LUKSO

LUKSO Standard Proposal (LSP)

SMART CONTRACT AUDIT

06.07.2022

Made in Germany by Chainsulting.de

Table of contents

1. Disclaimer 5
2. About the Project and Company 6
   2.1 Project Overview 7
3. Vulnerability & Risk Level 8
4. Auditing Strategy and Techniques Applied 9
   4.1 Methodology 9
5. Metrics 10
   5.1 Tested Contract Files 10
   5.2 Used Code from other Frameworks/Smart Contracts 15
   5.3 CallGraph LSP0 18
   5.3.1 CallGraph LSP1 19
   5.3.2 CallGraph LSP2 20
   5.3.3 CallGraph LSP3 21
   5.3.4 CallGraph LSP4 21
   5.3.5 CallGraph LSP5 22
   5.3.6 CallGraph LSP6 23
   5.3.7 CallGraph LSP7 24
   5.3.8 CallGraph LSP8 25
   5.3.9 CallGraph LSP9 26
   5.3.10 CallGraph LSP10 27
   5.4 Inheritance Graph LSP0 27
   5.4.1 Inheritance Graph LSP1 28

5.4.2 Inheritance Graph LSP2 28
5.4.3 Inheritance Graph LSP3 (Constants) 28
5.4.4 Inheritance Graph LSP4 29
5.4.5 Inheritance Graph LSP5 29
5.4.6 Inheritance Graph LSP6 30
5.4.7 Inheritance Graph LSP7 31
5.4.8 Inheritance Graph LSP8 32
5.4.9 Inheritance Graph LSP9 33
5.4.10 Inheritance Graph LSP10 (Constants) 33
5.5 Source Lines & Risk 34
5.6 Capabilities 35
5.7 Source Unites in Scope 36
6. Scope of Work 47
6.1 Findings Overview 48
6.2 Manual and Automated Vulnerability Test 49
6.2.1 Misleading Key Hash Value 49
6.2.2 Floating Pragma Version Identified 50
6.2.3 Unused Code 51
6.3 SWC Attacks 52
6.4 Test Deployment 56
6.4.1 LSP0 – ERC725 Account 56
6.4.2 LSP1 – Universal Receiver 56
6.4.3 LSP3 – Universal Profile Metadata 57
6.4.4 LSP5 – Received Assets 58

6.4.5 LSP6 – Key Manager 60
6.4.6 LSP9 – Vault 65
6.4.7 LSP10 – Received Vaults 68
6.4.8 LSP4 – Digital Asset Metadata 76
6.4.9 LSP7 – Digital Asset (Token) 77
6.4.10 LSP8 – Identifiable Digital Asset (NFT) 78
7. Executive Summary 80
8. About the Auditor 81

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 LUKSO Blockchain GmbH. 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 (15.06.2022)	Layout
0.4 (18.06.2022)	Automated Security Testing Manual Security Testing
0.5 (26.06.2022)	Test Deployment
0.6 (27.06.2022)	Testing SWC Checks
0.9 (01.07.2022)	Summary and Recommendation
1.0 (07.07.2022)	Final document

2. About the Project and Company

Company address:

LUKSO Blockchain GmbH
Köpenicker Chaussee 3a
10317 Berlin, Germany

Website: https://lukso.network

Twitter: https://twitter.com/lukso_io

Instagram: https://instagram.com/lukso

LinkedIn: https://linkedin.com/company/lukso

Telegram: https://t.me/LUKSO

Discord: https://discord.gg/lukso

Reddit: https://reddit.com/r/lukso

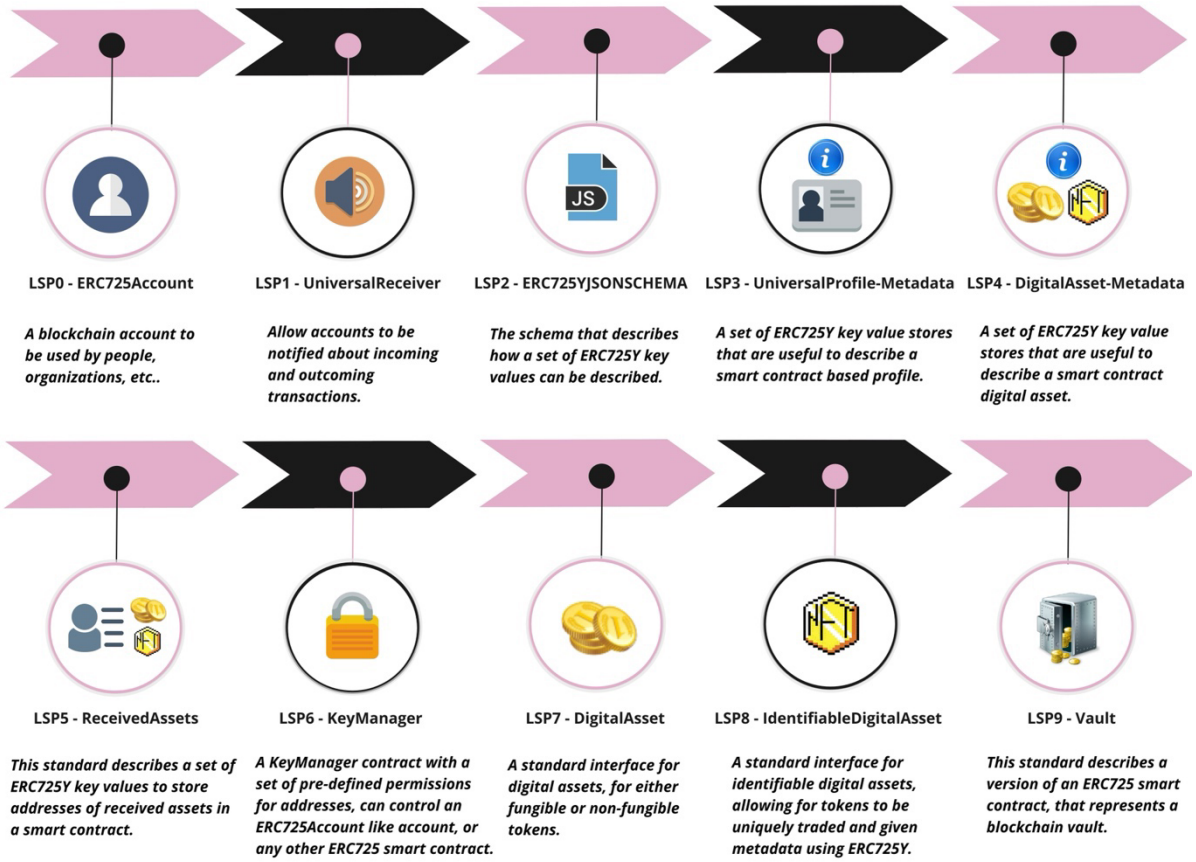
Medium: https://medium.com/lukso

GitHub: https://github.com/lukso-network

2.1 Project Overview

LUKSO is a Blockchain network dedicated to existing and coming digital lifestyle use cases. It will be running with a Casper consensus algorithm (PoS) combined with an Ethereum Virtual Machine (EVM) execution engine.

LUKSO Standard Proposals (LSPs) are smart contract standards which are the main building blocks of the LUKSO ecosystem. They can be used by developers to create more user friendly and interactable applications.



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.

5. Metrics

The metrics section should give the reader an overview on the size, quality, flows and capabilities of the codebase, without the knowledge to understand the actual code.

5.1 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)
./LSP10ReceivedVaults/LSP10Constants.sol	e8583bda7121402f3bd7ad552419aca1
./LSP5ReceivedAssets/LSP5Utils.sol	84466d5cf8fae431e1a6e566110027d3
./LSP5ReceivedAssets/LSP5Constants.sol	07dfbbb0316fbddec843cc42fa4e5dd9
./UniversalProfile.sol	0b572bebcbe309f4e73941dd26848f21
./LSP9Vault/LSP9VaultInit.sol	f8de70c22bfb31b4aa6175d320fcb86e
./LSP9Vault/LSP9Constants.sol	0b0533b928f099748b6c9db5b711bc8d
./LSP9Vault/LSP9VaultInitAbstract.sol	fc74396b3bbbd94c1c1af6338e698b5e
./LSP9Vault/LSP9Vault.sol	d5e617686449a04115adcfed650aaec3
./LSP9Vault/LSP9VaultCore.sol	5b934429c696eb8dc7fbfbc35072ce82
./LSP1UniversalReceiver/LSP1Constants.sol	8090e68e1a8c68f7465681ab0db6bd1b
./LSP1UniversalReceiver/LSP1UniversalReceiverDelegateUP/Handling/TokenAndVaultHandling.sol	c349de01eed05c01d4da05bece504efd
./LSP1UniversalReceiver/LSP1UniversalReceiverDelegateUP/LSP1UniversalReceiverDelegateUP.sol	983b0ddb1b14b52b59a299b073682936
./LSP1UniversalReceiver/ILSP1UniversalReceiverDelegate.sol	357ac33134f569997f01249c6a7d447e
./LSP1UniversalReceiver/ILSP1UniversalReceiver.sol	02e3b93369fe7d9a306cf1bd70e3a474
./LSP1UniversalReceiver/LSP1Utils.sol	88e0299cc55976542a4181c5c72780b8

./LSP1UniversalReceiver/LSP1UniversalReceiverDelegateVault/Handling/TokenHandling.sol	c566745c1a549ed4ce058ceda859016d
./LSP1UniversalReceiver/LSP1UniversalReceiverDelegateVault/LSP1UniversalReceiverDelegateVault.sol	c3f92d3504815336977f969ddfaee00b
./UniversalProfileInit.sol	80a821f6141f0cda41b4232882695eaa
./LSP6KeyManager/LSP6KeyManagerInit.	9ef0ba8f88b7a469f33c314ae64f9f98
./LSP6KeyManager/LSP6Constants.sol	df34df20e45f1e5e26cc350a5ffe1ec1
./LSP6KeyManager/LSP6KeyManager.sol	0827e5c672e36bf5e4b0f38c4400d283
./LSP6KeyManager/LSP6KeyManagerInitAbstract.sol	b4795d97379bdca67902c0d1df02e9ee
./LSP6KeyManager/ILSP6KeyManager.sol	31f6a25da820ead070009bbd99d33d75
./LSP6KeyManager/LSP6Utils.sol	2240a70b7e646c8ca866b6e6c46d99da
./LSP6KeyManager/LSP6Errors.sol	690f09840ff6ba191ca8327c7e524243
./LSP6KeyManager/LSP6KeyManagerCore.sol	6dd913309d75ce3f1cdbeeba3e638e01
./Legacy/Registries/AddressRegistry.sol	3d6872dfbc77dd04b2729275b1152799
./Legacy/Registries/AddressRegistryRequiresERC725.sol	3724949d7cf842d4f9ee75c38ea6d8e1
./Legacy/UniversalReceiverAddressStore.sol	e8159227dc8045a16b6fc6f4cacf851d
./Utils/UtilsLib.sol	d70eeb8fd5f8f984c5414035868fe1da
./UniversalProfileInitAbstract.sol	d41d8cd98f00b204e9800998ecf8427e
./LSP3UniversalProfile/LSP3Constants.	f44af4d634e0b122cbbbfba4e555a480
./Factories/Create2Factory.sol	f357cd9076d39f94c2d8e01d366d5e4c
./Factories/UniversalFactory.sol	4c44eb3c6b6441184a747e4b098a3e5a
./LSP0ERC725Account/LSP0Constants.sol	33fc9a3a465530ac0ac980ea5eeec498
./LSP0ERC725Account/LSP0ERC725Account.sol	678140f7fa511e88dbdc5d4263b49948
./LSP0ERC725Account/LSP0ERC725AccountCore.sol	61f677ffbb4c8ef617586f418d60dcc4
./LSP0ERC725Account/LSP0ERC725AccountInit.sol	3dabe54a0882925c1465ebcd4a753b0a
./LSP0ERC725Account/LSP0ERC725AccountInitAbstract.sol	f63c20abdcfb11748b8344515e31eee1
./LSP4DigitalAssetMetadata/LSP4DigitalAssetMetadataInit.sol	41f1561b9e99352b25af06543ba1d1ea
./LSP4DigitalAssetMetadata/LSP4DigitalAssetMetadataInitAbstract.sol	fa00b327d324d44bf668c928dffc8fb4
./LSP4DigitalAssetMetadata/LSP4Compatibility.sol	6a471a4789cdeabb18914c2954d7d6de
./LSP4DigitalAssetMetadata/LSP4DigitalAssetMetadata.sol	0acc6a17f5f12cf8eafccd2a4d312e54
./LSP4DigitalAssetMetadata/LSP4Constants.sol	ab16d584e82725955154ff6d683fd286

./LSP4DigitalAssetMetadata/ILSP4Compatibility.sol	07d466836e289b01bc454166878f3791
./LSP8IdentifiableDigitalAsset/LSP8IdentifiableDigitalAssetCore.sol	d952ff522757221fe522dd29a048c33c
./LSP8IdentifiableDigitalAsset/LSP8IdentifiableDigitalAsset.sol	20cd37f56fdf7e32f1509814983c390c
./LSP8IdentifiableDigitalAsset/LSP8Errors.sol	a498ed111477d427129c7d7d7296861a
./LSP8IdentifiableDigitalAsset/LSP8IdentifiableDigitalAssetInitAbstract.sol	dd75dbde62ed5815d9e7f32eea47bfe1
./LSP8IdentifiableDigitalAsset/extensions/ILSP8CompatibilityForERC721.sol	1317e3286fb9311d38f5277295236513
./LSP8IdentifiableDigitalAsset/extensions/LSP8CompatibilityConstants.sol	d9d241dc0963bd4d3490642fe7c358ed
./LSP8IdentifiableDigitalAsset/extensions/LSP8CompatibilityForERC721Init.sol	c06116f18bca249cdfafb5c9b018c656
./LSP8IdentifiableDigitalAsset/extensions/LSP8CompatibilityForERC721InitAbstract.sol	4adec418acd5cbb48133bb7f6a32d4a9
./LSP8IdentifiableDigitalAsset/extensions/LSP8CappedSupplyCore.sol	f15e474a9d0beb24c36ae2c7ca6d20e4
./LSP8IdentifiableDigitalAsset/extensions/LSP8CompatibilityForERC721.sol	34c865865b14acf1799ff133bc9577c5
./LSP8IdentifiableDigitalAsset/extensions/ILSP8CappedSupply.sol	279abe1b7d5c7f5654c7ac532f753d36
./LSP8IdentifiableDigitalAsset/extensions/LSP8CappedSupplyInit.sol	1e06acdd3436b905ba5729b48d808a20
./LSP8IdentifiableDigitalAsset/extensions/LSP8CappedSupply.sol	9a7e0f25a5351f98db4d9ebd99c708ac
./LSP8IdentifiableDigitalAsset/extensions/LSP8CompatibilityForERC721Core.sol	390c9746c5a7cc4ab94aadde30861b8b
./LSP8IdentifiableDigitalAsset/extensions/LSP8CappedSupplyInitAbstract.sol	d1e4c975926b857577095a7112374e22
./LSP8IdentifiableDigitalAsset/LSP8Constants.sol	6b84b253f02e13ef094c670d0668243a
./LSP8IdentifiableDigitalAsset/ILSP8IdentifiableDigitalAsset.sol	94ec3ed5c558e5059396d4ff9657ba9a
./LSP8IdentifiableDigitalAsset/presets/LSP8MintableInitAbstract.sol	79ce7c7afe859528ade904c4043c6c72
./LSP8IdentifiableDigitalAsset/presets/LSP8MintableInit.sol	fb01559fdba090c268548a80ff506a12
./LSP8IdentifiableDigitalAsset/presets/ILSP8Mintable.sol	202725098780aab6faec1fc66467f45d
./LSP8IdentifiableDigitalAsset/presets/LSP8Mintable.sol	055eed90642322c5e491ad928fbedc02
./LSP8IdentifiableDigitalAsset/presets/LSP8MintableCore.sol	5e18eba2d84a33bc0412680ad65a45a9
./LSP8IdentifiableDigitalAsset/LSP8IdentifiableDigitalAssetInit.sol	093543f448fe7d538484058161101813
./LSP7DigitalAsset/ILSP7DigitalAsset.sol	e344bc8680be449b2ba808d3c4db1c75
./LSP7DigitalAsset/LSP7Constants.sol	6df5b52e662ac45c4fdef2b7c93ce105
./LSP7DigitalAsset/LSP7DigitalAssetInitAbstract.sol	892526478f2cfee26486b7cbed05f671
./LSP7DigitalAsset/LSP7DigitalAssetInit.sol	9e88a50c82939b207b0047cfeee13d96
./LSP7DigitalAsset/extensions/LSP7CompatibilityForERC20Init.sol	fdc6f5a8b295114d27ecca0720ad4e35
./LSP7DigitalAsset/extensions/LSP7CappedSupplyCore.sol	3a7456e3f3c8a726a8976cec078f6bf9
./LSP7DigitalAsset/extensions/ILSP7CappedSupply.sol	4c5bbe8d5700e8723aa734b6d50e0404

