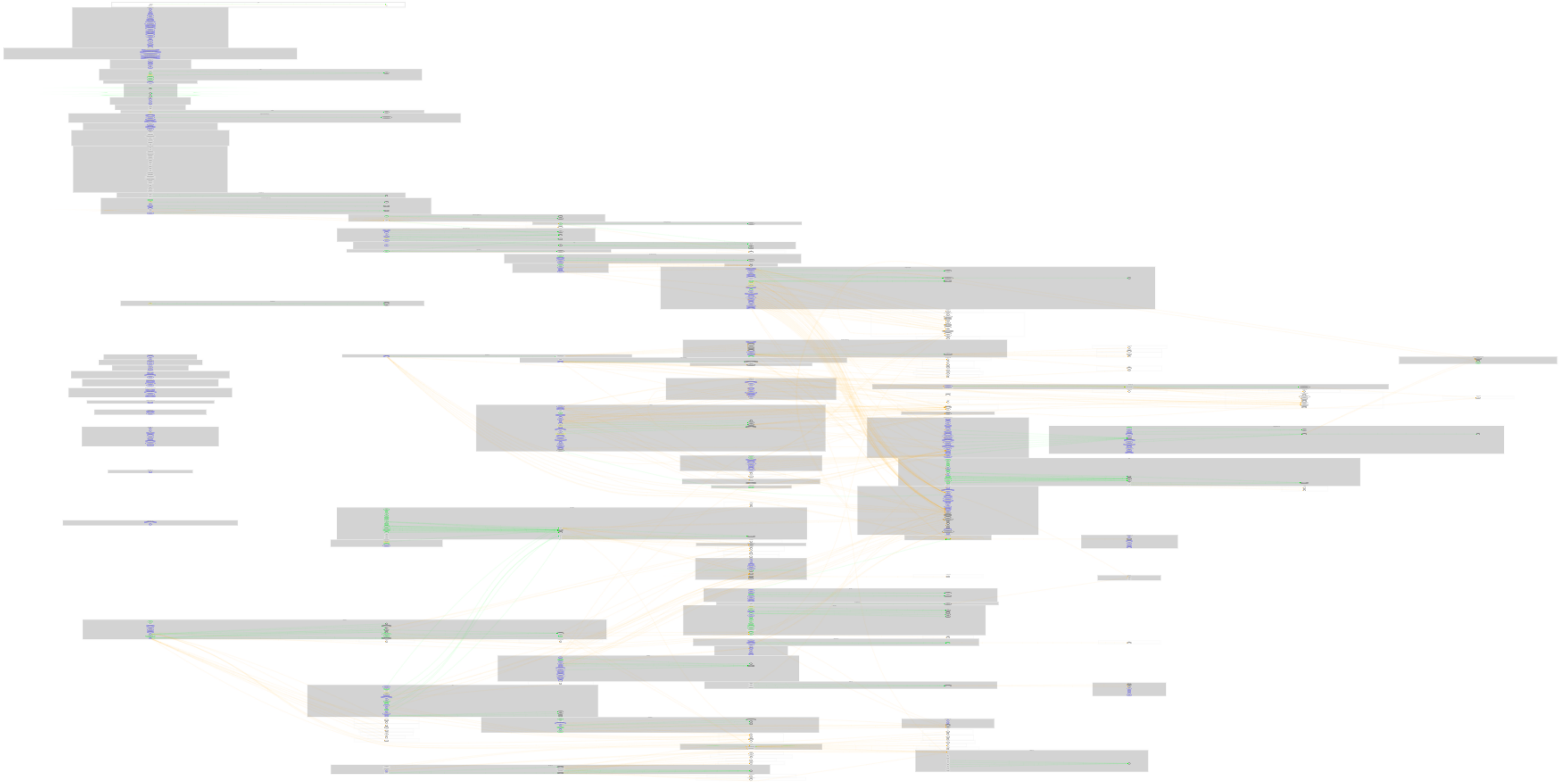
Dependency / Import Path	Source
https://github.com/Agave-DAO/protocol-v2/tree/audited-contracts	**https://github.com/aave/protocol-v2**
@openzeppelin/contracts/access/Ownable.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.1.0/contracts/access/Ownable.sol
@openzeppelin/contracts/math/SafeMath.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.1.0/contracts/math/SafeMath.sol
@openzeppelin/contracts/util/Address.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.1.0/contracts/util/Address.sol
@openzeppelin/contracts/token/ERC20/ERC20.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.1.0/contracts/token/ERC20/ERC20.sol
@openzeppelin/contracts/token/ERC20/IERC20.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.1.0/contracts/token/ERC20/IERC20.sol
@openzeppelin/contracts/token/ERC20/SafeERC20.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.1.0/contracts/token/ERC20/SafeERC20.sol
@openzeppelin/contracts/GSN/Context.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v3.1.0/contracts/GSN/Context.sol

4.3 Tested Contract Files

The following are the MD5 hashes of the reviewed files. A file with a different MD5 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different MD5 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review

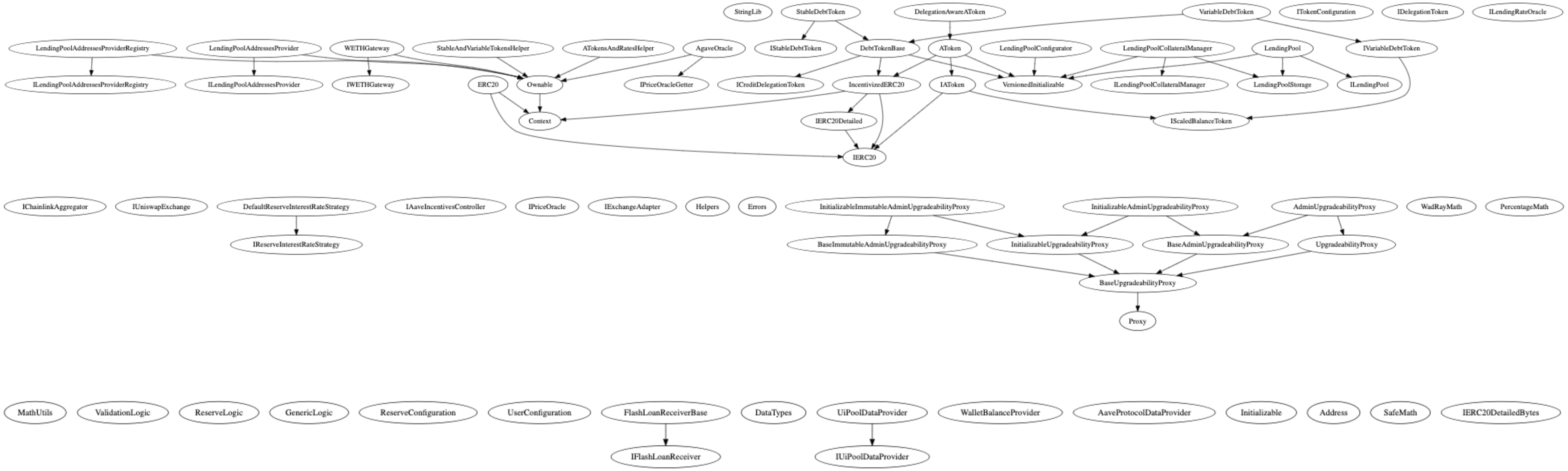
File	Fingerprint (MD5)
../misc/AgaveOracle.sol	3077672a73010d109d606176c8b86e60
../interfaces/IPriceOracleGetter.sol	56181b85c44591f931b25fa0b92ed581
../interfaces/IChainlinkAggregator.sol	0b3a001c61c1a79f42e676506ad32d1a
../deployments/ATokensAndRatesHelper.sol	fb0ff9ccc6ad2a765b905fa5bd229de6

