Deployment Verification
NM-0299 Lido wstETH on Starknet

NETHERMIND
SECURITY

(Nov 14, 2024)

Contents

1 Executive Summary

This document outlines the deployment verification and audit reports' assessment by Nethermind Security for Lido wstETH (Wrapped Staked ETH to Starknet through StarkGate, the canonical bridge of Starknet.

The verification reported in this document assess compliance with the following key areas:

- **Audit Report Verification**: Section 3 provides verification of audit report existence, scope compliance, vulnerability fixes, and final commits.
- **Role Model Verification**: Section 4 outlines the findings on defined roles, verifying the presence of members assigned to each respective role.
- **State Investigation**: Section 5 details collected information, including slot values and the most relevant state variables for each inspected contract, with the latest transaction hashes where updates occurred.
- **Deployment Script Check**: Section 6 provides insights on deployment scripts for Ethereum and Starknet contracts.
- **Documentation Check**: Section 7 evaluates documentation across contexts such as bridge architecture, updates, and deployment procedures.
- **wstETH Rollup Bridging Guide**: Section 8 reports on the verification of control measures aligning with Lido's best practices, including pause functionality and lock-and-mint bridge mechanisms.

2 Scope

This report encompasses the verification of the following deployed contracts:

Contract	Address
StarkGate: wstETH Bridge Ethereum (Proxy)	0xBf67F59D2988A46FBFF7ed79A621778a3Cd3985B
StarkGate: wstETH Bridge Ethereum (Implementation	0x6ad74d4b79a06a492c288ef66ef868dd981fdc85
StarkGate: wstETH Bridge Starknet	0x0088eedbe2fe3918b69ccb411713b7fa72079d4eddf291103ccbe41e78a9615c
StarkGate: wstETH Token Starknet	0x042b8f0484674ca266ac5d08e4ac6a3fe65bd3129795def2dca5c34ecc5f96d2
GovernorForwarder	0x07ba4bb6a9ec398598c9c08424af6bdb83f56e78ffc8f07f0da0dfae8deca432

Audit Reports

- **StarkGate 2.0 bridge and L2 ERC20 - Oct 2024**: StarkGate - Zellic Audit Report
- **StarkGate 2.0 bridge and L2 ERC20 - Oct 2023**: StarkGate - Zellic Audit Report
- **Lido - Starknet governance forwarder**: QuillAudit_Reports/Lido Starknet Governance Forwarder Audit Report

3 Audit report verification

Description: The Nethermind Security team verified the presence of three audit reports. The investigation relies on scope compliance, fixing vulnerabilities, and the final commit review (checking if the recommendations were applied correctly). The outcome of this investigation is listed below:

- **Lido Starknet Governance Forwarder** - This audit was conducted by QuillAudits for the `BridgeExecutor` contract implemented in the Cairo language. The initial commit hash is `80545985b65e151729cfb9a0b4173a7a102b4e0e`. The report consists of five issues classified into different severity risks: High (1), Medium (1), and Low (3). All of them were fixed, except for one Low. The commit `a4965a1663beb28ca4913aacfcf8775e330c35e4` contains all the fixes for the issues, except for one low issue with the status `Acknowledged`. **Report typo**: Page 3 summarizes the number of issues, their severities, and statuses. However, it lacks one Low severity issue and its status.
- **StarkGate 2.0 bridge and L2 ERC20**: This audit was conducted in October 2023 by Zellic for several contracts: `TokenBridge`, `PermissionedERC20`, `StarknetTokenBridg`, `StarkgateManager`, `StarkgateRegistry`, and `StarkgateEthBridge`. The initial commit hash is `78b73afe5de30a0cd6d7e05bce8ca110be57d10a`. Zellic team detected six issues classified into Medium (1), Low (2), and Informational (3). The final commit hash `eedee8304e8c407c2e0e03c83187dbc5dcc6787e` contains all the fixes for issues, with the exception of two low severity issues with the status `Acknowledged`. **Remarks**: The commit hash that contains the deployed contracts is not the final commit hash described in the audit report. The deployed versions corresponds to the commit `5a10fd263d29cd032b7229691d043520edae0737`. The differences between the two versions can be seen in the following link.
- **StarkGate 2.0 bridge and L2 ERC20**: This audit was conducted in October 2024 by Zellic for all Cairo and Solidity code in the `src` directory. The initial commit hash is `5a10fd263d29cd032b7229691d043520edae0737`. Zellic team detected one issue classified into High (1) and the final commit hash `45941888479663ac93e898cd7f8504fa9066c54c` contains the fix for it.