./LSP7DigitalAsset/extensions/ILSP7CompatibilityForERC20.sol	51f678e6bb414fc0c705f1f150a1e4b2
./LSP7DigitalAsset/extensions/LSP7CappedSupplyInit.sol	c163e1cbc3feee35ae54acd21ed7aa71
./LSP7DigitalAsset/extensions/LSP7CappedSupplyInitAbstract.sol	20c5e07538bc5c9a5dd96d27d4fa1947
./LSP7DigitalAsset/extensions/LSP7CappedSupply.sol	81cb749293cab82c5c09257777545ff3
./LSP7DigitalAsset/extensions/LSP7CompatibilityForERC20Core.sol	504387343343be21728992520b36e460
./LSP7DigitalAsset/extensions/LSP7CompatibilityForERC20.sol	519cc5c56f98a4644b04555e93940bd2
./LSP7DigitalAsset/extensions/LSP7CompatibilityForERC20InitAbstract.sol	ff02731a58e5df73c2da73cc083fde1d
./LSP7DigitalAsset/LSP7DigitalAssetCore.sol	9c1a3a6886a6829b751a9747304e668f
./LSP7DigitalAsset/LSP7DigitalAsset.sol	29ff1a24ec3b1bb3ba7b9c3e6b496d02
./LSP7DigitalAsset/LSP7Errors.sol	9e97adbbd983e576503d507cc6e97b4a
./LSP7DigitalAsset/presets/LSP7MintableInitAbstract.sol	5b2caf80a6692539562ead4fe815bab4
./LSP7DigitalAsset/presets/LSP7MintableInit.sol	1a8802cc94fc206a2992d2558cbca9eb
./LSP7DigitalAsset/presets/ILSP7Mintable.sol	a05b2d5b7cce1f690b1730d612b818ee
./LSP7DigitalAsset/presets/LSP7MintableCore.sol	c17877af14a48979722595ba17e3ea5f
./LSP7DigitalAsset/presets/LSP7Mintable.sol	d530fc634877b9ec2c696003be4a793f
./Custom/IClaimOwnership.sol	bf3d8b3cfc417350d3828ec3ca96b11e
./Custom/ClaimOwnership.sol	c077e9758320d6681f3e3524d7052495
./Custom/ERC165Checker.sol	93dfadfc8f6e4f86fc25b534b4d4be31
./Helpers/TargetContract.sol	d8f8999031f916a0513ba1c0bd36632a
./Helpers/Security/Reentrancy.sol	cc7df99eb56b7655fc883db9fd0c167d
./Helpers/SignatureValidatorContract.sol	d901f3ff8d9b8483dddd0dff974aea79
./Helpers/FallbackContract.sol	ac00d3bed3ea32b05ce63b7df0ab9194
./Helpers/UniversalReceivers/UniversalReceiverTester.sol	96ccbb9648a67b44890fc6f2f7d79e56
./Helpers/UniversalReceivers/UniversalReceiverDelegateVaultSetter.sol	b4acf2200d69009711e261f35ebc2ce3
./Helpers/UniversalReceivers/UniversalReceiverDelegateRevert.sol	0023bbf88d89bb654ab26a9415db1b24
./Helpers/PayableContract.sol	0691af194a097cd23c8254d89704cbc6
./Helpers/LSP2UtilsLibraryTester.sol	14aa5c5ebb61aa72453719429a8f341f
./Helpers/ERC165Interfaces.sol	456d4bc7446dd70fe759ddfae1df8489
./Helpers/Tokens/LSP7Tester.sol	cbc478bff895a0c03ba0f54ee7a1b965
./Helpers/Tokens/IERC223.sol	70d5c6934c715ba62347817e957f46aa
./Helpers/Tokens/LSP7CappedSupplyInitTester.sol	ef80d71b2f76801a202eae4562550137

./Helpers/Tokens/LSP8CappedSupplyInitTester.sol	adaf5b2b54869208b9ed9d496da15aa5
./Helpers/Tokens/LSP8CompatibilityForERC721Tester.sol	4304c06e5705aa5a6e1f98bd355382f9
./Helpers/Tokens/LSP8InitTester.sol	344baa6bd86097c716fd35194d2fb539
./Helpers/Tokens/TokenReceiverWithLSP1.sol	366eb0859ad392817cdaee7bbdf34f83
./Helpers/Tokens/LSP7CappedSupplyTester.sol	8491ef89856b3e5a6eec243405229966
./Helpers/Tokens/LSP8Tester.sol	0f9899379b5bd43fb8c00eb284021b48
./Helpers/Tokens/LSP8CompatibilityForERC721TesterInit.sol	277baf0703735a4ab05e77c8406882db
./Helpers/Tokens/LSP8CappedSupplyTester.sol	c76757cec70d8d2125c32745efb60f0f
./Helpers/Tokens/LSP4CompatibilityTester.sol	92fb939c7271aae75901737994f1385f
./Helpers/Tokens/TokenReceiverWithoutLSP1.sol	9f369e65cda0327eced880583f4a8fb7
./Helpers/Tokens/LSP7CompatibilityForERC20InitTester.sol	49e03f9d28c175cfe280e7a18609c51c
./Helpers/Tokens/LSP7CompatibilityForERC20Tester.sol	615cce80747215ae704d6ab6ddcbd9bb
./Helpers/Tokens/LSP7InitTester.sol	12fd3b11e7ecc9c54e8ecd755d297f42
./Helpers/ERC165CheckerCustomTest.sol	90654bdf655f295c789428f26330de16
./Helpers/NFTStorage.sol	1fea29329aca2fd1308442d79938f3b6
./Helpers/Executor.sol	4aee80c2a8d442cb574c26d865da5f6f
./Helpers/KeyManager/KeyManagerInternalsTester.sol	b2ff01d0403475f5b0b2c642b7319a7b
./Helpers/KeyManager/TargetPayableContract.sol	307121cc885fc450f236744dee224f0d
./Helpers/KeyManager/ERC725YDelegateCall.sol	0abaf6e6a6ec794122b4e08e7af2044a
./LSP2ERC725YJSONSchema/LSP2Utils.sol	f20be00ef3c0bd7f2f95c464188560e0

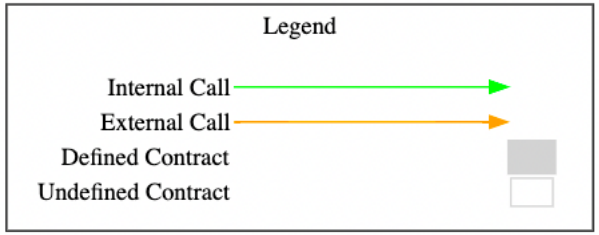
5.2 Used Code from other Frameworks/Smart Contracts (direct imports)

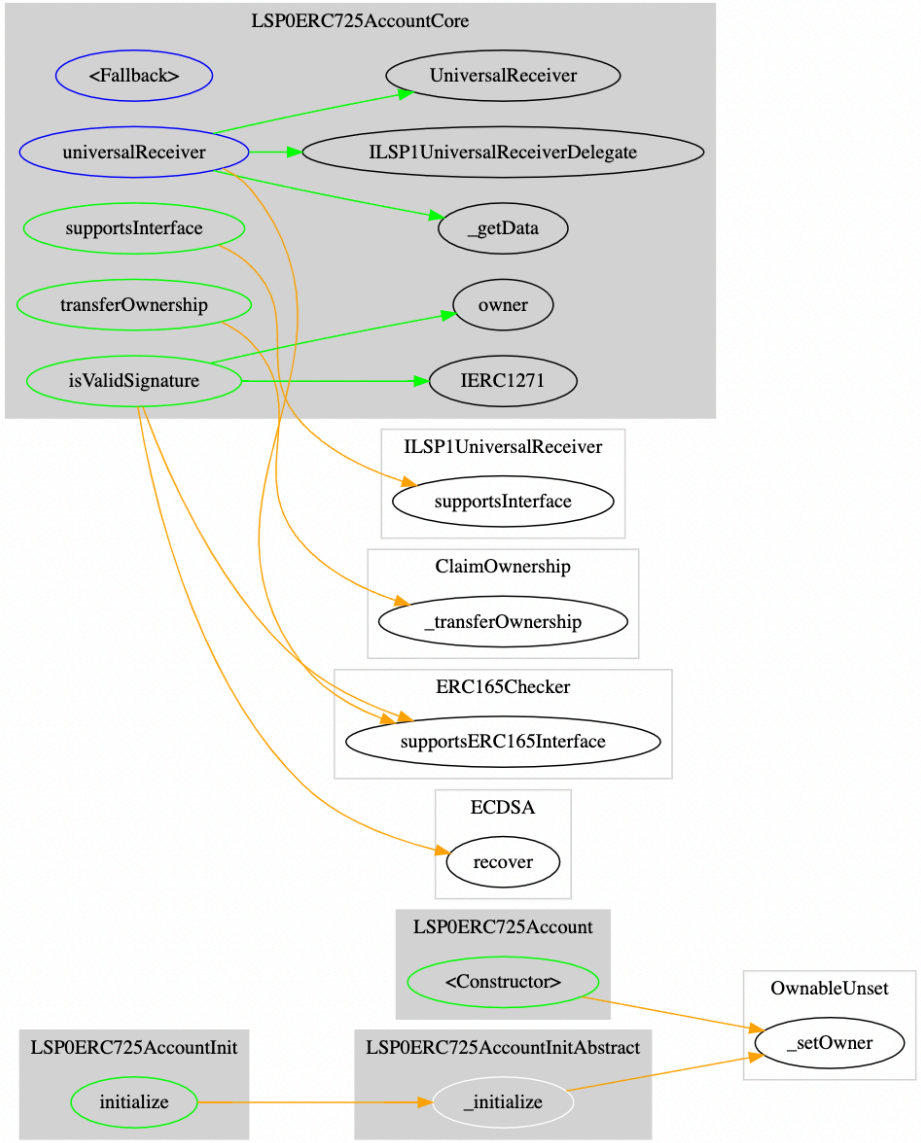
Dependency / Import Path	Source
@erc725/smart-contracts/contracts/ERC725.sol	https://github.com/ERC725Alliance/ERC725/tree/v3.1.1/implementations/contracts/ERC725.sol
@erc725/smart-contracts/contracts/ERC725XCore.sol	https://github.com/ERC725Alliance/ERC725/tree/v3.1.1/implementations/contracts/ERC725XCore.sol
@erc725/smart-contracts/contracts/ERC725Y.sol	https://github.com/ERC725Alliance/ERC725/tree/v3.1.1/implementations/contracts/ERC725Y.sol
@erc725/smart-contracts/contracts/ ERC725Y.sol	https://github.com/ERC725Alliance/ERC725/tree/v3.1.1/implementations/contracts/ERC725Y.sol
@erc725/smart-contracts/contracts/ERC725YInitAbstract.sol	https://github.com/ERC725Alliance/ERC725/tree/v3.1.1/implementations/contracts/
@erc725/smart-contracts/contracts/constants.sol	https://github.com/ERC725Alliance/ERC725/tree/v3.1.1/implementations/contracts/constants.sol
@erc725/smart-contracts/contracts/custom/Initializable.sol	https://github.com/ERC725Alliance/ERC725/tree/v3.1.1/implementations/contracts/Initializable.sol
@erc725/smart-contracts/contracts/custom/OwnableUnset.sol	https://github.com/ERC725Alliance/ERC725/tree/v3.1.1/implementations/contracts/OwnableUnset.sol
@erc725/smart-contracts/contracts/interfaces/IERC725X.sol	https://github.com/ERC725Alliance/ERC725/tree/v3.1.1/implementations/contracts/interfaces/IERC725X.sol

Dependency / Import Path	Source
@erc725/smart-contracts/contracts/interfaces/IERC725Y.sol	https://github.com/ERC725Alliance/ERC725/tree/v3.1.1/implementations/contracts/interfaces/IERC725Y.sol
@erc725/smart-contracts/contracts/utils/ErrorHandlerLib.sol	https://github.com/ERC725Alliance/ERC725/tree/v3.1.1/implementations/contracts/utils/ErrorHandlerLib.sol
@openzeppelin/contracts/interfaces/IERC1271.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/interfaces/IERC1271.sol
@openzeppelin/contracts/proxy/Clones.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/proxy/Clones.sol
@openzeppelin/contracts/proxy/utils/Initializable.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/proxy/utils/Initializable.sol
@openzeppelin/contracts/token/ERC1155/IERC1155.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/token/ERC1155/IERC1155.sol
@openzeppelin/contracts/token/ERC20/IERC20.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/token/ERC20/IERC20.sol
@openzeppelin/contracts/token/ERC721/IERC721.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/token/ERC721/IERC721.sol
@openzeppelin/contracts/token/ERC721/extensions/IERC721Metadata.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/token/ERC721/extensions/IERC721Metadata.sol
@openzeppelin/contracts/token/ERC777/IERC777.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/token/ERC777/IERC777.sol

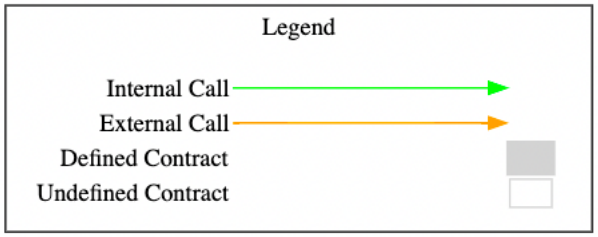
Dependency / Import Path	Source
@openzeppelin/contracts/utils/Address.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/utils/Address.sol
@openzeppelin/contracts/utils/Context.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/utils/Context.sol
@openzeppelin/contracts/utils/Create2.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/utils/Create2.sol
@openzeppelin/contracts/utils/cryptography/ECDSA.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/utils/cryptography/ECDSA.sol
@openzeppelin/contracts/utils/cryptography/MerkleProof.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/utils/cryptography/MerkleProof.sol
@openzeppelin/contracts/utils/introspection/ERC165.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/utils/introspection/ERC165.sol
@openzeppelin/contracts/utils/introspection/ERC165Storage.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/utils/introspection/ERC165Storage.sol
@openzeppelin/contracts/utils/introspection/IERC165.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/utils/introspection/IERC165.sol
@openzeppelin/contracts/utils/structs/EnumerableSet.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.6.0/contracts/utils/structs/EnumerableSet.sol
solidity-bytes-utils/contracts/BytesLib.sol	https://github.com/GNSPS/solidity-bytes-utils/tree/v0.8.0/contracts/BytesLib.sol