4.4 Metrics / CallGraph

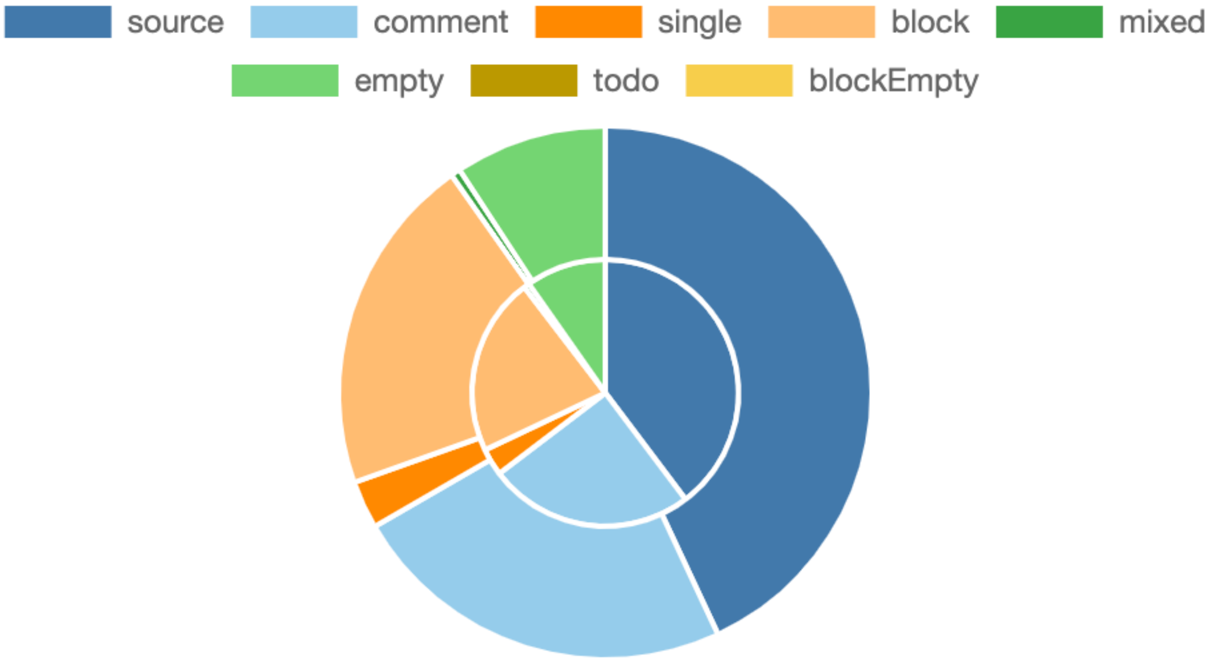


Full Version: https://chainsulting.de/wp-content/uploads/2021/04/agve_solidity-metrics.html

4.5 Inheritance Graph



4.6 Metrics / Source Lines



4.7 Metrics / Capabilities

Solidity Versions observed	🧪 Experimental Features	💰 Can Receive Funds	🖥 Uses Assembly	💣 Has Destroyable Contracts

`0.6.12`
`^0.6.12`
`^0.6.0`
`>=0.4.24 <0.7.0`
`>=0.6.2`
	`ABIEncoderV2`	`yes`	`yes`
			(9 asm blocks)	

📤 Transfers ETH	⚡ Low-Level Calls	👥 DelegateCall	🧮 Uses Hash Functions	🔖 ECRecover	🌀 New/Create/Create2
`yes`		`yes`	`yes`	`yes`	`yes` → `NewContract:AToken` → `NewContract:DefaultReserveInterestRateStrategy` → `NewContract:StableDebtToken` → `NewContract:VariableDebtToken` → `NewContract:InitializableImmutableAdminUpgradeabilityProxy`

🌐Public	💰Payable
344	19

External	Internal	Private	Pure	View
273	365	3	54	177

4.8 Metrics / Source Unites in Scope

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
📚	contracts/deployments/StringLib.sol	1		8	8	6	1	3	
📝	contracts/deployments/ATokensAndRatesHelper.sol	1		122	103	93	1	78	🧪🌀
📝	contracts/deployments/StableAndVariableTokensHelper.sol	1		70	62	53	2	69	🧪🌀
🔍	contracts/interfaces/IPriceOracleGetter.sol		1	16	15	3	10	3	☀️
🔍	contracts/interfaces/ITokenConfiguration.sol		1	14	11	3	7	5	☀️
🔍	contracts/interfaces/IDelegationToken.sol		1	11	10	3	6	3	
🔍	contracts/interfaces/IStableDebtToken.sol		1	125	62	21	70	19	
🔍	contracts/interfaces/ILendingRateOracle.sol		1	19	13	3	11	5	
🔍	contracts/interfaces/ILendingPoolCollateralManager.sol		1	60	53	14	35	3	
🔍	contracts/interfaces/ILendingPoolAddressesProviderRegistry.sol		1	26	16	5	9	9	
🔍	contracts/interfaces/IChainlinkAggregator.sol		1	19	5	3	1	13	

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
🔍	contracts/interfaces/IAToken.sol		1	88	23	6	53	15	
🔍	contracts/interfaces/ILendingPoolAddressesProvider.sol		1	60	23	13	8	39	
🔍	contracts/interfaces/ILendingPool.sol		1	410	181	60	226	47	🧪
🔍	contracts/interfaces/IUniswapExchange.sol		1	21	21	19	1	1	
🔍	contracts/interfaces/IReserveInterestRateStrategy.sol		1	29	10	3	6	7	
🔍	contracts/interfaces/IScaledBalanceToken.sol		1	26	11	3	17	7	
🔍	contracts/interfaces/IVariableDebtToken.sol		1	55	30	5	33	7	
🔍	contracts/interfaces/IAaveIncentivesController.sol		1	11	6	4	1	3	🧪
🔍	contracts/interfaces/IPriceOracle.sol		1	17	11	3	10	5	
🔍	contracts/interfaces/ICreditDelegationToken.sol		1	28	19	9	14	5	
🔍	contracts/interfaces/IExchangeAdapter.sol		1	23	15	11	1	5	
📝	contracts/protocol/lendingpool/DefaultReserveInterestRateStrategy.sol	1		213	192	116	42	66	

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
📝	contracts/protocol/lendingpool/LendingPoolConfigurator.sol	1		562	541	262	177	190	🧪🌀
📝	contracts/protocol/lendingpool/LendingPoolStorage.sol	1		26	26	17	2	7	
📝	contracts/protocol/lendingpool/LendingPoolCollateralManager.sol	1		317	304	217	53	75	
📝	contracts/protocol/lendingpool/LendingPool.sol	1		923	826	502	206	287	🧪👥
🎨	contracts/protocol/tokenization/base/DebtTokenBase.sol	1		167	134	80	36	50	
📝	contracts/protocol/tokenization/StableDebtToken.sol	1		358	327	184	95	116	
📝	contracts/protocol/tokenization/VariableDebtToken.sol	1		142	128	58	50	51	
📝	contracts/protocol/tokenization/IncentivizedERC20.sol	1		253	227	133	57	97	
📝	contracts/protocol/tokenization/AToken.sol	1		342	293	150	103	132	🖥🧮🔖
📝	contracts/protocol/tokenization/DelegationAwareAToken.sol	1		49	49	35	10	16	