4 Role Model Verification

4.1 StarkGate: wstETH bridge

Both sides of the bridge use a Role Model composed of multiple roles with separate responsibilities. Each of these roles is administered by another role in charge of managing the role's members. Roles are identified by their `bytes32` identifiers, which are defined as public constant hash digests. The following table lists the existing roles and their corresponding hashes:

Role Admin	Value
GOVERNANCE_ADMIN	03711C9D994FAF6055172091CB841FD4831AA743E6F3315163B06A122C841846
UPGRADE_GOVERNOR	0251E864CA2A080F55BCE5DA2452E8CFCAFDBC951A3E7FFF5023D558452EC228
APP_ROLE_ADMIN	03E615638E0B79444A70F8C695BF8F2A47033BF1CF95691EC3130F64939CEE99
APP_GOVERNOR	00D2EAD78C620E94B02D0A996E99298C59DDCCFA1D8A0149080AC3A20DE06068
OPERATOR	023EDB77F7C8CC9E38E8AFE78954F703AEEDA7FFFE014EEB6E56EA84E62F6DA7
TOKEN_ADMIN	0128D63ADBF6B09002C26CAF55C47E2F26635807E3EF1B027218AA74C8D61A3E
SECURITY_ADMIN	026BD110619D11CFDFC28E281DF893BC24828E89177318E9DBD860CDAEDEB6B3
SECURITY_AGENT	037693BA312785932D430DCCF0F56FFEDD0AA7C0F8B6DA2CC4530C2717689B96

The admin relationship between these roles is expected to adhere to the following table:

Role	Role Admin
GOVERNANCE_ADMIN	GOVERNANCE_ADMIN
UPGRADE_GOVERNOR	GOVERNANCE_ADMIN
APP_ROLE_ADMIN	GOVERNANCE_ADMIN
APP_GOVERNOR	APP_ROLE_ADMIN
OPERATOR	APP_ROLE_ADMIN
TOKEN_ADMIN	APP_ROLE_ADMIN
SECURITY_ADMIN	SECURITY_ADMIN
SECURITY_AGENT	SECURITY_ADMIN

4.2 L1 Bridge

For the `L1 Endpoint`, the following two roles contain at least one member:

- `GOVERNANCE_ADMIN` - This role has the authority to assign an `account` as proxy governor and clears pending governor candidate. It can also remove an `account` from proxy governor role and clears pending governor candidate. Additionally, only this role can add and remove members from the roles `GOVERNANCE_ADMIN`, `UPGRADE_GOVERNOR`, and `APP_ROLE_ADMIN`.
- `SECURITY_ADMIN` - This role has the ability to unset the withdrawal limit for a token.

The only member of these two roles is the account `0x015277f49d5dD035A5F3Ce34aD5eBfDBaCA0C6Ec`.

4.3 L2 Bridge

For the `L2 Endpoint`, the following three roles contain at least one member:

- `GOVERNANCE_ADMIN` - This is the only role that can add and remove members from the roles `GOVERNANCE_ADMIN`, `UPGRADE_GOVERNOR`, and `APP_ROLE_ADMIN`.
- `UPGRADE_GOVERNOR` - This role can add a new implementation data, remove existing implementation data, and replace the non-finalized current implementation data to one that was previously added and whose activation time had passed.
- `SECURITY_ADMIN` - This role has the ability to disable withdrawal limit for a token.