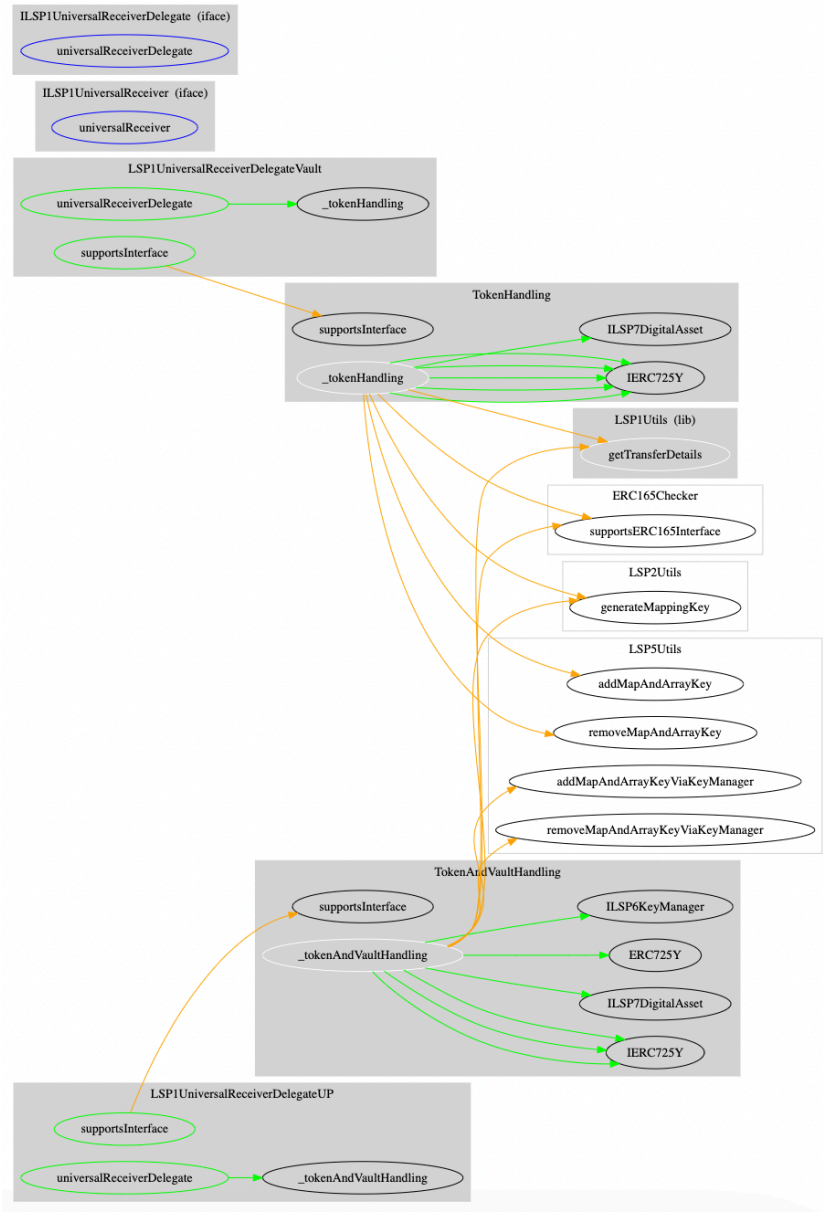
5.3 CallGraph LSP0



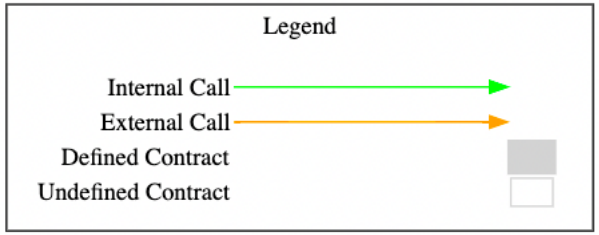


5.3.1 CallGraph LSP1



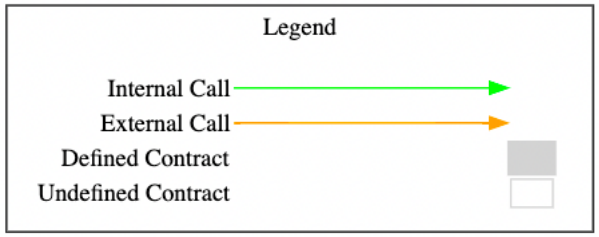


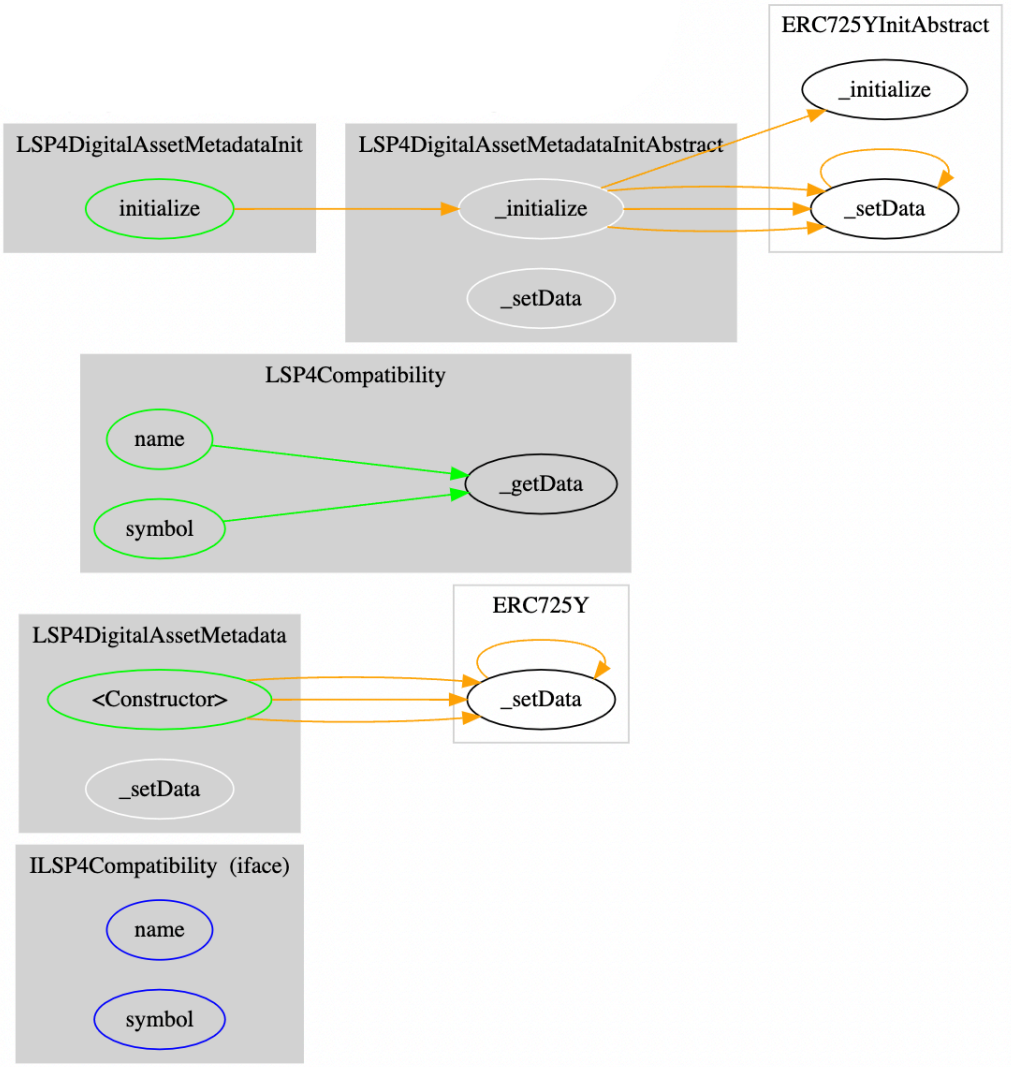
5.3.2 CallGraph LSP2



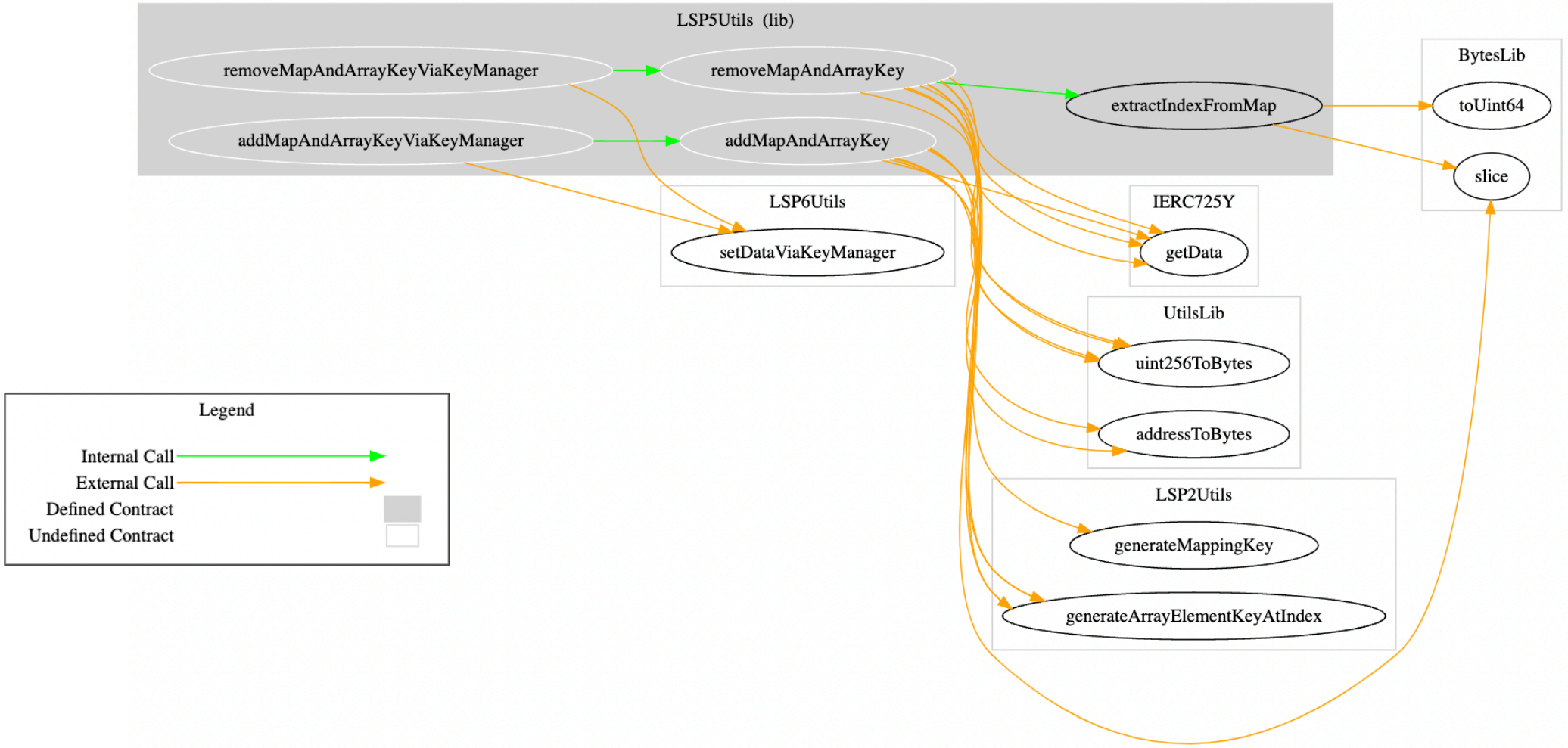
5.3.3 CallGraph LSP3 (Constants)