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
📚	contracts/protocol/libraries/helpers/Helpers.sol	1		39	31	17	11	9	
📚	contracts/protocol/libraries/helpers/Errors.sol	1		119	119	95	78	81	
🎨	contracts/protocol/libraries/aave-upgradeability/VersionedInitializable.sol	1		77	72	29	38	15	🖥
📝	contracts/protocol/libraries/aave-upgradeability/BaseImmutableAdminUpgradeabilityProxy.sol	1		80	76	33	34	30	💰👥
📝	contracts/protocol/libraries/aave-upgradeability/InitializableImmutableAdminUpgradeabilityProxy.sol	1		23	23	12	8	9	
📚	contracts/protocol/libraries/math/WadRayMath.sol	1		135	135	58	52	20	
📚	contracts/protocol/libraries/math/PercentageMath.sol	1		54	54	25	21	9	
📚	contracts/protocol/libraries/math/MathUtils.sol	1		84	72	29	28	25	
📚	contracts/protocol/libraries/logic/ValidationLogic.sol	1		476	412	244	116	112	🧪
📚	contracts/protocol/libraries/logic/ReserveLogic.sol	1		373	336	204	84	80	

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
📚	contracts/protocol/libraries/logic/GenericLogic.sol	1		278	244	161	45	68	🧪
📚	contracts/protocol/libraries/configuration/ReserveConfiguration.sol	1		345	286	136	126	36	
📚	contracts/protocol/libraries/configuration/UserConfiguration.sol	1		112	92	37	46	11	
🔍	contracts/flashloan/interfaces/IFlashLoanReceiver.sol		1	25	14	5	7	7	
📚	contracts/protocol/libraries/types/DataTypes.sol	1		49	49	24	21	1	
📝	contracts/protocol/configuration/LendingPoolAddressesProviderRegistry.sol	1		89	84	44	26	46	
🎨	contracts/flashloan/base/FlashLoanReceiverBase.sol	1		22	22	17	1	9	
📝	contracts/protocol/configuration/LendingPoolAddressesProvider.sol	1		215	211	102	78	109	🌀
📝	contracts/misc/UiPoolDataProvider.sol	1		160	142	123	5	76	🧪
📝	contracts/misc/WalletBalanceProvider.sol	1		110	102	57	29	84	🧪💰

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
📝	contracts/misc/WETHGateway.sol	1		182	170	86	64	104	🧪💰📤
📝	contracts/misc/AgaveOracle.sol	1		139	136	81	38	80	
📝	contracts/misc/AaveProtocolDataProvider.sol	1		180	128	108	1	107	🧪
📝	contracts/dependencies/openzeppelin/upgradeability/UpgradeabilityProxy.sol	1		28	28	12	14	16	💰👥🧮
📝	contracts/dependencies/openzeppelin/upgradeability/InitializableAdminUpgradeabilityProxy.sol	1		42	38	17	18	20	💰🧮
🎨	contracts/dependencies/openzeppelin/upgradeability/Proxy.sol	1		72	64	25	38	42	🖥💰👥
📝	contracts/dependencies/openzeppelin/upgradeability/InitializableUpgradeabilityProxy.sol	1		29	29	13	14	19	💰👥🧮
📝	contracts/dependencies/openzeppelin/upgradeability/BaseAdminUpgradeabilityProxy.sol	1		125	121	48	61	53	🖥💰👥
📝	contracts/dependencies/openzeppelin/upgradeability/BaseUpgradeabilityProxy.sol	1		65	65	27	30	22	🖥
📝	contracts/dependencies/openzeppelin/upgradeability/AdminUpgradeabilityProxy.sol	1		36	36	15	18	17	💰🧮

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
📝	contracts/dependencies/openzeppelin/upgradeability/Initializable.sol	1		66	66	28	30	14	🖥
📚	contracts/dependencies/openzeppelin/contracts/Address.sol	1		61	61	16	42	11	🖥
🎨	contracts/dependencies/openzeppelin/contracts/Context.sol	1		23	23	10	12	1	🔆
🔍	contracts/dependencies/openzeppelin/contracts/IERC20Detailed.sol		1	12	7	4	1	9	
📚	contracts/dependencies/openzeppelin/contracts/SafeMath.sol	1		163	151	39	99	10	
📝	contracts/misc/interfaces/IERC20DetailedBytes.sol	1		8	8	6	1	4	
🔍	contracts/misc/interfaces/IWETH.sol		1	16	5	3	1	12	💰
📚	contracts/dependencies/openzeppelin/contracts/SafeERC20.sol	1		64	51	29	13	19	
🔍	contracts/misc/interfaces/IUiPoolDataProvider.sol		1	93	70	49	30	3	🧪
📝	contracts/dependencies/openzeppelin/contracts/Ownable.sol	1		69	69	27	33	23	
🔍	contracts/misc/interfaces/IWETHGateway.sol		1	20	5	3	1	15	💰

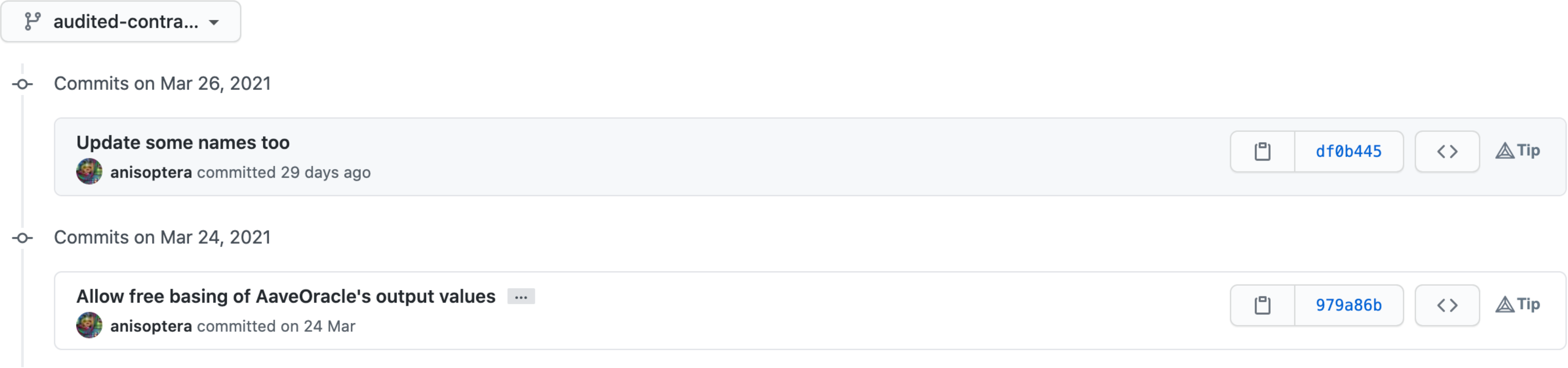
Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
🔍	contracts/misc/interfaces/IUniswapV2Router02.sol		1	51	7	4	1	16	💰
📝	contracts/dependencies/openzeppelin/contracts/ERC20.sol	1		344	318	102	184	81	
🔍	contracts/misc/interfaces/IUniswapV2Router01.sol		1	161	5	3	1	48	💰
🔍	contracts/dependencies/openzeppelin/contracts/IERC20.sol		1	80	25	17	57	13	🔆
📝📚 🔍🎨	**Totals**	**51**	**27**	**9604**	**8017**	**4321**	**3010**	**3034**	🖥🧪💰📤 👥🧮🏷🌀 🔆

Legend: [➖]

- **Lines**: total lines of the source unit
- **nLines**: normalized lines of the source unit (e.g. normalizes functions spanning multiple lines)
- **nSLOC**: normalized source lines of code (only source-code lines; no comments, no blank lines)
- **Comment Lines**: lines containing single or block comments
- **Complexity Score**: a custom complexity score derived from code statements that are known to introduce code complexity (branches, loops, calls, external interfaces, ...)

4.9 Fork details

Changes can be viewed here: https://github.com/Agave-DAO/protocol-v2/commits/audited-contracts



Commits on Mar 26, 2021

Update some names too

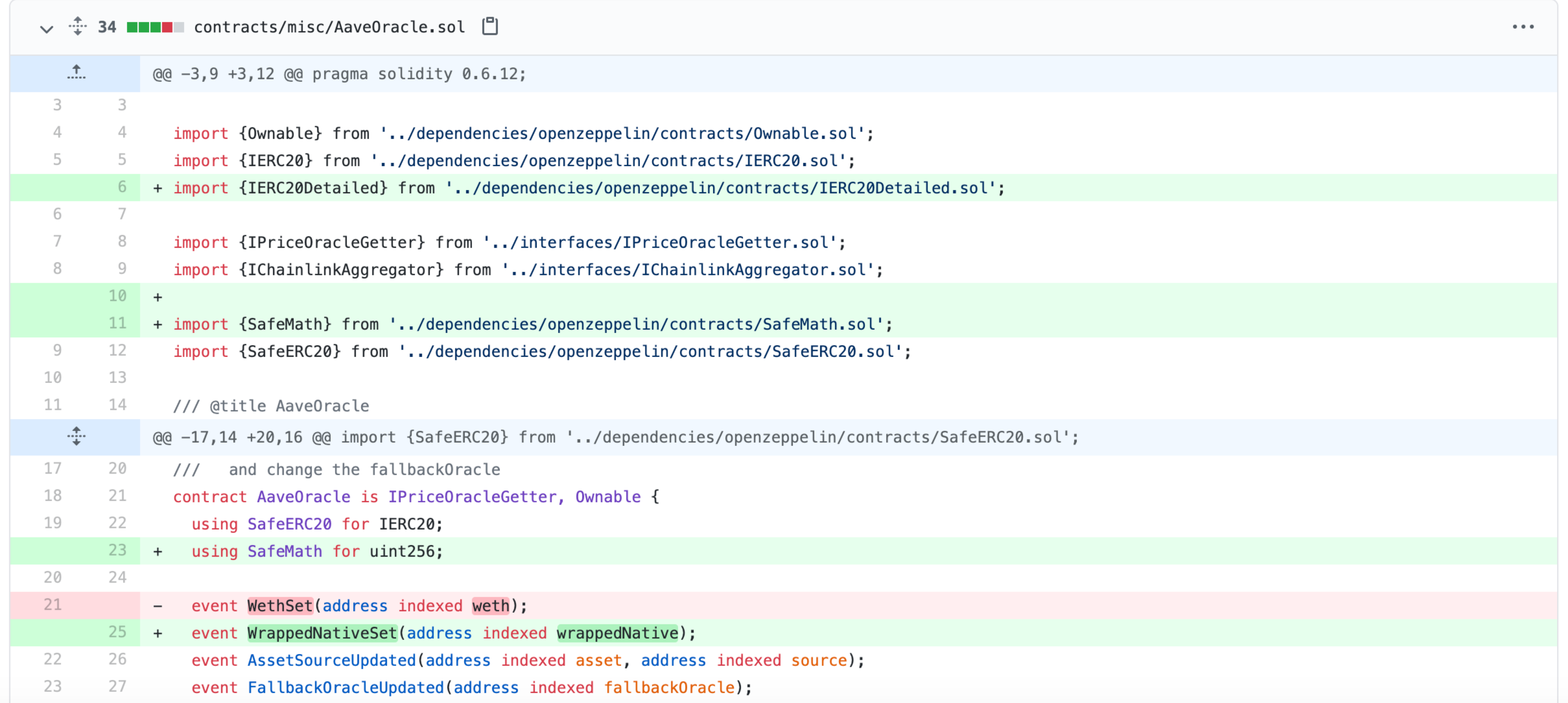
anisoptera committed 29 days ago

Commits on Mar 24, 2021

Allow free basing of AaveOracle's output values ...

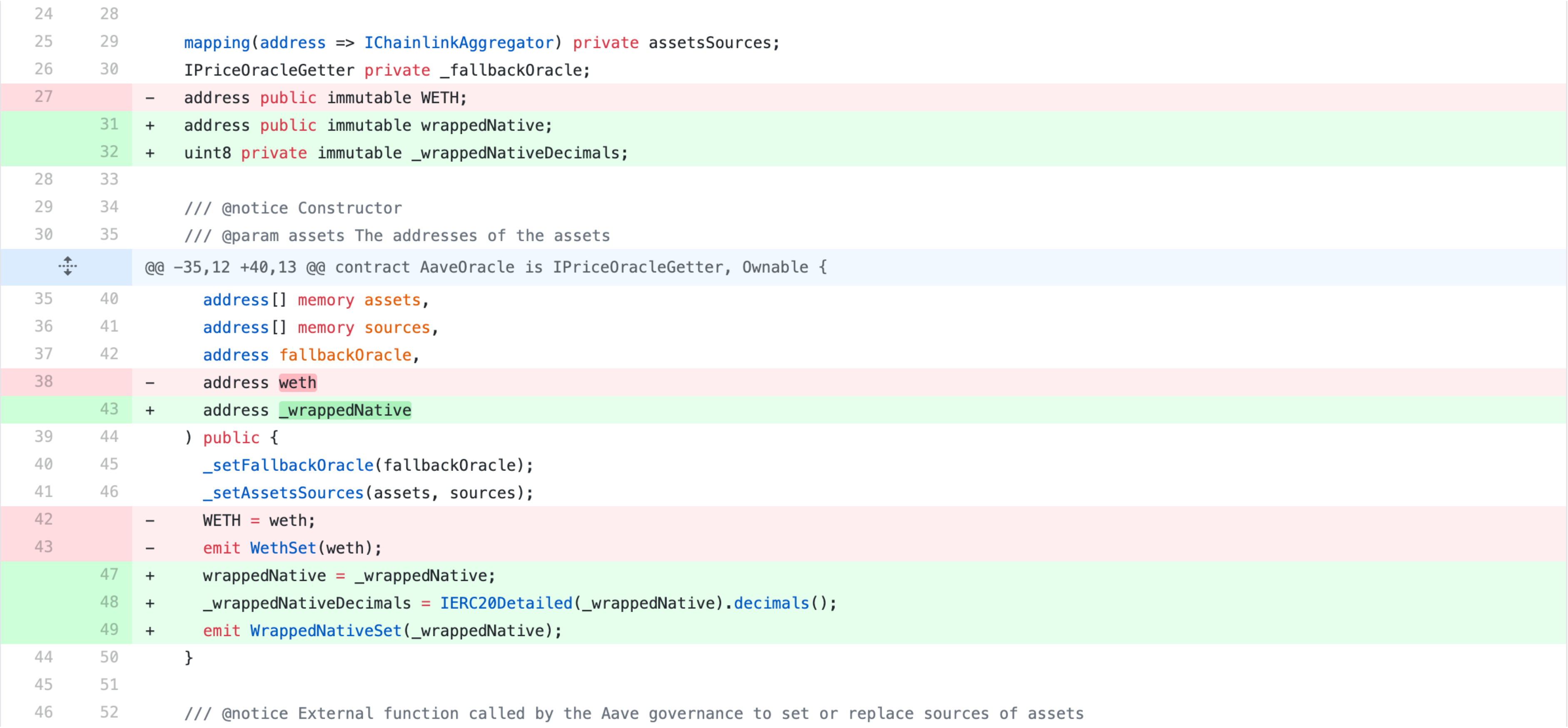
anisoptera committed on 24 Mar

Commit 979a86b:
https://github.com/Agave-DAO/protocol-v2/commit/979a86b66b03bdca3f3d6d7f3f1c40fc34a90c1a