5.3.4 CallGraph LSP4

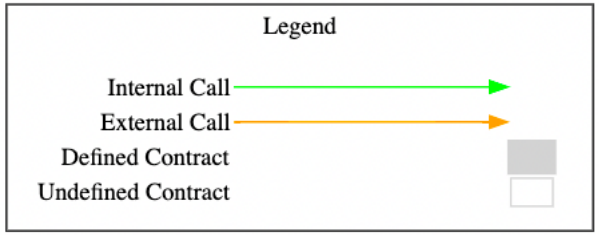


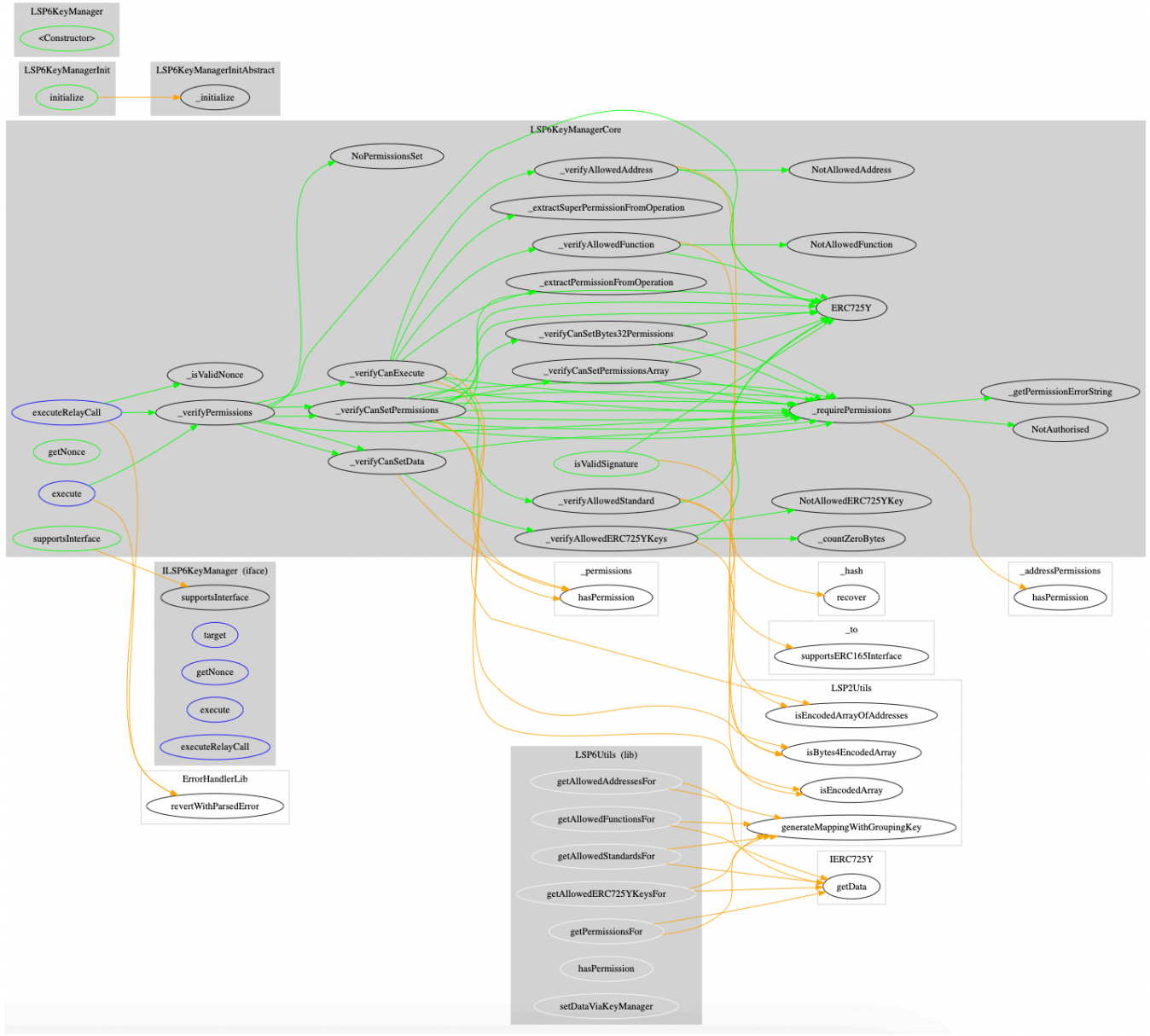


5.3.5 CallGraph LSP5

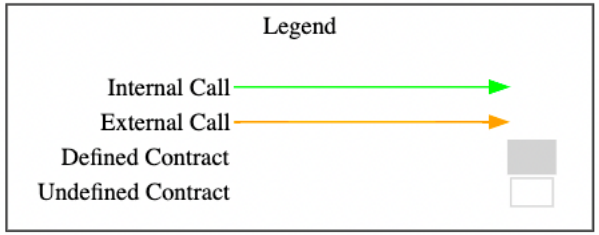


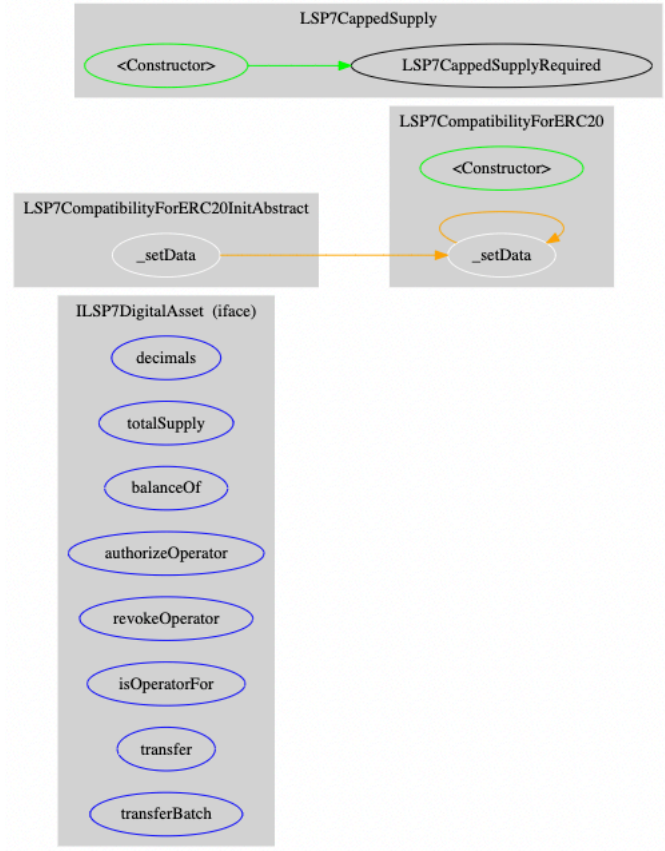
5.3.6 CallGraph LSP6

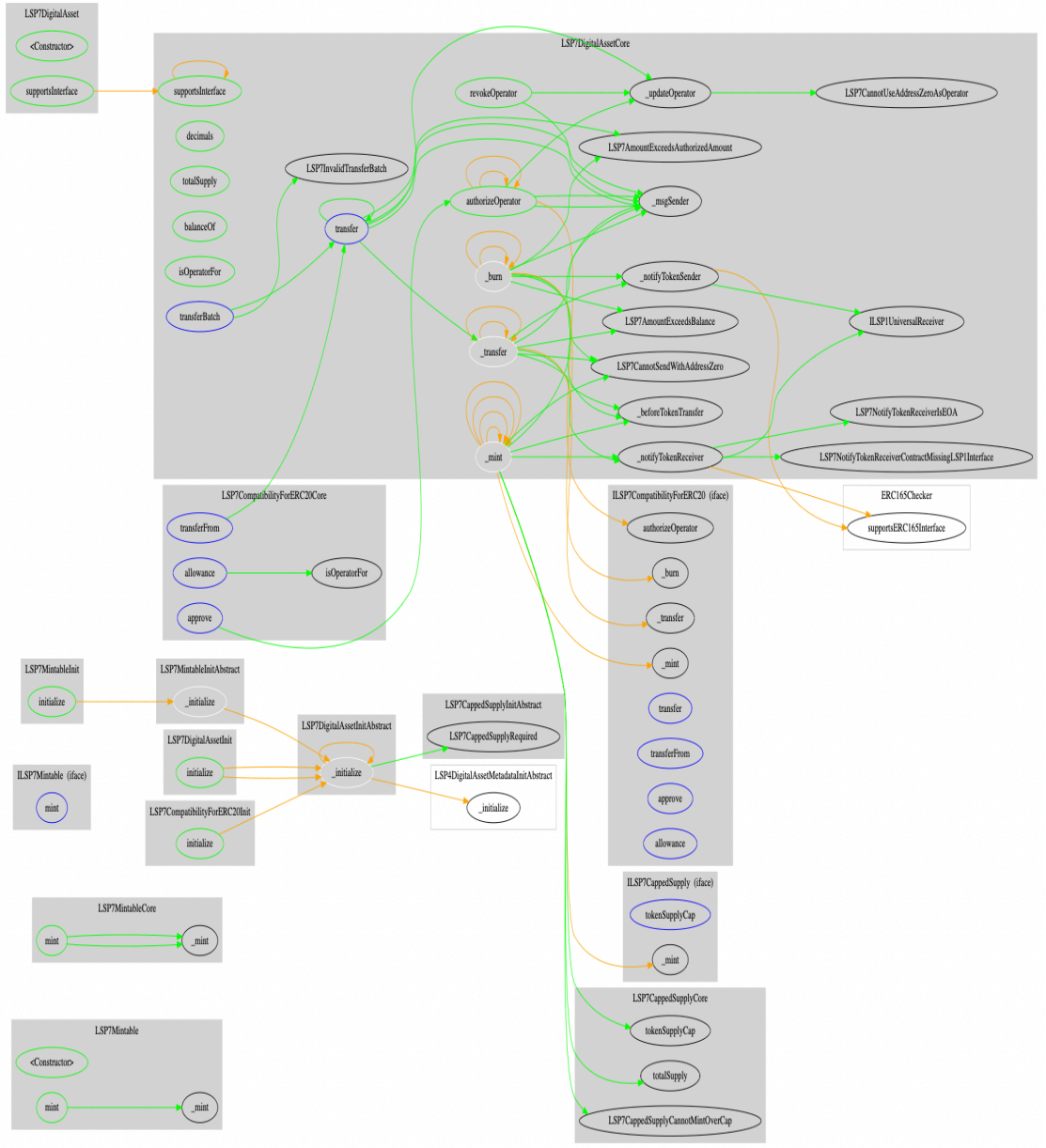




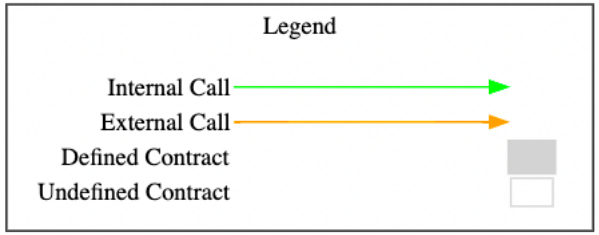
5.3.7 CallGraph LSP7

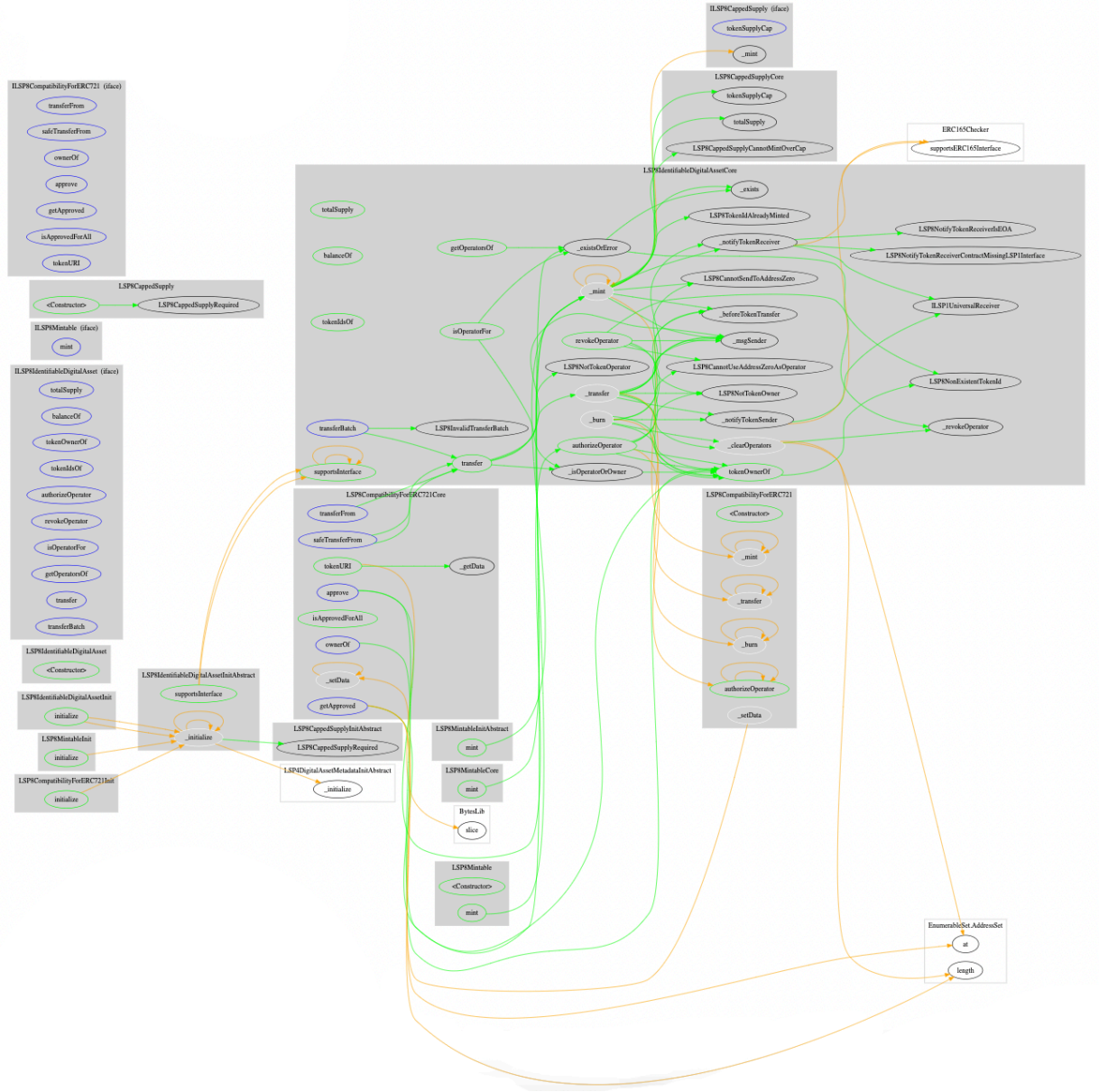






5.3.8 CallGraph LSP8

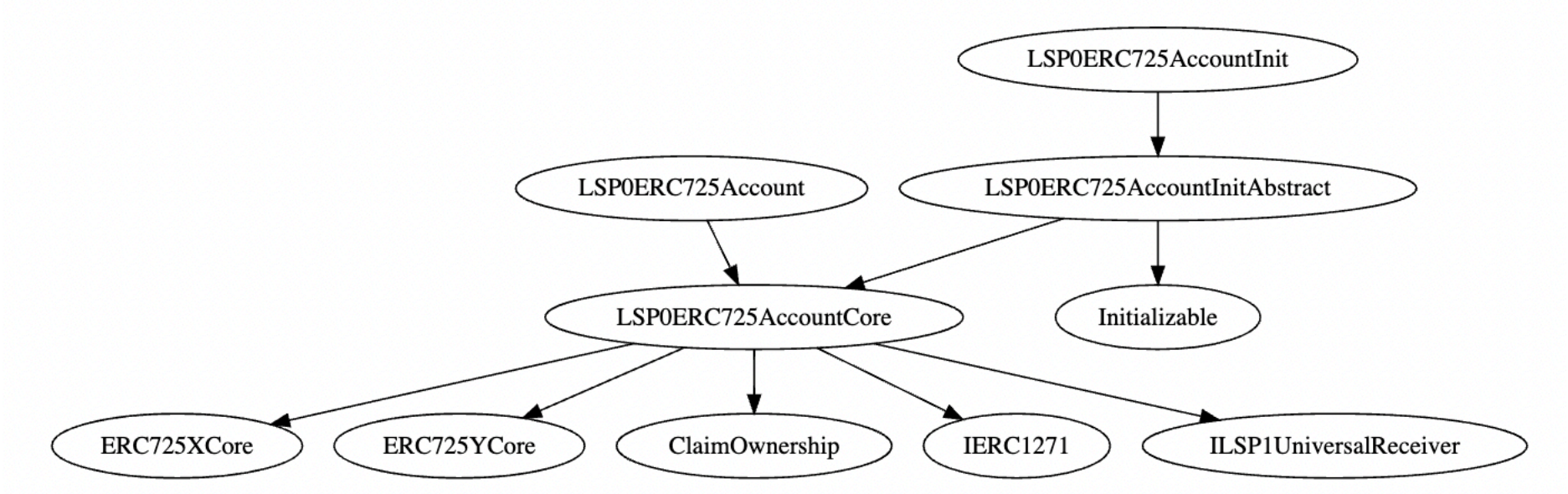




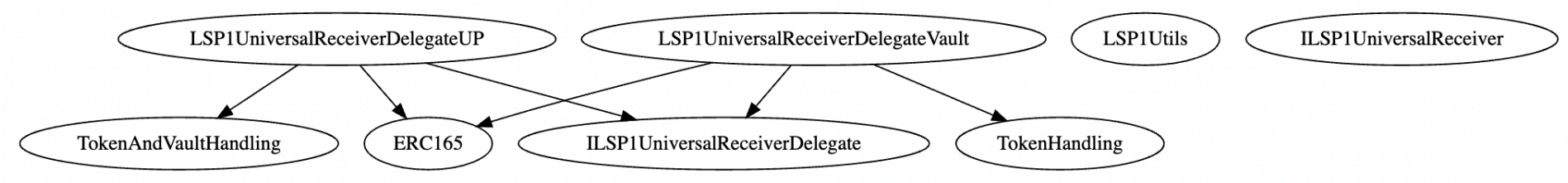
5.3.9 CallGraph LSP9

5.3.10 CallGraph LSP10 (Constants)

5.4 Inheritance Graph LSP0



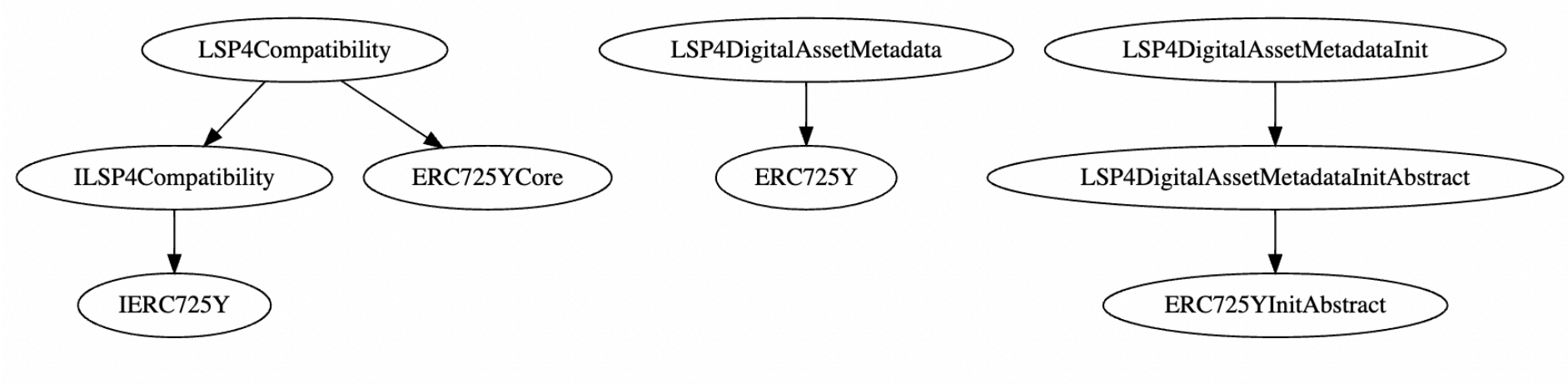
5.4.1 Inheritance Graph LSP1



5.4.2 Inheritance Graph LSP2

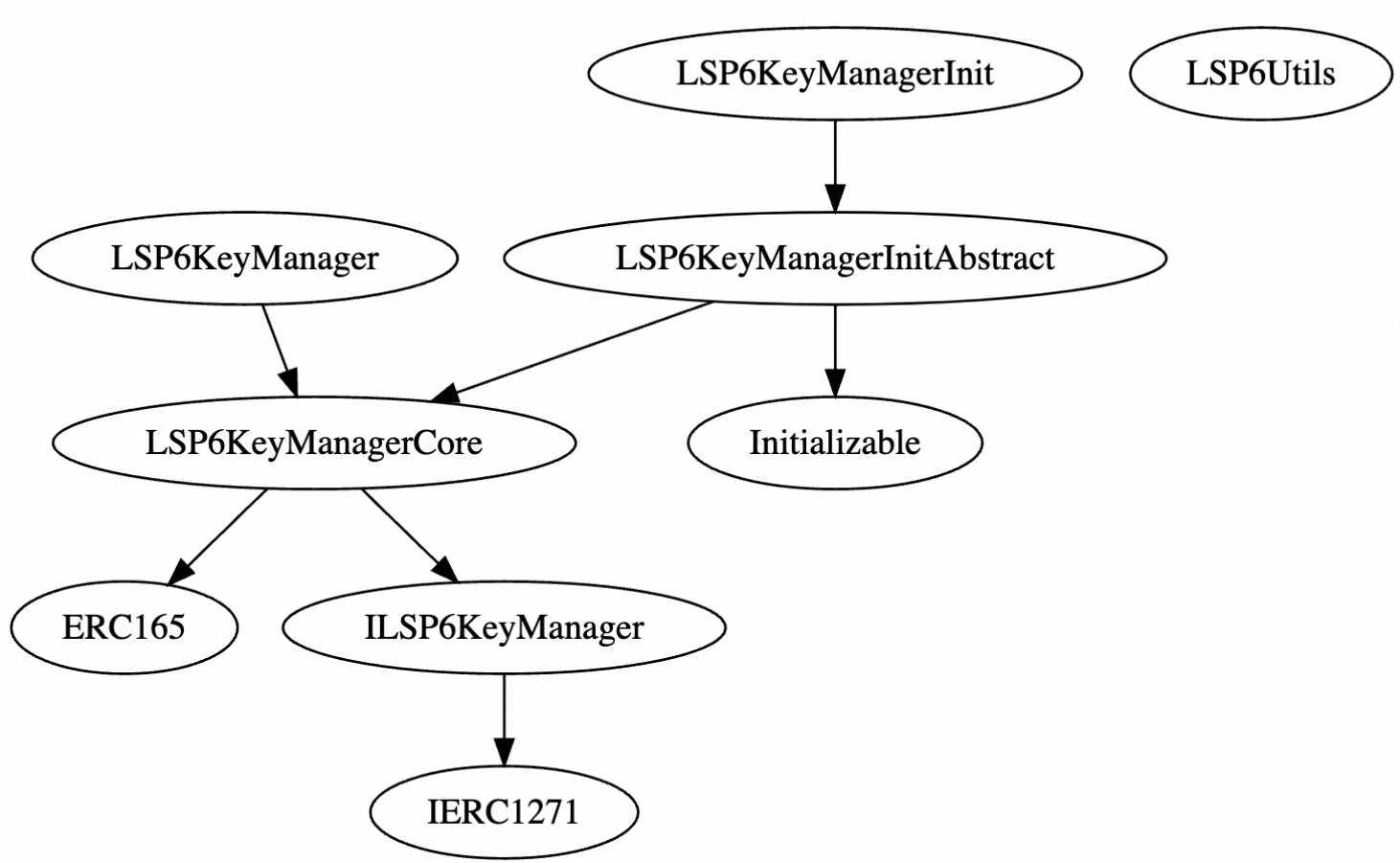
5.4.3 Inheritance Graph LSP3 (Constants)