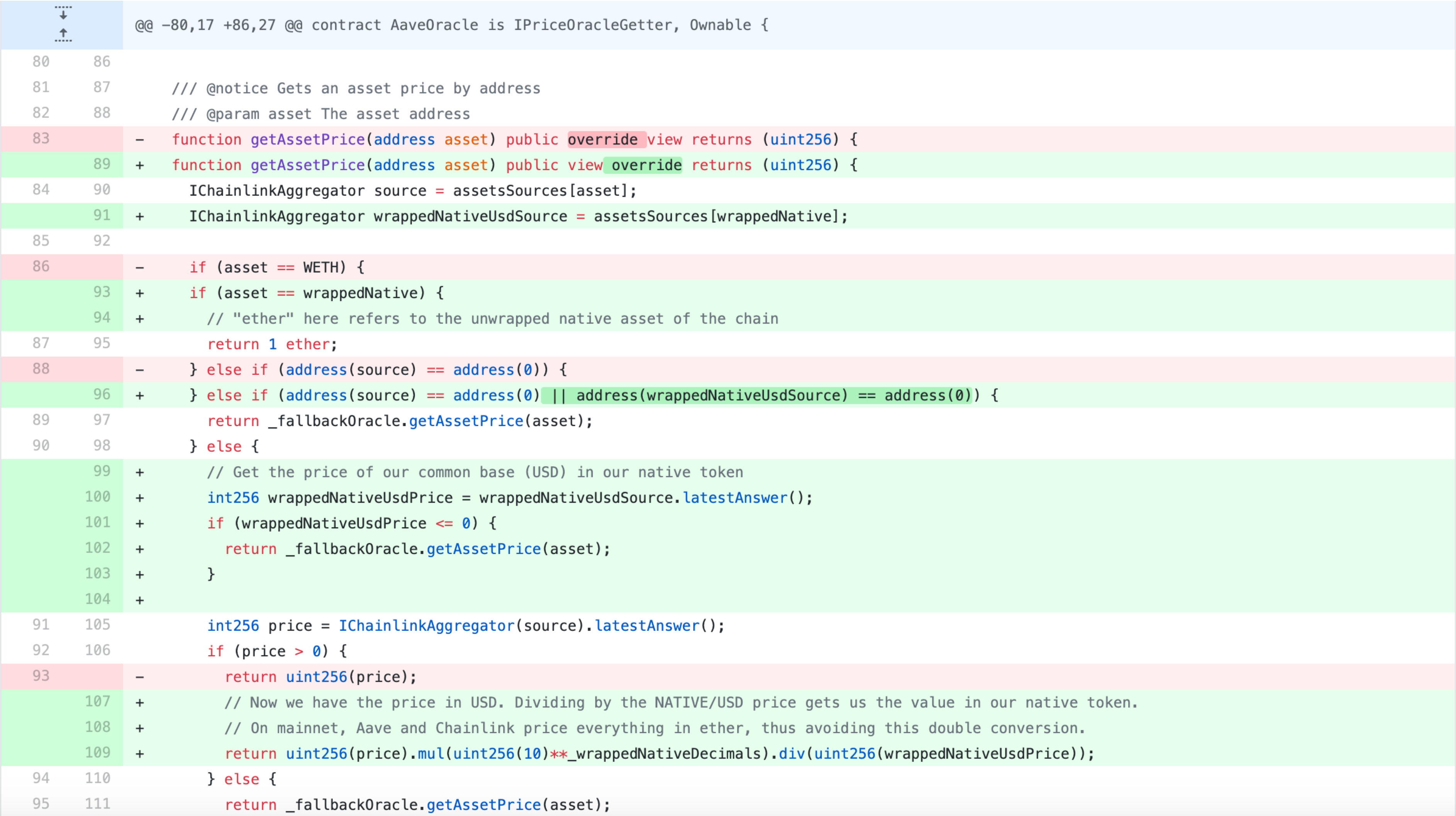


34 🟩🟩🟩🟥⬜ contracts/misc/AaveOracle.sol

```
@@ -3,9 +3,12 @@ pragma solidity 0.6.12;

  import {Ownable} from '../dependencies/openzeppelin/contracts/Ownable.sol';
  import {IERC20} from '../dependencies/openzeppelin/contracts/IERC20.sol';
+ import {IERC20Detailed} from '../dependencies/openzeppelin/contracts/IERC20Detailed.sol';

  import {IPriceOracleGetter} from '../interfaces/IPriceOracleGetter.sol';
  import {IChainlinkAggregator} from '../interfaces/IChainlinkAggregator.sol';
+
+ import {SafeMath} from '../dependencies/openzeppelin/contracts/SafeMath.sol';
  import {SafeERC20} from '../dependencies/openzeppelin/contracts/SafeERC20.sol';

  /// @title AaveOracle
@@ -17,14 +20,16 @@ import {SafeERC20} from '../dependencies/openzeppelin/contracts/SafeERC20.sol';
  ///   and change the fallbackOracle
  contract AaveOracle is IPriceOracleGetter, Ownable {
    using SafeERC20 for IERC20;
+   using SafeMath for uint256;

-   event WethSet(address indexed weth);
+   event WrappedNativeSet(address indexed wrappedNative);
    event AssetSourceUpdated(address indexed asset, address indexed source);
    event FallbackOracleUpdated(address indexed fallbackOracle);
```

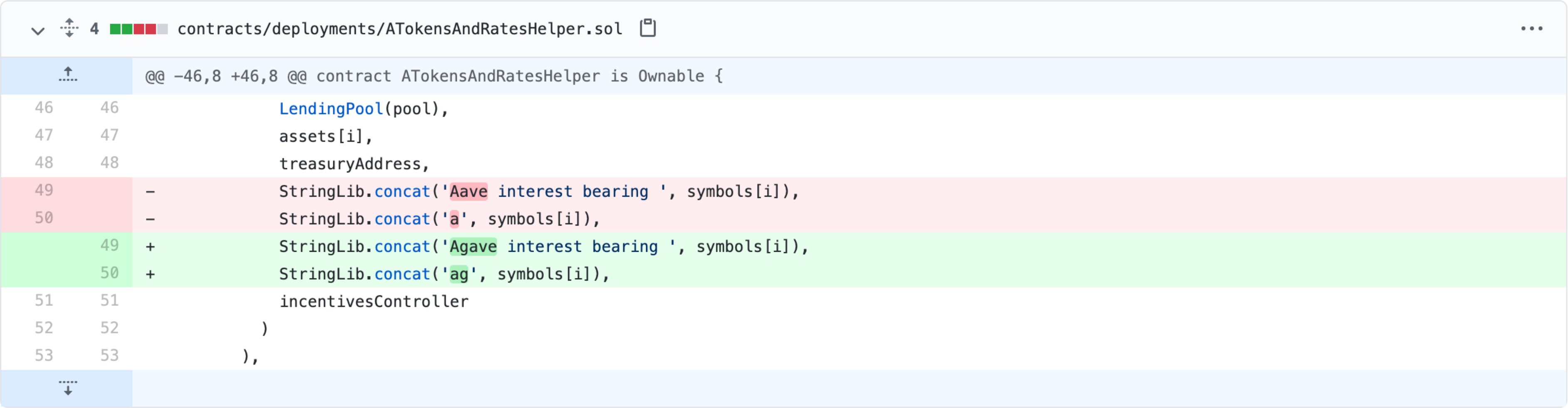


```diff
 
   mapping(address => IChainlinkAggregator) private assetsSources;
   IPriceOracleGetter private _fallbackOracle;
-  address public immutable WETH;
+  address public immutable wrappedNative;
+  uint8 private immutable _wrappedNativeDecimals;
 
   /// @notice Constructor
   /// @param assets The addresses of the assets
@@ -35,12 +40,13 @@ contract AaveOracle is IPriceOracleGetter, Ownable {
     address[] memory assets,
     address[] memory sources,
     address fallbackOracle,
-    address weth
+    address _wrappedNative
   ) public {
     _setFallbackOracle(fallbackOracle);
     _setAssetsSources(assets, sources);
-    WETH = weth;
-    emit WethSet(weth);
+    wrappedNative = _wrappedNative;
+    _wrappedNativeDecimals = IERC20Detailed(_wrappedNative).decimals();
+    emit WrappedNativeSet(_wrappedNative);
   }
 
   /// @notice External function called by the Aave governance to set or replace sources of assets
```