5.4.4 Inheritance Graph LSP4

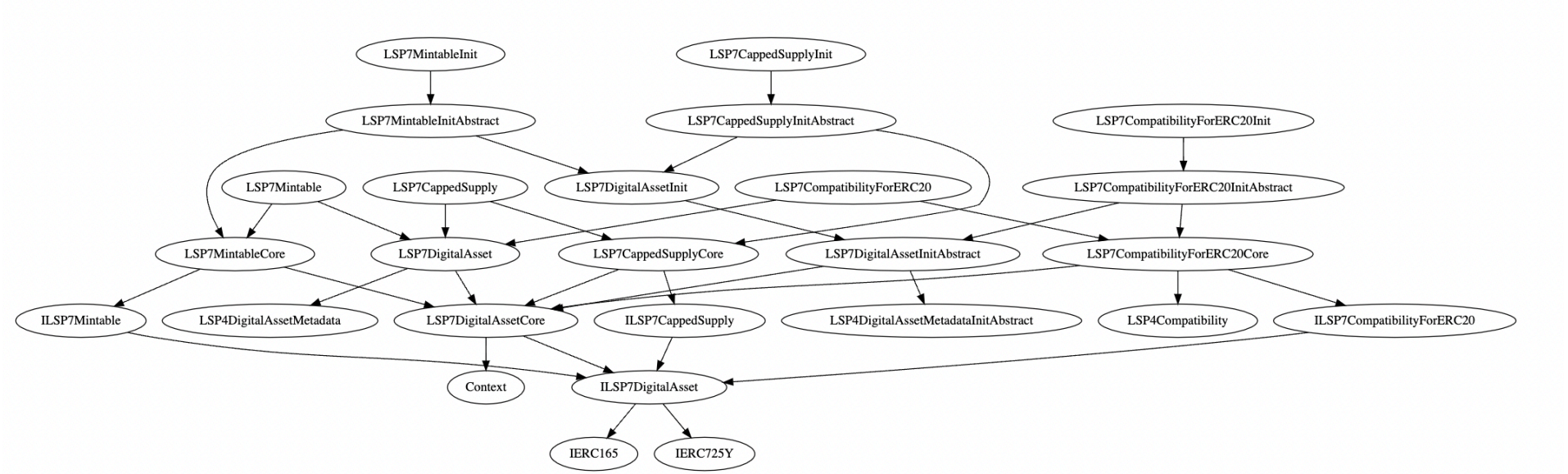


5.4.5 Inheritance Graph LSP5

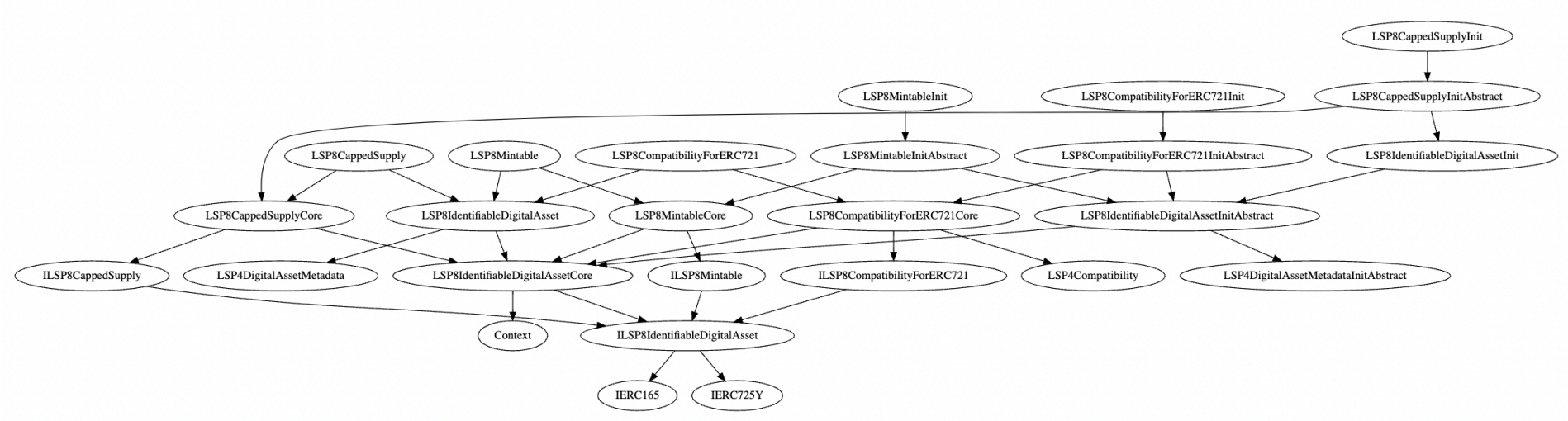
5.4.6 Inheritance Graph LSP6



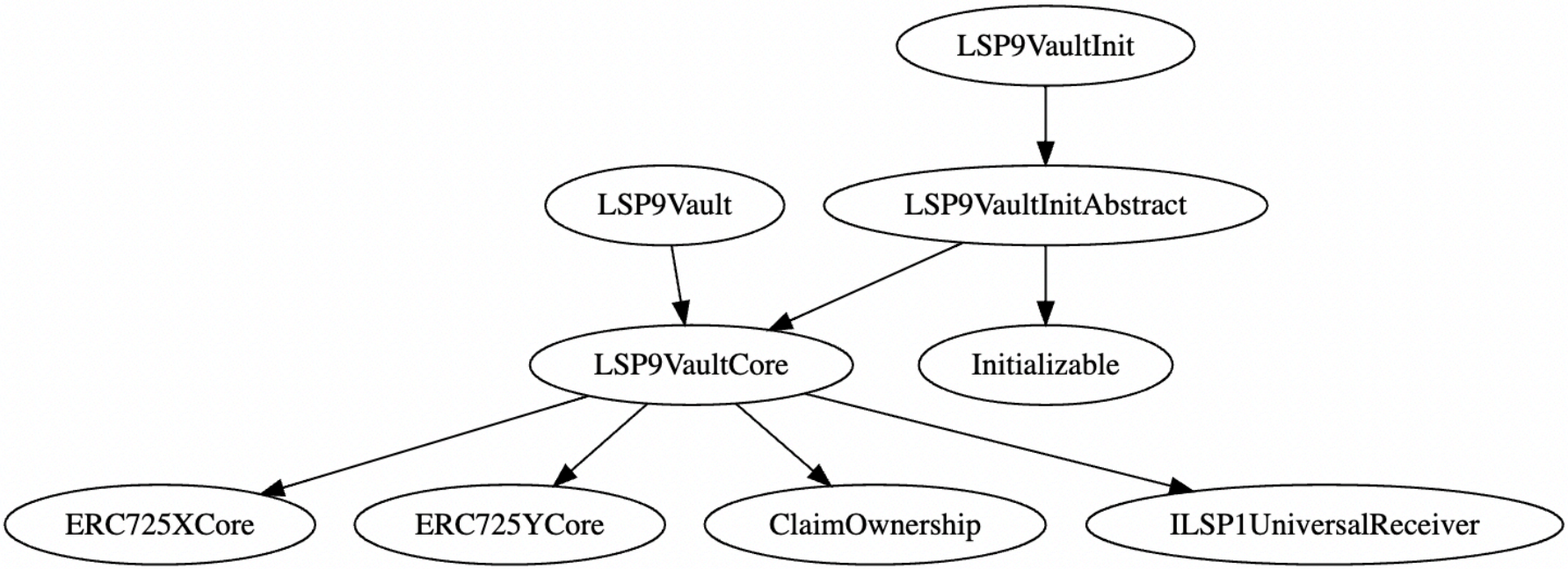
5.4.7 Inheritance Graph LSP7



5.4.8 Inheritance Graph LSP8

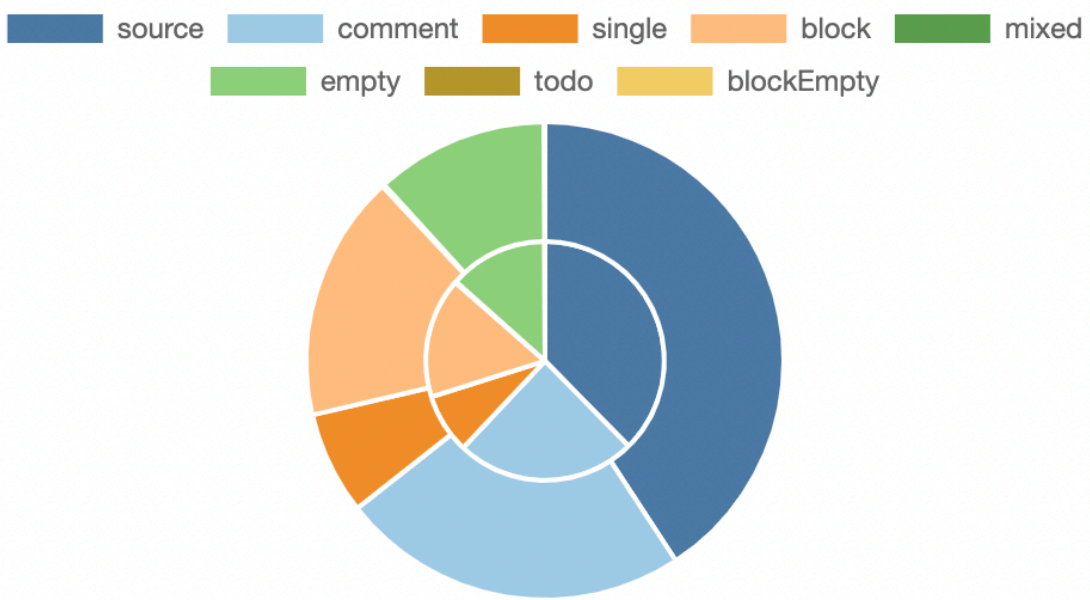


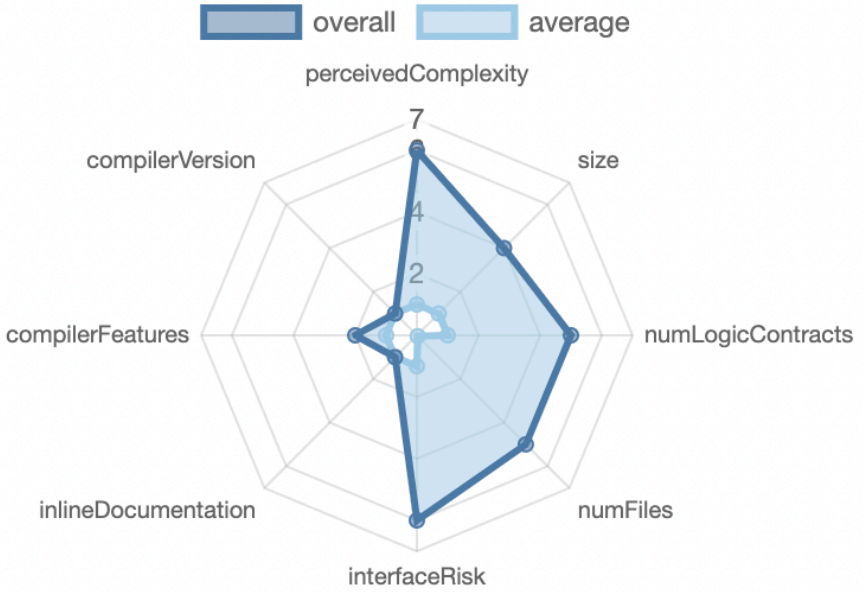
5.4.9 Inheritance Graph LSP9



5.4.10 Inheritance Graph LSP10 (Constants)

5.5 Source Lines & Risk





5.6 Capabilities

Solidity Versions observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
`^0.8.0` `^0.8.6`		`yes`	`yes` (6 asm blocks)	

Transfers ETH	Low-Level Calls	DelegateCall	Uses Hash Functions	ECRecover	New/Create/Create2
`yes`			`yes`		`yes` → `AssemblyCall:Name:create2`

Exposed Functions

This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.

Public	Payable
250	26

External	Internal	Private	Pure	View
77	338	0	49	103

StateVariables

Total	Public
29	4

5.7 Source Unites in Scope

Source: https://github.com/lukso-network/lsp-smart-contracts/tree/v0.6.1
Last commit: ac4133b40c10e0b73227caa6f3a2b88b32bf4403

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	contracts/LSP2ERC725YJSONSchema/LSP2Utils.sol	1		205	173	107	25	93	
	contracts/Helpers/KeyManager/ERC725YDelegateCall.sol	1		13	13	8	2	6	
	contracts/Helpers/KeyManager/TargetPayableContract.sol	1		10	10	7	1	7	
	contracts/Helpers/KeyManager/KeyManagerInternalsTester.sol	1		59	55	35	7	35	
	contracts/Helpers/Executor.sol	1		179	172	114	17	63	
	contracts/Helpers/NFTStorage.sol	1		15	11	7	2	4	
	contracts/Helpers/ERC165CheckerCustomTest.sol	1		18	14	7	5	4	

📝	contracts/Helpers/Tokens/LSP7InitTester.sol	1		33	20	13	2	13	
📝	contracts/Helpers/Tokens/LSP7CompatibilityForERC20Tester.sol	1		33	25	16	4	10	
📝	contracts/Helpers/Tokens/LSP7CompatibilityForERC20InitTester.sol	1		34	22	14	3	13	
📝	contracts/Helpers/Tokens/TokenReceiverWithoutLSP1.sol	1		9	9	5	2	11	💰
📝	contracts/Helpers/Tokens/LSP4CompatibilityTester.sol	1		21	21	14	3	10	
📝	contracts/Helpers/Tokens/LSP8CappedSupplyTester.sol	1		25	25	17	3	11	
📝	contracts/Helpers/Tokens/LSP8CompatibilityForERC721TesterInit.sol	1		36	27	16	4	16	
📝	contracts/Helpers/Tokens/LSP8Tester.sol	1		28	23	15	3	10	
📝	contracts/Helpers/Tokens/LSP7CappedSupplyTester.sol	1		26	26	17	4	11	
📝	contracts/Helpers/Tokens/TokenReceiverWithLSP1.sol	1		35	30	16	5	22	💰
📝	contracts/Helpers/Tokens/LSP8InitTester.sol	1		29	20	13	2	13	
📝	contracts/Helpers/Tokens/LSP8CompatibilityForERC721Tester.sol	1		34	30	20	4	13	
📝	contracts/Helpers/Tokens/LSP8CappedSupplyInitTester.sol	1		27	22	15	2	14	

	contracts/Helpers/Tokens/LSP7CappedSupplyInitTester.sol	1		28	23	15	3	14	
	contracts/Helpers/Tokens/IERC223.sol	1		54	10	3	23	17	
	contracts/Helpers/Tokens/LSP7Tester.sol	1		32	23	15	3	10	
	contracts/Helpers/ERC165Interfaces.sol	2		169	169	125	14	62	
	contracts/Helpers/LSP2UtilsLibraryTester.sol	1		12	12	8	1	4	
	contracts/Helpers/PayableContract.sol	1		16	16	6	7	11	
	contracts/Helpers/UniversalReceivers/UniversalReceiverDelegateRevert.sol	1		29	24	13	6	10	
	contracts/Helpers/UniversalReceivers/UniversalReceiverDelegateVaultSetter.sol	1		31	27	17	4	30	
	contracts/Helpers/UniversalReceivers/UniversalReceiverTester.sol	1		30	26	17	3	28	
	contracts/Helpers/FallbackContract.sol	1		9	9	4	4	6	
	contracts/Helpers/SignatureValidatorContract.sol	1		34	29	11	14	9	
	contracts/Helpers/Security/Reentrancy.sol	1		21	21	14	3	11	
	contracts/Helpers/TargetContract.sol	1		37	37	20	10	14	
	contracts/Custom/ERC165Checker.sol	1		130	118	42	63	39	

	contracts/Custom/ClaimOwnership.sol	1		31	31	17	6	15	
	contracts/Custom/IClaimOwnership.sol		1	10	7	4	1	5	
	contracts/LSP7DigitalAsset/presets/LSP7Mintable.sol	1		42	37	14	18	10	
	contracts/LSP7DigitalAsset/presets/LSP7MintableCore.sol	1		26	21	8	9	8	
	contracts/LSP7DigitalAsset/presets/ILSP7Mintable.sol		1	32	26	4	19	5	
	contracts/LSP7DigitalAsset/presets/LSP7MintableInit.sol	1		27	22	7	12	7	
	contracts/LSP7DigitalAsset/presets/LSP7MintableInitAbstract.sol	1		34	24	12	8	11	
	contracts/LSP7DigitalAsset/LSP7Errors.sol			23	23	13	2		
	contracts/LSP7DigitalAsset/LSP7DigitalAsset.sol	1		49	43	19	19	9	
	contracts/LSP7DigitalAsset/LSP7DigitalAssetCore.sol	1		368	316	148	114	106	
	contracts/LSP7DigitalAsset/extensions/LSP7CompatibilityForERC20InitAbstract.sol	1		66	39	27	3	14	
	contracts/LSP7DigitalAsset/extensions/LSP7CompatibilityForERC20.sol	1		68	45	26	10	13	
	contracts/LSP7DigitalAsset/extensions/LSP7CompatibilityForERC20Core.sol	1		107	78	39	23	33	
	contracts/LSP7DigitalAsset/extensions/LSP7CappedSupply.sol	1		46	41	15	20	8	

	contracts/LSP7DigitalAsset/extensions/LSP7CappedSupplyInitAbstract.sol	1		42	37	15	16	9	
	contracts/LSP7DigitalAsset/extensions/LSP7CappedSupplyInit.sol	1		19	19	7	9	7	
	contracts/LSP7DigitalAsset/extensions/ILSP7CompatibilityForERC20.sol		1	63	35	6	41	11	
	contracts/LSP7DigitalAsset/extensions/ILSP7CappedSupply.sol		1	17	16	4	9	5	
	contracts/LSP7DigitalAsset/extensions/LSP7CappedSupplyCore.sol	1		57	52	17	23	13	
	contracts/LSP7DigitalAsset/extensions/LSP7CompatibilityForERC20Init.sol	1		22	18	7	8	7	
	contracts/LSP7DigitalAsset/LSP7DigitalAssetInit.sol	1		28	23	7	14	7	
	contracts/LSP7DigitalAsset/LSP7DigitalAssetInitAbstract.sol	1		46	35	18	12	10	
	contracts/LSP7DigitalAsset/LSP7Constants.sol			13	13	4	5		
	contracts/LSP7DigitalAsset/ILSP7DigitalAsset.sol		1	181	83	39	124	21	
	contracts/LSP8IdentifiableDigitalAsset/LSP8IdentifiableDigitalAssetInit.sol	1		26	22	7	13	7	
	contracts/LSP8IdentifiableDigitalAsset/presets/LSP8MintableCore.sol	1		26	21	8	9	8	

📝	contracts/LSP8IdentifiableDigitalAsset/presets/LSP8Mintable.sol	1		37	32	13	15	10	
🔍	contracts/LSP8IdentifiableDigitalAsset/presets/ILSP8Mintable.sol		1	32	26	4	19	5	
📝	contracts/LSP8IdentifiableDigitalAsset/presets/LSP8MintableInit.sol	1		25	21	7	11	7	
🎨	contracts/LSP8IdentifiableDigitalAsset/presets/LSP8MintableInitAbstract.sol	1		35	26	14	8	11	
🔍	contracts/LSP8IdentifiableDigitalAsset/ILSP8IdentifiableDigitalAsset.sol		1	212	88	47	147	25	
	contracts/LSP8IdentifiableDigitalAsset/LSP8Constants.sol			21	21	6	8		
🎨	contracts/LSP8IdentifiableDigitalAsset/extensions/LSP8CappedSupplyInitAbstract.sol	1		46	41	18	17	9	
🎨	contracts/LSP8IdentifiableDigitalAsset/extensions/LSP8CompatibilityForERC721Core.sol	1		187	153	75	47	67	🔺
🎨	contracts/LSP8IdentifiableDigitalAsset/extensions/LSP8CappedSupply.sol	1		47	42	15	21	8	
🎨	contracts/LSP8IdentifiableDigitalAsset/extensions/LSP8CappedSupplyInit.sol	1		19	19	7	9	7	
🔍	contracts/LSP8IdentifiableDigitalAsset/extensions/ILSP8CappedSupply.sol		1	17	16	4	9	5	
📝	contracts/LSP8IdentifiableDigitalAsset/extensions/LSP8CompatibilityForERC721.sol	1		100	71	39	19	16	

🎨	contracts/LSP8IdentifiableDigitalAsset/extensions/LSP8CappedSupplyCore.sol	1		58	53	17	24	13	
📝	contracts/LSP8IdentifiableDigitalAsset/extensions/LSP8CompatibilityForERC721InitAbstract.sol	1		89	56	35	10	17	
📝	contracts/LSP8IdentifiableDigitalAsset/extensions/LSP8CompatibilityForERC721Init.sol	1		25	21	7	11	7	
	contracts/LSP8IdentifiableDigitalAsset/extensions/LSP8CompatibilityConstants.sol			6	6	3	2		
🔍	contracts/LSP8IdentifiableDigitalAsset/extensions/ILSP8CompatibilityForERC721.sol		1	103	36	6	64	19	
🎨	contracts/LSP8IdentifiableDigitalAsset/LSP8IdentifiableDigitalAssetInitAbstract.sol	1		44	34	17	12	10	
	contracts/LSP8IdentifiableDigitalAsset/LSP8Errors.sol			22	22	10	2		
📝	contracts/LSP8IdentifiableDigitalAsset/LSP8IdentifiableDigitalAsset.sol	1		48	42	19	19	9	
🎨	contracts/LSP8IdentifiableDigitalAsset/LSP8IdentifiableDigitalAssetCore.sol	1		437	378	183	121	142	📤
🔍	contracts/LSP4DigitalAssetMetadata/ILSP4Compatibility.sol		1	23	16	4	13	7	
	contracts/LSP4DigitalAssetMetadata/LSP4Constants.sol			25	25	8	9		
🎨	contracts/LSP4DigitalAssetMetadata/LSP4DigitalAssetMetadata.sol	1		39	39	19	15	12	

🎨	contracts/LSP4DigitalAssetMetadata/LSP4Compatibility.sol	1		38	38	14	18	13	
🎨	contracts/LSP4DigitalAssetMetadata/LSP4DigitalAssetMetadataInitAbstract.sol	1		35	31	16	9	13	
🎨	contracts/LSP4DigitalAssetMetadata/LSP4DigitalAssetMetadataInit.sol	1		26	22	7	13	7	
🎨	contracts/LSP0ERC725Account/LSP0ERC725AccountInitAbstract.sol	1		18	18	9	7	7	
📝	contracts/LSP0ERC725Account/LSP0ERC725AccountInit.sol	1		20	20	7	11	7	
🎨	contracts/LSP0ERC725Account/LSP0ERC725AccountCore.sol	1		163	139	66	58	49	💰
📝	contracts/LSP0ERC725Account/LSP0ERC725Account.sol	1		21	21	8	11	4	
	contracts/LSP0ERC725Account/LSP0Constants.sol			10	10	5	3		
📝	contracts/Factories/UniversalFactory.sol	1		155	135	64	58	51	💰🧱
📝	contracts/Factories/Create2Factory.sol	1		92	80	24	46	38	🖥️💰🧱🌀
	contracts/LSP3UniversalProfile/LSP3Constants.sol			11	11	4	4		
🎨	contracts/UniversalProfileInitAbstract.sol	1		27	27	13	10	6	
📚	contracts/Utils/UtilsLib.sol	1		59	47	28	15	77	🖥️

	contracts/Legacy/UniversalReceiverAddressStore.sol	1		60	55	34	8	26	
	contracts/Legacy/Registries/AddressRegistryRequiresERC725.sol	1		33	33	23	4	16	
	contracts/Legacy/Registries/AddressRegistry.sol	1		43	43	29	3	28	
	contracts/LSP6KeyManager/LSP6KeyManagerCore.sol	1		659	602	319	161	278	
	contracts/LSP6KeyManager/LSP6Errors.sol			37	37	6	26		
	contracts/LSP6KeyManager/LSP6Utils.sol	1		111	83	60	11	20	
	contracts/LSP6KeyManager/ILSP6KeyManager.sol		1	55	25	7	34	17	
	contracts/LSP6KeyManager/LSP6KeyManagerInitAbstract.sol	1		18	18	9	7	6	
	contracts/LSP6KeyManager/LSP6KeyManager.sol	1		20	20	7	11	3	
	contracts/LSP6KeyManager/LSP6Constants.sol			65	65	27	27	4	
	contracts/LSP6KeyManager/LSP6KeyManagerInit.sol	1		20	20	7	11	7	
	contracts/UniversalProfileInit.sol	1		20	20	7	11	7	
	contracts/LSP1UniversalReceiver/LSP1UniversalReceiverDelegateVault/LSP1UniversalReceiverDelegateVault.sol	1		57	52	26	21	14	
	contracts/LSP1UniversalReceiver/LSP1UniversalReceiverDelegateVault/Handling/TokenHandling.sol	1		74	74	48	12	27	

	contracts/LSP1UniversalReceiver/LSP1Utils.sol	1		48	39	32	5	5	
	contracts/LSP1UniversalReceiver/ILSP1UniversalReceiver.sol		1	40	36	10	24	6	
	contracts/LSP1UniversalReceiver/ILSP1UniversalReceiverDelegate.sol		1	24	18	3	14	3	
	contracts/LSP1UniversalReceiver/LSP1UniversalReceiverDelegateUP/LSP1UniversalReceiverDelegateUP.sol	1		69	64	29	29	14	
	contracts/LSP1UniversalReceiver/LSP1UniversalReceiverDelegateUP/Handling/TokenAndVaultHandling.sol	1		85	82	54	14	29	
	contracts/LSP1UniversalReceiver/LSP1Constants.sol			11	11	4	4		
	contracts/LSP9Vault/LSP9VaultCore.sol	1		194	172	82	65	72	
	contracts/LSP9Vault/LSP9Vault.sol	1		34	34	14	14	6	
	contracts/LSP9Vault/LSP9VaultInitAbstract.sol	1		30	30	14	10	9	
	contracts/LSP9Vault/LSP9Constants.sol			21	21	6	8		
	contracts/LSP9Vault/LSP9VaultInit.sol	1		20	20	7	11	7	
	contracts/UniversalProfile.sol	1		28	28	11	14	5	
	contracts/LSP5ReceivedAssets/LSP5Constants.sol			10	10	3	4		
	contracts/LSP5ReceivedAssets/LSP5Utils.sol	1		205	179	101	49	104	

	contracts/LSP10ReceivedVaults/LSP10Constants.sol			10	10	3	4		
	Totals	**100**	**13**	**74 43**	**62 03**	**31 02**	**231 6**	**2384**	

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, ...)

6. Scope of Work

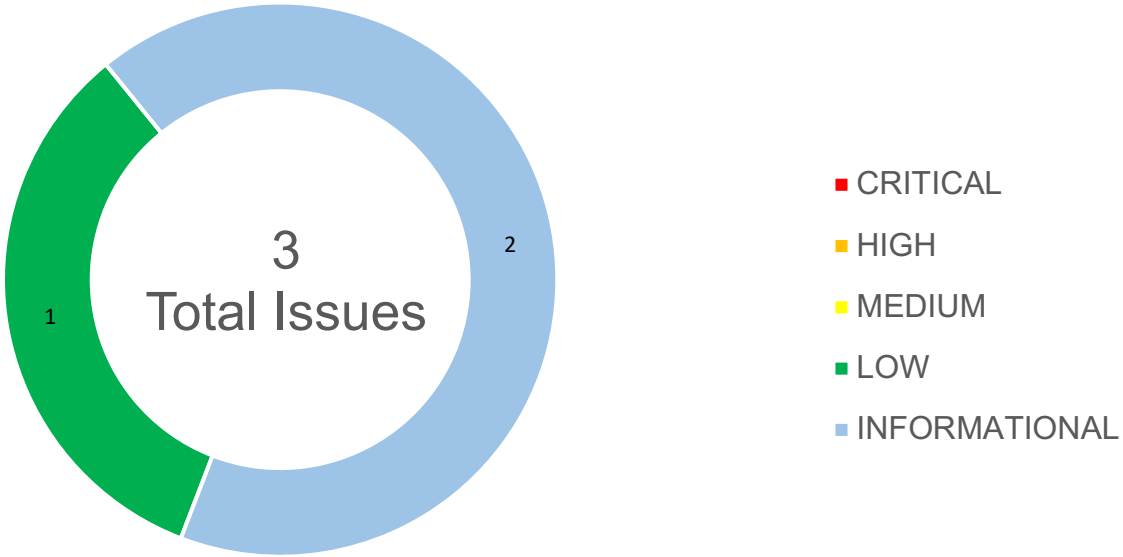
The LUKSO Team provided us with the files that needs to be tested. The scope of the audit are the LUKSO Standard Proposal (LSP) contracts.

The security assessment pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors
- Assessing the codebase to ensure compliance with current best practices and industry standards
- Ensuring contract logic meets the specifications and intentions of the client
- Testing of provided unit tests and the coverage
- Enhance general coding practices for better structures of source codes
- Line-by-line manual and automated review of the entire codebase

The security assessment will result in findings that range from critical to informational. We recommend addressing these findings to ensure a high level of security standards and readability.

6.1 Findings Overview



No	Title	Severity	Status
6.2.1	Misleading Key Hash Value	LOW	OPEN
6.2.2	Floating Pragma Version Identified	INFORMATIONAL	OPEN
6.2.3	Unused Code	INFORMATIONAL	OPEN

6.2 Manual and Automated Vulnerability Test

CRITICAL ISSUES
During the audit, Chainsulting‘s experts found **0 Critical issues** in the code of the smart contract.

HIGH ISSUES
During the audit, Chainsulting’s experts found **0 High issues** in the code of the smart contract.

MEDIUM ISSUES
During the audit, Chainsulting’s experts found **0 Medium issues** in the code of the smart contract.

LOW ISSUES
During the audit, Chainsulting’s experts found **1 Low issue** in the code of the smart contract.

6.2.1 Misleading Key Hash Value
Severity: LOW
Status: OPEN
Code: NA
File(s) affected: LSP9Vault/LSP9Constants.sol, LSP4DigitalAssetMetadata/LSP4Constants.sol, LSP3UniversalProfile/LSP3Constants.sol

Attack/Description	The current implementation defines a hash key value to indicate the supported standard. The first 10 bytes are not representing bytes10(keccak256('SupportedStandard')) as defined in the comments. This can lead to false usage of supported standard key.

Code	Line 4 – 5 (LSP9Vault/LSP9Constants.sol)
	`// bytes10(keccak256('SupportedStandard')) + bytes2(0) + bytes20(keccak256('LSP9Vault'))`
	`bytes32 constant _LSP9_SUPPORTED_STANDARDS_KEY = 0xeafec4d89fa9619884b600007c0334a14085fefa8b51ae5a40895018882bdb90;`
	
	Line 6 – 7 (LSP4DigitalAssetMetadata/LSP4Constants.sol)
	`// bytes10(keccak256('SupportedStandard')) + bytes2(0) + bytes20(keccak256('LSP4DigitalAsset'))`
	`bytes32 constant _LSP4_SUPPORTED_STANDARDS_KEY = 0xeafec4d89fa9619884b60000a4d96624a38f7ac2d8d9a604ecf07c12c77e480c;`
	
	Line 4 – 5 (LSP3UniversalProfile/LSP3Constants.sol)
	`// bytes10(keccak256('SupportedStandard')) + bytes2(0) + bytes20(keccak256('LSP3UniversalProfile'))`
	`bytes32 constant _LSP3_SUPPORTED_STANDARDS_KEY = 0xeafec4d89fa9619884b60000abe425d64acd861a49b8ddf5c0b6962110481f38;`

Result/Recommendation	It is recommended to change the hash value or its description to match each other and avoid different keys for the same field.

INFORMATIONAL ISSUES

During the audit, Chainsulting's experts found **2 Informational issues** in the code of the smart contract.

6.2.2 Floating Pragma Version Identified
Severity: INFORMATIONAL
Status: OPEN

Code: SWC-103
File(s) affected: ALL

Attack / Description	It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.
Code	Line 1 `^0.8.0`
Result/Recommendation	It is recommended to follow the latter example, as future compiler versions may handle certain language constructions in a way the developer did not foresee. It is advised that floating pragma should not be used in production. Both truffle-config.js and hardhat.config.js support locking the pragma version. i.e. Pragma solidity 0.8.0

6.2.3 Unused Code

Severity: INFORMATIONAL
Status: OPEN
Code: CWE-1164
File(s) affected: Helpers/UniversalReceiverDelegateRevert.sol

Attack / Description	The helper was probability introduced to test smart contracts functionality. However, it is not used in the current implementation.
Code	NA
Result/Recommendation	It is recommended to remove all unused code to keep the projects clean and well arranged.

6.3 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	X
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	✅

6.4 Test Deployment

Universal Profiles

6.4.1 LSP0 – ERC725 Account

1. Summary

This standard heavily relies on the ERC725 standard and represents a blockchain based account. The ERC725 standard describes a generic key-value store where arbitrary data can be stored in a bytes32 to bytes mapping (ERC725Y). In addition, there is an execute function which allows to execute arbitrary code by the contract owner (ERC725X). This contract owner can be an EOA or another contract like a Key Manager (LSP6). This makes the account universal accessible restricted by defined permissions. A LSP0 account also implements several other standards like the Universal Receiver (LSP1), signature verification (ERC1271) and interface detection (ERC165).