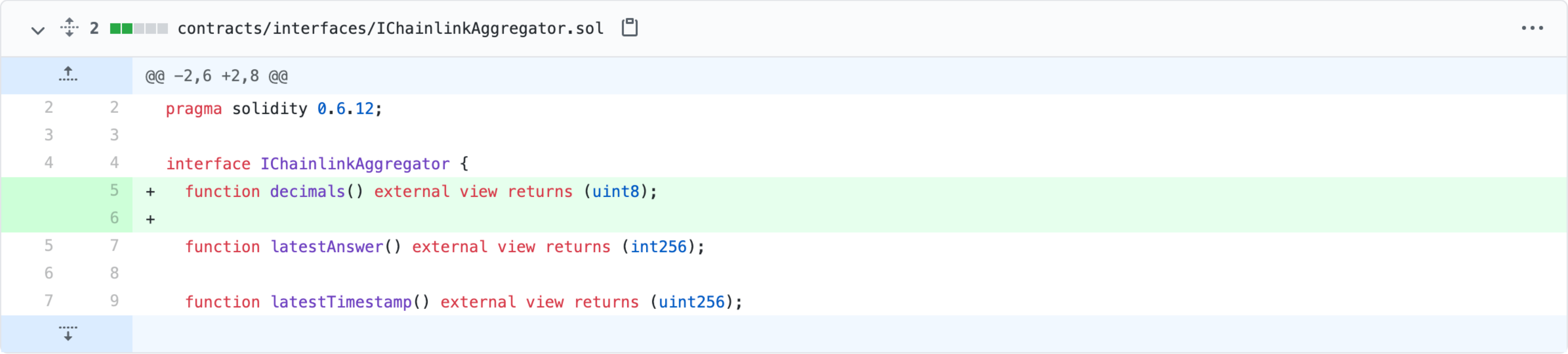


```
@@ -80,17 +86,27 @@ contract AaveOracle is IPriceOracleGetter, Ownable {

    /// @notice Gets an asset price by address
    /// @param asset The asset address
-   function getAssetPrice(address asset) public override view returns (uint256) {
+   function getAssetPrice(address asset) public view override returns (uint256) {
      IChainlinkAggregator source = assetsSources[asset];
+     IChainlinkAggregator wrappedNativeUsdSource = assetsSources[wrappedNative];

-     if (asset == WETH) {
+     if (asset == wrappedNative) {
+       // "ether" here refers to the unwrapped native asset of the chain
        return 1 ether;
-     } else if (address(source) == address(0)) {
+     } else if (address(source) == address(0) || address(wrappedNativeUsdSource) == address(0)) {
        return _fallbackOracle.getAssetPrice(asset);
      } else {
+       // Get the price of our common base (USD) in our native token
+       int256 wrappedNativeUsdPrice = wrappedNativeUsdSource.latestAnswer();
+       if (wrappedNativeUsdPrice <= 0) {
+         return _fallbackOracle.getAssetPrice(asset);
+       }
+
        int256 price = IChainlinkAggregator(source).latestAnswer();
        if (price > 0) {
-         return uint256(price);
+         // Now we have the price in USD. Dividing by the NATIVE/USD price gets us the value in our native token.
+         // On mainnet, Aave and Chainlink price everything in ether, thus avoiding this double conversion.
+         return uint256(price).mul(uint256(10)**_wrappedNativeDecimals).div(uint256(wrappedNativeUsdPrice));
        } else {
          return _fallbackOracle.getAssetPrice(asset);
```

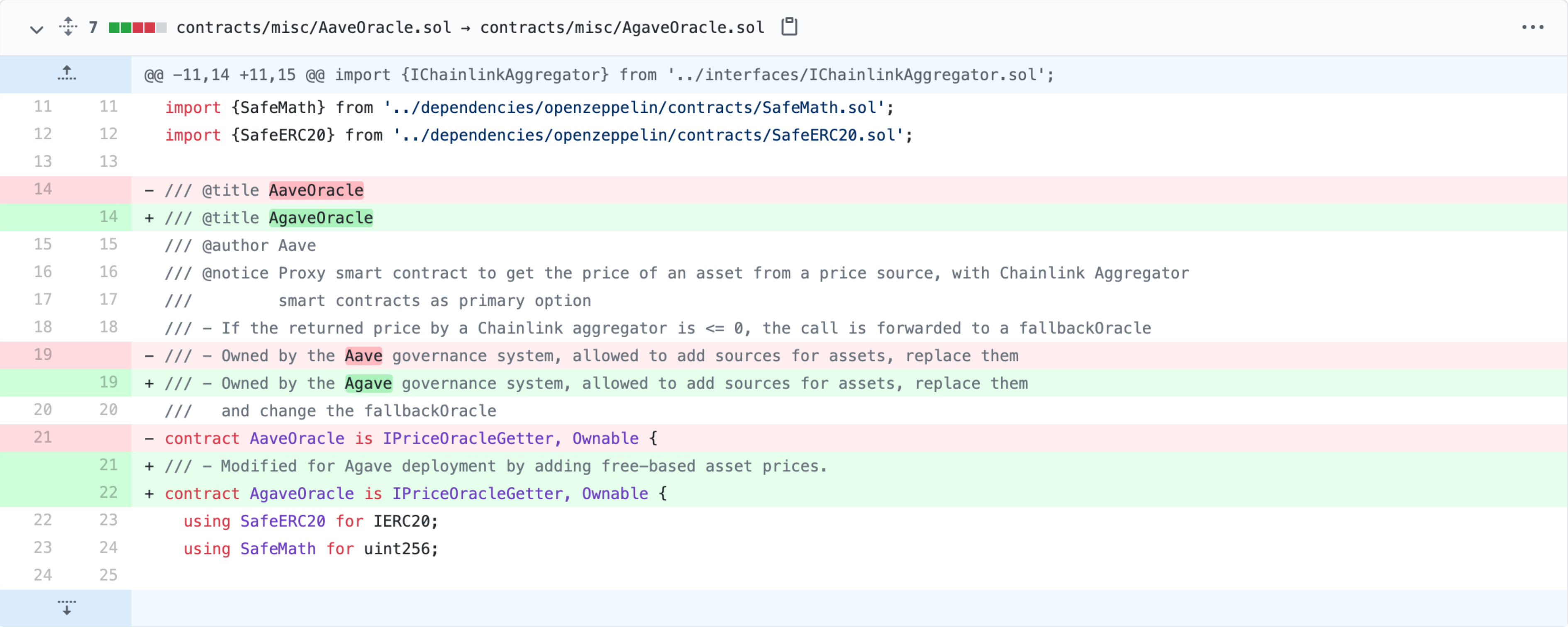
Commit df0b445:
https://github.com/Agave-DAO/protocol-v2/commit/df0b445e731f68b5bea171b168e85d2ea08ae0d5



```
@@ -46,8 +46,8 @@ contract ATokensAndRatesHelper is Ownable {
                  LendingPool(pool),
                  assets[i],
                  treasuryAddress,
-                 StringLib.concat('Aave interest bearing ', symbols[i]),
-                 StringLib.concat('a', symbols[i]),
+                 StringLib.concat('Agave interest bearing ', symbols[i]),
+                 StringLib.concat('ag', symbols[i]),
                  incentivesController
                )
              ),
```

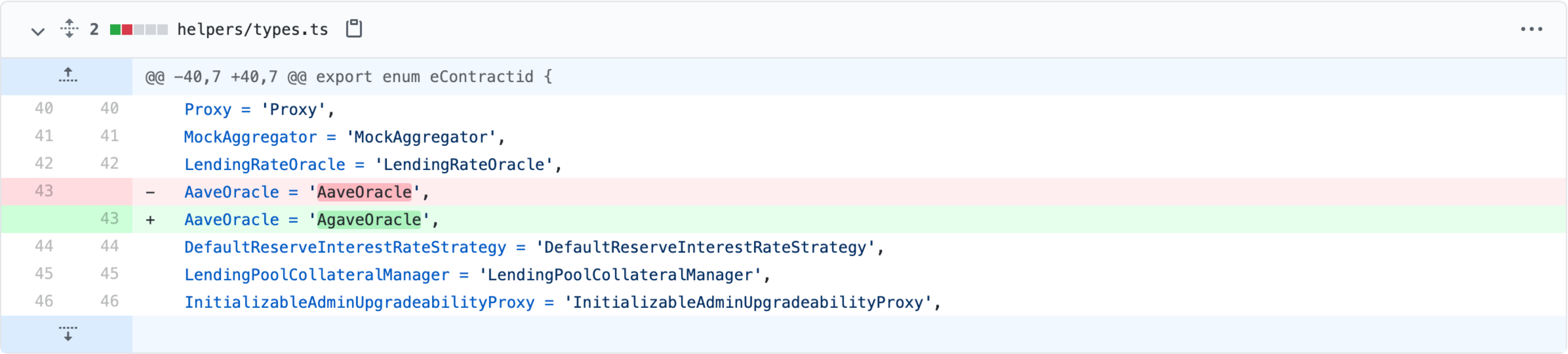


```
@@ -2,6 +2,8 @@
  pragma solidity 0.6.12;

  interface IChainlinkAggregator {
+   function decimals() external view returns (uint8);
+
    function latestAnswer() external view returns (int256);

    function latestTimestamp() external view returns (uint256);
```



7 contracts/misc/AaveOracle.sol → contracts/misc/AgaveOracle.sol

```
@@ -11,14 +11,15 @@ import {IChainlinkAggregator} from '../interfaces/IChainlinkAggregator.sol';
  import {SafeMath} from '../dependencies/openzeppelin/contracts/SafeMath.sol';
  import {SafeERC20} from '../dependencies/openzeppelin/contracts/SafeERC20.sol';

- /// @title AaveOracle
+ /// @title AgaveOracle
  /// @author Aave
  /// @notice Proxy smart contract to get the price of an asset from a price source, with Chainlink Aggregator
  ///         smart contracts as primary option
  /// - If the returned price by a Chainlink aggregator is <= 0, the call is forwarded to a fallbackOracle
- /// - Owned by the Aave governance system, allowed to add sources for assets, replace them
+ /// - Owned by the Agave governance system, allowed to add sources for assets, replace them
  ///   and change the fallbackOracle
- contract AaveOracle is IPriceOracleGetter, Ownable {
+ /// - Modified for Agave deployment by adding free-based asset prices.
+ contract AgaveOracle is IPriceOracleGetter, Ownable {
    using SafeERC20 for IERC20;
    using SafeMath for uint256;

```



```
2 helpers/types.ts
@@ -40,7 +40,7 @@ export enum eContractid {
    Proxy = 'Proxy',
    MockAggregator = 'MockAggregator',
    LendingRateOracle = 'LendingRateOracle',
-   AaveOracle = 'AaveOracle',
+   AaveOracle = 'AgaveOracle',
    DefaultReserveInterestRateStrategy = 'DefaultReserveInterestRateStrategy',
    LendingPoolCollateralManager = 'LendingPoolCollateralManager',
    InitializableAdminUpgradeabilityProxy = 'InitializableAdminUpgradeabilityProxy',
```

5. Scope of Work

The Agave Team provided us with the files that needs to be tested. The scope of the audit are the Agave Protocol contracts.

Following contracts with the direct imports has been tested:

- AgaveOracle.sol
- IPriceOracleGetter.sol
- IChainlinkAggregator.sol
- ATokensAndRatesHelper.sol

The team put forward the following assumptions regarding the security, usage of the contracts:

- The changes that have been made to the forked AAVE Protocol are not affecting the overall security

The main goal of this audit was to verify these claims. The auditors can provide additional feedback on the code upon the client's request.

5.1 Manual and Automated Vulnerability Test

CRITICAL ISSUES

During the audit, Chainsulting‘s experts found **no Critical issues** in the code of the smart contract.

HIGH ISSUES

During the audit, Chainsulting’s experts found **no High issues** in the code of the smart contract.

MEDIUM ISSUES

During the audit, Chainsulting’s experts found **no Medium issues** in the code of the smart contract.

LOW ISSUES

During the audit, Chainsulting’s experts found **no Low issues** in the code of the smart contract.

5.2. SWC Attacks

ID	Title	Relationships	Test Result
SWC-131	Presence of unused variables	CWE-1164: Irrelevant Code	✅
SWC-130	Right-To-Left-Override control character (U+202E)	CWE-451: User Interface (UI) Misrepresentation of Critical Information	✅
SWC-129	Typographical Error	CWE-480: Use of Incorrect Operator	✅
SWC-128	DoS With Block Gas Limit	CWE-400: Uncontrolled Resource Consumption	✅
SWC-127	Arbitrary Jump with Function Type Variable	CWE-695: Use of Low-Level Functionality	✅
SWC-125	Incorrect Inheritance Order	CWE-696: Incorrect Behavior Order	✅
SWC-124	Write to Arbitrary Storage Location	CWE-123: Write-what-where Condition	✅
SWC-123	Requirement Violation	CWE-573: Improper Following of Specification by Caller	✅

ID	Title	Relationships	Test Result
SWC-122	Lack of Proper Signature Verification	CWE-345: Insufficient Verification of Data Authenticity	✅
SWC-121	Missing Protection against Signature Replay Attacks	CWE-347: Improper Verification of Cryptographic Signature	✅
SWC-120	Weak Sources of Randomness from Chain Attributes	CWE-330: Use of Insufficiently Random Values	✅
SWC-119	Shadowing State Variables	CWE-710: Improper Adherence to Coding Standards	✅
SWC-118	Incorrect Constructor Name	CWE-665: Improper Initialization	✅
SWC-117	Signature Malleability	CWE-347: Improper Verification of Cryptographic Signature	✅
SWC-116	Timestamp Dependence	CWE-829: Inclusion of Functionality from Untrusted Control Sphere	✅
SWC-115	Authorization through tx.origin	CWE-477: Use of Obsolete Function	✅
SWC-114	Transaction Order Dependence	CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')	✅

ID	Title	Relationships	Test Result
SWC-113	DoS with Failed Call	CWE-703: Improper Check or Handling of Exceptional Conditions	✅
SWC-112	Delegatecall to Untrusted Callee	CWE-829: Inclusion of Functionality from Untrusted Control Sphere	✅
SWC-111	Use of Deprecated Solidity Functions	CWE-477: Use of Obsolete Function	✅
SWC-110	Assert Violation	CWE-670: Always-Incorrect Control Flow Implementation	✅
SWC-109	Uninitialized Storage Pointer	CWE-824: Access of Uninitialized Pointer	✅
SWC-108	State Variable Default Visibility	CWE-710: Improper Adherence to Coding Standards	✅
SWC-107	Reentrancy	CWE-841: Improper Enforcement of Behavioral Workflow	✅
SWC-106	Unprotected SELFDESTRUCT Instruction	CWE-284: Improper Access Control	✅
SWC-105	Unprotected Ether Withdrawal	CWE-284: Improper Access Control	✅
SWC-104	Unchecked Call Return Value	CWE-252: Unchecked Return Value	✅