2. Deployment

Transaction:
https://blockscout.com/lukso/l14/tx/0x130d29e990d6f8aaf22cf76f1fc9e3f57ee4417cd2b5e7ab0b12b027f5d5c14f

Contract:
https://blockscout.com/lukso/l14/address/0x07D67d5C90a0Dc736A9579458F7ec9f128CAdE8B

6.4.2 LSP1 – Universal Receiver

1. Summary

This standard is introduced to be able to notify contracts on receiving any kind of assets. In the past developers had to implement multiple functions for receiving assets because there are different standards for different kind of assets, like ERC20 and ERC721.
The LSP1 solves this by defining a single standardized function called *universalReceiver(…).* Contract implementing this standard can receive arbitrary data by calling the function with a given typId. This gives contracts the ability to deny receiving assets, store data related to received assets or execute some custom logic.
An extension for the Universal Receiver is the Universal Receiver Delegate which allows delegating functionality to another contract. Afterwards the other contract can execute custom logic with received data.

2. Deployment
Transaction:
https://explorer.execution.l16.lukso.network/tx/0x1336fcb3f853cf969ef3224ef1ad103d03348726a78ef3d69f56c9dc9e40a103

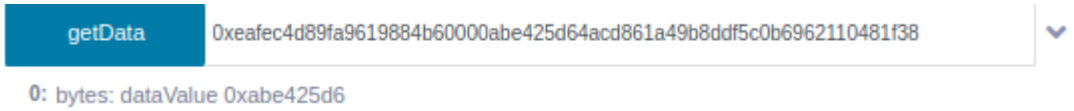
Contract:
https://explorer.execution.l16.lukso.network/address/0x0261738838348029fCb07101180aA4BE8e91Fbb2

2.1. Emitting UniversalReceiver event on receiving token
https://explorer.execution.l16.lukso.network/tx/0xb793529b305d09049ec4cdc3a8f7b219109bc4cb42e62537a63853794076e3ae/logs

6.4.3 LSP3 – Universal Profile Metadata

1. Summary
A LSP0 account itself has no definitions of how to store data. It has a generic key-value store thanks to the ERC725 standard, but the LSP0 does not define how data is stored. Here comes LSP3 – Universal Profile Metadata into play. The standard describes data keys and structures to store data in a LSP0 account to become a universal profile.
One key is set for determining that the in implementing contract is a universal profile. The second key stores a reference to a JSON file. The standard defines how the JSON file should be structured. This standardization enables a proper decoding of data after storing it in the given format because everyone knows how the data should be interpreted.

2. Deployment
The Metadata standard can be seen at a Universal Profile.
https://explorer.execution.l16.lukso.network/address/0x3E85651B56Bb35D4afceD3699e363FD24Bc43411/read-contract



6.4.4 LSP5 – Received Assets

1. Summary

This metadata standard is introduced to define the storage data representing received assets. If a contract implements the LSP1 – Universal Receiver, it may store information about different received assets in a single contract. Therefore, this standard defines the way the data should be stored.
The LSP5 defines two data key type: one for storing a list of all tokens owned by the implementing contract and one for a map with an identifier for the asset standard and the index in the given list. In this way it is possible to efficiently store a list of owned assets and quickly querying for the type of asset like ERC20 or NFTs.

2. Deployment

The Metadata standard can be seen at a Universal Profile with an Universal Receiver Delegate on receiving tokens.
https://explorer.execution.l16.lukso.network/tx/0xb9463dad93dddcef9c3723b20e4de6f0ba6df59d88403e402b94e003865c3556/logs

a) sets LSP5ReceivedAssets[]

0x3E85651B56Bb35D4afceD3699e363FD24Bc43411

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Name	Type	Indexed?	Data
dataKey	bytes32	true	0x6460ee3c0aac563ccbf76d6e1d07bada78e3a9514e6382b736ed3f478ab7b90b

b) sets LSP5ReceivedAssetsMap

0x3E85651B56Bb35D4afceD3699e363FD24Bc43411

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Name	Type	Indexed?	Data
dataKey	bytes32	true	0x6460ee3c0aac563ccbf76d6e1d07bada0000000000000000000000000000000000

c) sets token address in LSP5ReceivedAssetsMap

0x3E85651B56Bb35D4afceD3699e363FD24Bc43411

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Name	Type	Indexed?	Data
dataKey	bytes32	true	0x812c4334633eb816c80d0000c6d6073251bd3486e833c12058bbf9aefd8d1df7

6.4.5 LSP6 – Key Manager

1. Summary

This standard defines a Key Manager to manage access to a Universal Profile (UP) for different addresses. These addresses can belong to external owned accounts (EOA) or other smart contracts. The LSP6 offers the opportunity to give any address selected permissions. The Key Manager offers a wide range of different permissions, which can be set stand alone or combined. There are ten different permission types: change owner, change permissions, add permissions, set data, call, static call, delegate call, deploy, transfer value and sign.
Information about which address has permissions for which actions is stored in the targeted Universal Profile's ERC725Y storage. Only the Key Manager is allowed to call functions of the UP and thus users have to call the Key Manager to interact with the UP. The Key Manager fetches permission information from a related UP and checks if the caller is allowed to execute given call data. The call will only be forwarded to an UP if the caller is authorized by Key Manager.
Permissions can be restricted to calls for specific addresses, functions, standards or ERC725Y data keys.

2. Deployment

To demonstrate the functionality of the Key Manager we deploy a Universal Profile and a Key Manager, which is managing the access to the Universal Profile. In the following the access management to Alice's UP will be shown.

2.1. Set up Universal Profile with Key Manager
2.1.1. Deploy Alice's Universal Profile
https://explorer.execution.l16.lukso.network/address/0x3B59009049552E2C1D973c04600b6ad144829556

2.1.2. Deploy Key Manager of Alice's UP
https://explorer.execution.l16.lukso.network/address/0xAee594290A1DBC675c05ca255Ed7276C28b35C8a

2.1.3. Set permission for Alice (all permissions) and Universal Receiver Delegate (set data)
https://explorer.execution.l16.lukso.network/tx/0xd9e75e76a92fd70421db7488da591d2c934f29f7225d63840355c0adcc37fcb6/logs

2.1.4. Successfully mint a LSP7 token with Alice

https://explorer.execution.l16.lukso.network/tx/0xb793529b305d09049ec4cdc3a8f7b219109bc4cb42e62537a63853794076e3ae/logs

2.2. Permission to call specific function on specific address
In this case we want to allow Bob to call the transfer function of a single specified LSP7 token.

2.2.1. Reverts on directly calling any UP's function
https://explorer.execution.l16.lukso.network/tx/0x124c9a4166726e7ac9c8e4511f85a0c54e76193875fb6fb4847e8f0b86579d8e

2.2.2. Reverts on calling execute function through Key Manager
https://explorer.execution.l16.lukso.network/tx/0x349fd86bb698eaf83cdbaaaa7faeccf0c3c594b0097f611bd1ea1bc73df2f945

2.2.3. Alice sets permissions for Bob to only call transfer function of specified token
https://explorer.execution.l16.lukso.network/tx/0xd166a2bb4c9c7376420e65bff421192ea4a8add9178eb3ad4eba1d02f6c92fce

a) Permission type (CALL) set in ERC725Y storage of UP

0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Name	Type	Indexed?	Data
dataKey	bytes32	true	0x4b80742de2bf82acb363000044ea081195bb1141a7cc07a13d5e7a74c8c4f458

b) Specific target address allowed to be called by Bob set in ERC725Y storage of UP

0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Name	Type	Indexed?	Data
dataKey	bytes32	true	0x4b80742de2bfc6dd6b3c000044ea081195bb1141a7cc07a13d5e7a74c8c4f458

c) Specific function (transfer) allowed to be called by Bob set in ERC725Y storage of UP

0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Name	Type	Indexed?	Data
dataKey	bytes32	true	0x4b80742de2bf8efea1e8000044ea081195bb1141a7cc07a13d5e7a74c8c4f458

2.3.4. Bob transfers Alice's tokens by calling tokens transfer function through Alice's UP with Key Manager
https://explorer.execution.l16.lukso.network/tx/0x80d963e6cbd3567fa3ad973b9cf7bc63dc5240a0b5e63465ebf497a87184148b

a) Universal receiver of sender (Alice's UP) triggered after transfer

0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0x9c3ba68e
Call	UniversalReceiver(address indexed from, uint256 value, bytes32 indexed typeId, bytes indexed returnedValue, bytes receivedData)

Name	Type	Indexed?	Data
from	address	true	0xc6d6073251bd3486e833c12058bbf9aefd8d1df7

b) Universal receiver of recipient (Bob UP) triggered after transfer

0x3E85651B56Bb35D4afceD3699e363FD24Bc43411

Method Id	0x9c3ba68e
Call	UniversalReceiver(address indexed from, uint256 value, bytes32 indexed typeId, bytes indexed returnedValue, bytes receivedData)

Name	Type	Indexed?	Data
from	address	true	0xc6d6073251bd3486e833c12058bbf9aefd8d1df7

2.3.5. Still reverts on calling other functions of specified token (only transfer function is permitted). Calling mint function reverts:
https://explorer.execution.l16.lukso.network/tx/0xbc239f0a6a1853c082a76835ef546d5c8668b00e7062b23a0fda8a043a2bc739

2.4. Different Permission types for interacting with UP
Previously we permitted Bob to call the transfer function of a specific LSP7 Token with the CALL permission. Now we want Bob to be able to change ERC725Y storage of Alice's profile

2.4.1. Reverts on Bob trying to set Name data in Alice's UP
https://explorer.execution.l16.lukso.network/tx/0xf02ab1d008407520a32f44af95e7f03e50679ef0aba59370138f757530a51469

2.4.2. Giving Bob permission for setting data field Name in Alice's UP
https://explorer.execution.l16.lukso.network/tx/0xd6f33cd549ac4b71d3440b4238fb41cfffc656ada93a97ebd0dd926ee9720387

a) Permission type (CALL) set in ERC725Y storage of Alice's UP

0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Name	Type	Indexed?	Data
dataKey	bytes32	true	0x4b80742de2bf82acb363000044ea081195bb1141a7cc07a13d5e7a74c8c4f458

b) Allowed ERC725Y-datafield set in ERC725Y storage of Alice's UP

0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Name	Type	Indexed?	Data
dataKey	bytes32	true	0x4b80742de2bf90b8b485000044ea081195bb1141a7cc07a13d5e7a74c8c4f458

2.4.3. Bob successfully sets Name field in Alice's UP
https://explorer.execution.l16.lukso.network/tx/0xbbd97401f6ba656cb288b9ae65ef6ea68701eada1f431a859f8fd2e334500490

0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Name	Type	Indexed?	Data
dataKey	bytes32	true	0x55e49609591f684ecf6f2909c9e20c2439b990887b9c3fe108b154c9077d85cf

2.4.4. Reverting if Bob tries to set other fields in Alice's UP
https://explorer.execution.l16.lukso.network/tx/0xf6f167c0bff41d61d62f7349163f54eeaa52b199fd160792977ddbaa1275d27b

6.4.6 LSP9 – Vault

1. Summary

This standard is introduced to define a isolated contract where funds can be stored by a user. The general idea of Universal Profiles is, that funds can be stored safely in a user's UP. The access to this UP should be managed with a LSP6 – Key Manager. If a user (Alice) wants to allow access to specific funds for another user (Bob), she has to give the user permission to access the execute function. This would give Bob full execution control and could drain out any funds from the profile. A more secure way is the LSP9 vault. Alice can store any assets in this vault contract and manage access to only this contract address through his UP. Subsequently, Bob is only allowed to call functions on the vault contract and the funds in Alice's UP are safe and cannot be touched by Bob.

2. Deployment

To simulate the functionality of the LSP9 vault we deploy the Vault contract, the Universal Receiver Delegate Vault (URD) and set different permission in the vault ERC725Y storage for interaction.

2.1. Deploy Vault contract with EOA
Transaction:
https://explorer.execution.l16.lukso.network/tx/0xa2674370c03fed82503924670fa37de9a4ba0d7afd3cd7a80a3433798e218bd3

Contract:
https://explorer.execution.l16.lukso.network/address/0x1af8e15f0eE6a32467aC7674DC53C5e700DCb63B

2.2. Set permissions for UP and URD
https://explorer.execution.l16.lukso.network/tx/0x6c585d620a042e0a14ba748b3e9d5922f61d19a2fba229e6172e57b6d702cba6

2.3. Transfer ownership to UP
https://explorer.execution.l16.lukso.network/tx/0x2304d658e493d9f39ad6c012cdd398edde3ced8bd951cc148185a58a0c65c6b6/logs
a) Ownership transferred and new owner set in vault

0x1af8e15f0eE6a32467aC7674DC53C5e700DCb63B

Method Id	0x8be0079c
Call	OwnershipTransferred(address indexed previousOwner, address indexed newOwner)

Name	Type	Indexed?	Data
previousOwner	address	true	0x1966bf71868dc9d210cf8b1b8e9f8ff9b6d425f8
newOwner	address	true	0x3b59009049552e2c1d973c04600b6ad144829556

b) LSP10 Received vaults stored in UP

0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Name	Type	Indexed?	Data
dataKey	bytes32	true	0x192448c3c0f88c7f238c00001af8e15f0ee6a32467ac7674dc53c5e700dcb63b

2.4. Transferring token from UP to the vault
https://explorer.execution.l16.lukso.network/tx/0x789c118f316f7c2660c0edf0a89425c17a70a6e813265d615b04fb4399fa87f0

2.5. Failing accessing vault funds with unauthorized user

a) Calling Vault's execute function directly reverts
https://explorer.execution.l16.lukso.network/tx/0x1898529fd1ccd0556fb23780057b471b939a0b6a7f59d333697112ce946be01a

b) Calling Vault' execute function through UP with Key Manager reverts
https://explorer.execution.l16.lukso.network/tx/0xfd94485c0cbea1618e99bdfee7fedc371515928f7bb92b6a7c537316a5774e7e

2.6. Set permissions in UP for other user calling Vault's functions
https://explorer.execution.l16.lukso.network/tx/0x7364f9cc54deab8118a49ba03c8c456c53f6726a633c7068d870cb0c1230fb8b

0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Name	Type	Indexed?	Data
dataKey	bytes32	true	0x4b80742de2bf82acb363000044ea081195bb1141a7cc07a13d5e7a74c8c4f458

2.7. Other user calling Vault's execute function through the UP with Key Manager and transfers token to his UP
https://explorer.execution.l16.lukso.network/tx/0x271f973284e1950a2341efd995abf3ad1f5d00247f1887dd86e6fe7526bc02a1

0x3E85651B56Bb35D4afceD3699e363FD24Bc43411

Method Id	0x9c3ba68e
Call	UniversalReceiver(address indexed from, uint256 value, bytes32 indexed typeId, bytes indexed returnedValue, bytes receivedData)

Name	Type	Indexed?	Data
from	address	true	0xc6d6073251bd3486e833c12058bbf9aefd8d1df7

6.4.7 LSP10 – Received Vaults

1. Summary

This metadata standard is introduced to define the storage data representing received vaults. If a contract implements the LSP1 – Universal Receiver, it may store information about different received vaults in its contract. Therefore, this standard defines the way data should be stored.
The LSP10 defines two data key type: one for storing a list of all vaults owned by the implementing contract and one for a map with an identifier for the vault standard and the index in the given list. In this way it is possible to efficiently store a list of owned assets and quickly querying for the type of vault.

2. Deployment

The LSP10 metadata standard can be seen in action by having a Universal Profile with an Universal Receiver Delegate which is owning a Vault. The information of the Vault will be stored in the UP according to the LSP10.

Transaction UP receiving Vault
https://explorer.execution.l16.lukso.network/tx/0x2304d658e493d9f39ad6c012cdd398edde3ced8bd951cc148185a58a0c65c6b6/logs

a) Sets LSP10Vaults[] in UP's ERC725Y storage

0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Name	Type	Indexed?	Data
dataKey	bytes32	true	0x55482936e01da86729a45d2b87a6b1d3bc582bea0ec00e38bdb340e3af6f9f06

b) Sets LSP10VaultMap in UP's ERC725 storage

0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Name	Type	Indexed?	Data
dataKey	bytes32	true	0x55482936e01da86729a45d2b87a6b1d3000000000000000000000000000000000

c) Sets Vault's address in LSP10VaultMap of UP's ERC725 storage

0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Name	Type	Indexed?	Data
dataKey	bytes32	true	0x192448c3c0f88c7f238c00001af8e15f0ee6a32467ac7674dc53c5e700dcb63b

Use Case Universal Profiles
In the following use case are for two users Universal Profiles (UP), Key Manager (KM) and Universal Receiver Delegate (URD) deployed. Users will interact through the KM and UP with LSP7 token contracts.