ID	Title	Relationships	Test Result
SWC-103	Floating Pragma	CWE-664: Improper Control of a Resource Through its Lifetime	✅
SWC-102	Outdated Compiler Version	CWE-937: Using Components with Known Vulnerabilities	✅
SWC-101	Integer Overflow and Underflow	CWE-682: Incorrect Calculation	✅
SWC-100	Function Default Visibility	CWE-710: Improper Adherence to Coding Standards	✅

5.3. Associated audits with the forked codebase

PeckShield	https://github.com/aave/protocol-v2/blob/feat/light-deployments/audits/Peckshield-aave-v2-light-16-03-2021.pdf	Mar. 2021
SigmaPrime	https://github.com/aave/protocol-v2/blob/master/audits/SigmaPrime-aave-v2-01-2021.pdf	Jan. 2021
Consensys Diligence	https://consensys.net/diligence/audits/2020/09/aave-protocol-v2/	Sep. 2020
Certik	https://github.com/aave/protocol-v2/blob/master/audits/Certik-aave-v2-03-12-2020.pdf	Sep. 2020
PeckShield	https://github.com/aave/protocol-v2/blob/master/audits/Peckshield-aave-v2-03-12-2020-EN.pdf	Sep. 2020
MixBytes	https://github.com/aave/protocol-v2/blob/master/audits/Mixbytes-aave-v2-03-12-2020.pdf	Sep. 2020

6. Executive Summary

Two (2) independent Chainsulting experts performed an unbiased and isolated audit of the smart contract codebase. The focus on our audit have been the changes, that has been made to the forked codebase of aave protocol v2.

The main goal of the audit was to verify the claims regarding the security of the smart contract and the functions. During the audit, no critical issues were found, after the manual and automated security testing.

7. Deployed Smart Contract

VERIFIED BY BYTECODE OR CODEBASE

Smart Contracts are deployed here:

LendingPoolAddressesProviderRegistry
https://blockscout.com/xdai/mainnet/address/0xa5E80AEAa020Ae41b1cBEe75dE7826297F7D803E/contracts
LendingPoolAddressesProvider
https://blockscout.com/xdai/mainnet/address/0xA91B9095eFa6C0568467562032202108e49c9Ef8/contracts
ReserveLogic
https://blockscout.com/xdai/mainnet/address/0x73917b79297e64fEf65eE9BcfC42EB7F350795DA/contracts
GenericLogic
https://blockscout.com/xdai/mainnet/address/0xF25D23913C6c8De1c3C25539fDabEe1e51a06C63/contracts
ValidationLogic
https://blockscout.com/xdai/mainnet/address/0x10EA06D74F8d7681F584558d9200cc7A9b852a9F/contracts
LendingPoolImpl
https://blockscout.com/xdai/mainnet/address/0x2d5bC0F897ad211ef9f954EADF59c43FB1EC4788/contracts

LendingPool

https://blockscout.com/xdai/mainnet/address/0x207E9def17B4bd1045F5Af2C651c081F9FDb0842/contracts

LendingPoolConfiguratorImpl

https://blockscout.com/xdai/mainnet/address/0xb74B4bD769c2F909fECD9Ef7A329d6E7EF896e13/contracts

LendingPoolConfigurator

https://blockscout.com/xdai/mainnet/address/0x4078Be5aBe5AD1FB2A3eD9b933798972Fa853e4A/contracts

StableAndVariableTokensHelper

https://blockscout.com/xdai/mainnet/address/0x77dD00583906A70a143b75d36dF6F763b04f85ad/contracts

ATokensAndRatesHelper

https://blockscout.com/xdai/mainnet/address/0x37FE1Fe2287d45d71A049693B71Ea88684E8B89d/contracts

LendingRateOracle

https://blockscout.com/xdai/mainnet/address/0x2B73F555A39c69D4a3947ae3B01470E1c1754B8e/contracts

AaveProtocolDataProvider

https://blockscout.com/xdai/mainnet/address/0xa874f66342a04c24b213BF0715dFf18818D24014/contracts

stableDebtUSDC

https://blockscout.com/xdai/mainnet/address/0x07417aa181b8E1Be69147F920BBDd7f6210F3cfc/contracts

variableDebtUSDC

https://blockscout.com/xdai/mainnet/address/0x3907712A69C1a45B08Eb0e787F5F089077F8Bbfe/contracts

agUSDC

https://blockscout.com/xdai/mainnet/address/0x36328f69539a9FBAdfDd088Cd969cD9ec76bE24b/contracts

strategyUSDC

https://blockscout.com/xdai/mainnet/address/0xCDD313C81594bF588d9f816C82A33Cf83228cb13/contracts

stableDebtWXDAI

https://blockscout.com/xdai/mainnet/address/0xF8a239e4244dF1Fafebeb7F02fc182ea145c9D74/contracts

variableDebtWXDAI

https://blockscout.com/xdai/mainnet/address/0x15260BC6d3Ae02b62A607852EC92C261AE30D96D/contracts

agWXDAI

https://blockscout.com/xdai/mainnet/address/0xC73983C1dC24d6f997240f4f9074E3634A4a4246/contracts

strategyWXDAI

https://blockscout.com/xdai/mainnet/address/0xC170cb7fa2A5d43c2dD8260aDd2F1B06E7499fBE/contracts

stableDebtWBTC
https://blockscout.com/xdai/mainnet/address/0x0fb45A70b92545b7AA0a91B331ccb94B1d1e4Ca4/contracts

variableDebtWBTC
https://blockscout.com/xdai/mainnet/address/0x72A2AA64a59E138dDE48a66e788a502E93c2af0C/contracts

stableDebtWETH
https://blockscout.com/xdai/mainnet/address/0x89c5a17c381F4eaD9fF8478A3DC84F3eE65e4315/contracts

variableDebtWETH
https://blockscout.com/xdai/mainnet/address/0x1784F3277ABbBa95fBcb08a0a39b7C8cecee8769/contracts

agWBTC
https://blockscout.com/xdai/mainnet/address/0x6a4cCf5d642c85b6C9FD4Fd7B70d4878CD6b7c10/contracts

strategyWBTC
https://blockscout.com/xdai/mainnet/address/0x7C6BeaDe2cbEEC10B8CDe0D6Aea8206f302795c2/contracts

agWETH
https://blockscout.com/xdai/mainnet/address/0x85f5825BaB2C3550ae73EE90aCf33dbD8fF7Ae1a/contracts

strategyWETH
https://blockscout.com/xdai/mainnet/address/0x8dF939094b9906739DAeB3e76d85d2758E76075D/contracts

|LendingPoolCollateralManagerImpl
https://blockscout.com/xdai/mainnet/address/0x85b79018C781b499F2878BEe14E4e8B5DB31Ac61/contracts

|LendingPoolCollateralManager
https://blockscout.com/xdai/mainnet/address/0x85b79018C781b499F2878BEe14E4e8B5DB31Ac61/contracts

WalletBalanceProvider
https://blockscout.com/xdai/mainnet/address/0xC7Ecc651EBaA97a4E73b9128104d997064269db4/contracts

WETHGateway
https://blockscout.com/xdai/mainnet/address/0x0bb31c42D0692369Ba681A925C254fEB605c327b/contracts

AgaveOracle
https://blockscout.com/xdai/mainnet/address/0x80E08A2042F4135f6cA72BA2fd0e7cAEb2Ee30ef/contracts