1. Set up Universal Profile for Alice
EOA: 0x1966Bf71868dC9D210cf8b1b8e9F8Ff9B6d425f8
Profile: 0x3B59009049552E2C1D973c04600b6ad144829556
KeyManager: 0xAee594290A1DBC675c05ca255Ed7276C28b35C8a
URD: 0x0261738838348029fCb07101180aA4BE8e91Fbb2

1.1. Deployment
Deploys the three-core contract for Alice's account.

1.1.1. Deploy Universal Profile
https://explorer.execution.l16.lukso.network/address/0x3B59009049552E2C1D973c04600b6ad144829556

1.1.2. Deploy Key Manager
https://explorer.execution.l16.lukso.network/address/0xAee594290A1DBC675c05ca255Ed7276C28b35C8a

1.1.3. Deploy Universal Receiver Delegate for Universal Profiles
https://explorer.execution.l16.lukso.network/address/0x0261738838348029fCb07101180aA4BE8e91Fbb2

1.2. Set up permissions and ownership
To allow URD and EOA interacting with the UP it is required to set permissions in the ERC725Y storage of the Universal Profile.

1.2.1. Set Permissions for URD (set data permission) as well as for Key Manager (all permissions) and URD reference:
https://explorer.execution.l16.lukso.network/tx/0xd9e75e76a92fd70421db7488da591d2c934f29f7225d63840355c0adcc37fcb6/logs

1.2.2. Transfer ownership to Key Manager
a) set new pending owner with EOA
https://explorer.execution.l16.lukso.network/tx/0xae700e6274829ec48cc642d0750a618965f7cc41a99cd8e328fbe0eeccb530f5

b) claim ownership with Key Manager
https://explorer.execution.l16.lukso.network/tx/0x9ee251a200e5b7de712e2553c6cf872502c82d6c7e892e674708d192889f20cf

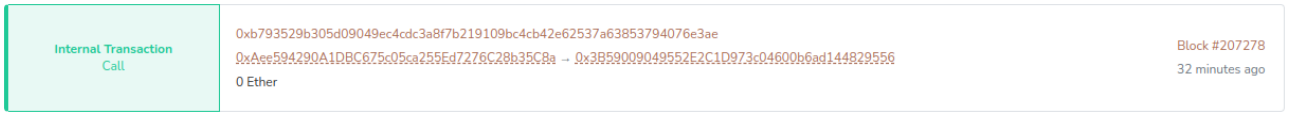
1.3. Minting LSP7 Token to Alice's UP by calling mint function with use of Key Manager and UP
https://explorer.execution.l16.lukso.network/tx/0xb793529b305d09049ec4cdc3a8f7b219109bc4cb42e62537a63853794076e3ae/logs

a) Calling Key Manager execute function with authorized account

execute(bytes _calldata):

Method Id	0x09c5eabe
Call	execute(bytes _calldata)

b) Internal call to UP's execute function



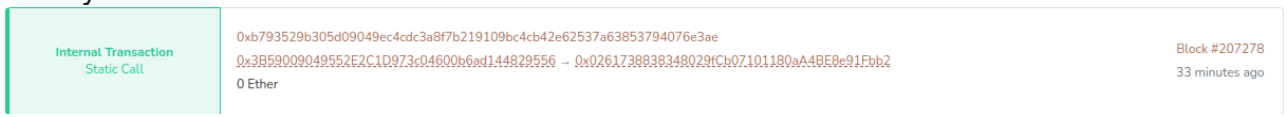
c) Internal call to LSP7 token mint funtion

d) Transfer on minting triggers UniversalReceiver event in UP

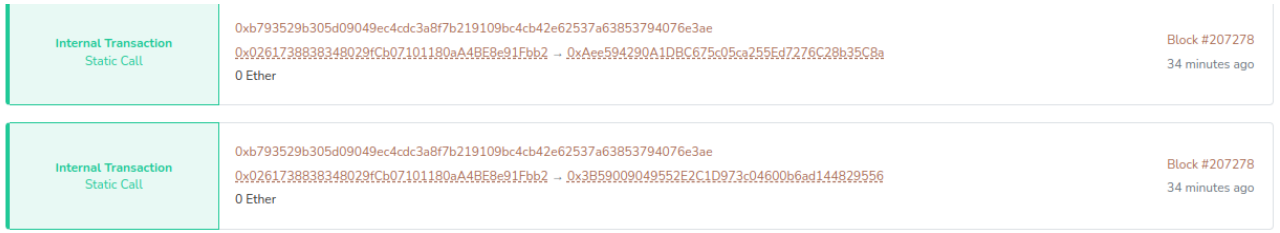
0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0x9c3ba68e
Call	UniversalReceiver(address indexed from, uint256 value, bytes32 indexed typeId, bytes indexed returnedValue, bytes receivedData)

e) UP delegates functionality to URD



d) URD sets LSP5 Asset values in UP



0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Name	Type	Indexed?	Data
dataKey	bytes32	true	0x812c4334633eb816c80d0000c6d6073251bd3486e833c12058bbf9aefd8d1df7

2. Set up Universal Profile for Bob
EOA: 0x44Ea081195bb1141A7CC07a13D5E7A74c8C4F458
Profile: 0x3E85651B56Bb35D4afceD3699e363FD24Bc43411
KeyManager: 0xEb23431630bdfdd1101943656e0BC2c7B7b832eC
URD: 0x128fDe909D7F863ceBa88bEdabF4e8d7c41E3665

2.1. Deployment
Deploys the three core contract for Bob's account
2.1..1. Deploy Universal Profile
https://explorer.execution.l16.lukso.network/address/0x3E85651B56Bb35D4afceD3699e363FD24Bc43411

2.1.2. Deploy Key Manager
https://explorer.execution.l16.lukso.network/address/0xEb23431630bdfdd1101943656e0BC2c7B7b832eC

2.1.3. Deploy Universal Receiver Delegate for Universal Profiles
https://explorer.execution.l16.lukso.network/address/0x128fDe909D7F863ceBa88bEdabF4e8d7c41E3665

2.2. Set up permissions and ownership
2.2.1. Set Permissions for URD (set data permission) as well as for Key Manager (all permissions) and URD reference:
https://explorer.execution.l16.lukso.network/tx/0xf415c74daeb5c082b4be19be340e9c77619799cd991b60a6686d4331657b4193/logs

2.2.2. Transfer ownership to Key Manager
a) set new pending owner with EOA
https://explorer.execution.l16.lukso.network/tx/0x1e16853468ee9f29e657ca95180d5f5bf022b3919b483fe198f80b24dbc8787e

b) claim ownership with Key Manager
https://explorer.execution.l16.lukso.network/tx/0x3ea259f2d138119a199ad377ea96b7222ac331f95b971ea4ac092cf2580867ea

3. Transfer LSP7 Token from Alice UP to Bobs UP
https://explorer.execution.l16.lukso.network/tx/0xb9463dad93dddcef9c3723b20e4de6f0ba6df59d88403e402b94e003865c3556

a) Calling Key Manager execute function with authorized account

0xAee594290A1DBC675c05ca255Ed7276C28b35C8a

Method Id	0x6b934045
Call	Executed(uint256 indexed _value, bytes4 _selector)

b) Internal call to Alice's UP execute function

0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0x48108744
Call	Executed(uint256 indexed operation, address indexed to, uint256 indexed value, bytes4 selector)

c) Internal call to LSP7 token transfer function

a) Transfer triggers UniversalReceiver event on Alice's UP

0x3B59009049552E2C1D973c04600b6ad144829556

Method Id	0x9c3ba68e
Call	UniversalReceiver(address indexed from, uint256 value, bytes32 indexed typeId, bytes indexed returnedValue, bytes receivedData)

b) Triggers UniversalReceiver event on Bob's UP

0x3E85651B56Bb35D4afceD3699e363FD24Bc43411

Method Id	0x9c3ba68e
Call	UniversalReceiver(address indexed from, uint256 value, bytes32 indexed typeId, bytes indexed returnedValue, bytes receivedData)

c) URD sets LSP5 metadata in Bob's UP

0x3E85651B56Bb35D4afceD3699e363FD24Bc43411

Method Id	0xcdf4e344
Call	DataChanged(bytes32 indexed dataKey)

Token Standards

6.4.8 LSP4 – Digital Asset Metadata

1. Summary

The LSP4 is a metadata standard for all digital assets. It defines which data should be stored by them and in which format. Therefore, LSP4 introduces six key-value pairs, which may be attached to any digital asset's storage (ERC725Y). The first key defines if the given contract represents a digital asset, followed by keys for token name and symbol. Furthermore, there are keys for the token creator address or multiple token creator addresses as well as a key for token metadata. The metadata must be stored in a JSON file and should have the defined format to make en- and decoding easier.
The standard also holds an extension for external callers expecting ERC20 or ERC721 token standard by holding a *name()* and *symbol()* function.

2. Deployment

2.1. LSP7 with LSP4 metadata standard
https://blockscout.com/lukso/l14/address/0xAee594290A1DBC675c05ca255Ed7276C28b35C8a

Queried for token name (0xdeba1e292f8ba88238e10ab3c7f88bd4be4fac56cad5194b6ecceaf653468af1):
returns: *dataValue* *(bytes)* : 0x416c69616e20546f6b656e
converted to string: Alian Token

2.2. LSP8 with LSP4 metadata standard
https://blockscout.com/lukso/l14/address/0x1af8e15f0eE6a32467aC7674DC53C5e700DCb63B
Queried for token name (0xdeba1e292f8ba88238e10ab3c7f88bd4be4fac56cad5194b6ecceaf653468af1):
returns: *dataValue* *(bytes)* : 0x437573746f6d204e4654
converted to string: Custom NFT

6.4.9 LSP7 – Digital Asset (Token)

1. Summary

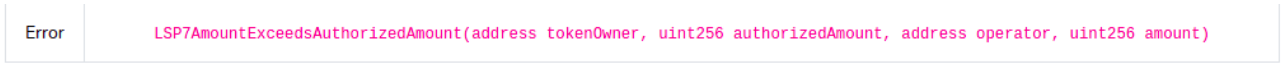
The LSP7 Digital Asset represents a fungible token. It is based on the ERC20 token standard and thus is similar to it. It allows the caller to transfer tokens from one address to another and manages token approval for an so called *operator*. In addition, it prevents sending funds to an address, where the funds may be lost. The *transfer()* function gets a force parameter as input which is checking if the receiving contract implements LSP1 - Universal Receiver. If this is not the case and the force parameter is set to false, the transfer reverts. This security parameter enables safer transfers between Universal Profiles and reduces risks of human errors.
The LSP7 standard can implement tokens divisible or non-divisible (like NFTs). This means it can only have none or 18 decimals. Also, it holds extensions to be compatible with ERC20 token standard and to deploy a token with a capped supply.

2. Deployment

Transaction: https://blockscout.com/lukso/l14/tx/0x823e490ec0cfc529f2de3e31d6016ca585639ea8b881e2696c70f858dd0844fc

Contract: https://blockscout.com/lukso/l14/address/0xAee594290A1DBC675c05ca255Ed7276C28b35C8a

2.1. Fails transferring tokens of other users without allowance:



https://blockscout.com/lukso/l14/tx/0xf077b20169d6e59ec74415ecfcbd5a03669ec1d8251fa90352cac0c0dec9e08e

2.2. Allowing operator to transfer tokens:
https://blockscout.com/lukso/l14/tx/0x9f03f859d65597997eccfb60c32f10d98b2c376b3c02aea8d309b4aba5b553a7

2.3. Transferring tokens of another user with allowance:
https://blockscout.com/lukso/l14/tx/0xce08c44086f0eff77d178e70c3b5e09b5a018dca67a744d93ed23670a6860b58

2.4.Transferring to address implementing LSP1 with force parameter set to false:
https://blockscout.com/lukso/l14/tx/0xcf8b4de094453e72f49aadeed50e971a119f9db393e5533f58cc2504720be973

2.5. Prevents transferring to address not implementing LSP1 with force parameter set to false:

Error	LSP7NotifyTokenReceiverIsEOA(address tokenReceiver)

https://blockscout.com/lukso/l14/tx/0xb9d3687ece7c3890c1ac1dd2f858b00a9d2c156f144aea99dbea18304cc773a1

2.6. Transferring to address not implementing LSP1 by setting force parameter set to true:
https://blockscout.com/lukso/l14/tx/0x57f58422aa1f9be05fec13de09a3fad8ba1e854a38aa998c686edf79f6a14d17

6.4.10 LSP8 – Identifiable Digital Asset (NFT)

1. Summary

The LSP8 defines a standard for Non Fungible Tokens. The standard relies on ERC725Y, which provides the ability to store unlimited arbitrary data. This makes the LSP8 more flexible than known standards like ERC721 because it is possible to store any metadata. In addition, the standard allows numbers, bytes32 hash values or even addresses as unique tokenId. Metadata for any single NFT can thus be stored in another ERC725Y storage contract and linked to the NFT contract by setting its address as tokenId. This makes every single NFT more customizable and enables more flexibility for creating NFTs.

2. Deployment

Transaction: https://blockscout.com/lukso/l14/tx/0xc1c4fe84faf75445c732beee8e27a33b3b4443058a5ca448dbaeedc20427ba0a

Contract:
https://blockscout.com/lukso/l14/address/0x1af8e15f0eE6a32467aC7674DC53C5e700DCb63B

2.1. Minting NFT with id 0:
https://blockscout.com/lukso/l14/tx/0x03eed5dfa107131339a32d96ffd117939dd721ee90b0148c868dfec12ae9c0c9

2.2. Fails minting NFT with same id again:

Error	LSP8TokenIdAlreadyMinted(bytes32 tokenId)

https://blockscout.com/lukso/l14/tx/0x4d5eb72c440f537b7fb7b2f0b3dc2647c90d1c6bb07f5829f1ab86cbeb487514

2.3. Transferring NFT to another user:

https://blockscout.com/lukso/l14/tx/0x19aadae363024e6481ab6028392792915d9ab8002e86f43dd8415bce223d90a9

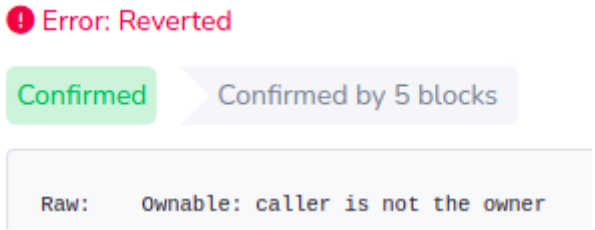
2.4. Transferring NFT of other user with allowance:
https://blockscout.com/lukso/l14/tx/0x24bac525f9e702c197fa7c01a80b334349aa6e4f8f95692779020c5cdc7dab60

2.5. Fails transferring NFT of other user without allowance:

Error	LSP8NotTokenOperator(bytes32 tokenId, address caller)

https://blockscout.com/lukso/l14/tx/0x888c28edef536ac54401d3a976938d2718b1f6f3a9009fc5253b81d5ebb95964

2.6. Fails minting NFT without creator rights:



https://blockscout.com/lukso/l14/tx/0x166eaa1f0d2859d162f913c270455296e7ba9b7fc0420cfed2297bc0369a1fd6

2.7. Fails transferring to address not implementing LSP1 with force parameter set to false:

Error	LSP8NotifyTokenReceiverIsEOA(address tokenReceiver)

https://blockscout.com/lukso/l14/tx/0x341be031fd4cac8ad011c9c9b16a43b9e2d6db1701b9d85cc1c394dbfb0f82ed

7. Executive Summary

Three (3) independent Chainsulting experts performed an unbiased and isolated audit of the smart contract codebase.

The main goal of the audit was to verify the LSP Standard Proposals (LSPs) regarding the security and functions of the smart contracts. During the audit, no critical, no high, no medium, 1 low and 2 informational issues have been found, after the manual and automated security testing.

No necessary need for action, as the recommendations only further enhance the code's readability, not security. The extensive documentation within the codebase and as docs (https://docs.lukso.tech/standards/introduction) have greatly benefiting the understanding of the usage and showed us a high professionalism of the LUKSO Team.

8. About the Auditor

Chainsulting is a professional software development firm, founded in 2017 and based in Germany. They show ways, opportunities, risks and offer comprehensive blockchain solutions. Some of their services include blockchain development, smart contract audits and consulting.

Chainsulting conducts code audits on market-leading blockchains such as Hyperledger, Tezos, Ethereum, Binance Smart Chain, and Solana to mitigate risk and instil trust and transparency into the vibrant crypto community. They have also reviewed and secure the smart contracts of 1Inch, POA Network, Unicrypt, Amun, Furucombo among numerous other top DeFi projects.

Chainsulting currently secures $100 billion in user funds locked in multiple DeFi protocols. The team behind the leading audit firm relies on their robust technical know-how in the blockchain sector to deliver top-notch smart contract audit solutions, tailored to the clients' evolving business needs.

Check our website for further information: https://chainsulting.de