The only member of these three roles is the account `0x45653a585ec600d7f050279376d353f84c7d6d09cf225aefbcb83bfaf4abb07`.

4.4 StarkGate: wstETH Token

For the `L2 token`, the following three roles contain at least one member; however, one of them is not used in the current implementation:

- `GOVERNANCE_ADMIN` - This role exclusively has the authority to add and remove members from both `GOVERNANCE_ADMIN` and `UPGRADE_GOVERNOR`.
- `UPGRADE_GOVERNOR` - This role alone can add new implementation data, remove existing implementation data, and replace the non-finalized current implementation data to one that was previously added and whose activation time had passed.
- `SECURITY_ADMIN` - (This role is not used in the current implementation)

The only member of these three roles is the account `0x45653a585ec600d7f050279376d353f84c7d6d09cf225aefbcb83bfaf4abb07`.

4.5 Governance Forwarder

The `Governance Forwarder` contract uses two roles for executing critical actions.

- The `Guardian` role is assigned to a Starknet account. The `Guardian` is the only role that can `Cancel` actions. The role is currently assigned to the account `0x0399eb3460eb885b5e1f5f2aebf63dadb7493f4cbf34868434366bbb55422c4e`.
- The `Ethereum Governance Executor` is assigned to an Ethereum account. Only the account assigned with this role can send actions that will be executed by the `Governance Forwarder`. The current account is `0x46c1e48b26d1b35b63b1e852cf34bee589184557`.

Besides `Cancel` operations, all the other operations for managing the `Governance Forwarder` contract must be executed through messages sent by the `Ethereum Governance Executor`.

5 State Investigation

In this section, the Nethermind Security team presents the information collected during the storage inspection. The information includes the slots and values for the most relevant state variables of each contract, together with the latest transaction where each variable was updated.

5.1 L1 Bridge

- **L1 Bridge address - Mainnet**: 0xBf67F59D2988A46FBFF7ed79A621778a3Cd3985B
- **Implementation contract address**: 0x6ad74d4b79a06a492c288ef66ef868dd981fdc85

The following table presents the governance slots for entities (`GovernanceInfoStruct` struct). The constant `PROXY_GOVERNANCE_TAG` is the string key used in the Governance storage mapping `governanceInfo`. The value stored in the constant is:

```
PROXY_GOVERNANCE_TAG ="StarkEx.Proxy.2019.GovernorsInformation"
```

For simplicity, the label `MEMBER_ADDR_L1` in the tables below represents the only member set as effective governor and the address is `0x015277f49d5dD035A5F3Ce34aD5eBfDBaCA0C6Ec`.

State Variable	Slot	Value	Last Update at Txn Hash
governanceInfo[PROXY_GOVERNANCE_TAG] .effectiveGovernors[MEMBER_ADDR_L1]	0x90b92208de9555671961e2b66c6af94c0b25f16282e9f6cd 28c527610659567d	1	0x3403a8...73030d
governanceInfo[PROXY_GOVERNANCE_TAG] .candidateGovernor	0x45f38e273862f8834bd2fe7a449988f63de55a7a5b685dea 46ccedeb69cf0e27	0x0	0x3403a8...73030d
governanceInfo[PROXY_GOVERNANCE_TAG] .initialized	0x01b0dddd7d759bd205fef59d2fa4288af3a4d6c44d638e11 57ef86b32e9e1c1f	1	0x430625...e55f91

Governance slots for entities

The table below presents the storage slots used throughout the Proxy pattern. They define an arbitrary location to avoid overlapping by the logical contracts.

State Variable	Slot
IMPLEMENTATION_SLOT	0x177667240aeeea7e35eabe3a35e18306f336219e1386f7710a6bf8783f761b24
L2_BRIDGE_TAG	hash of STARKNET_TOKEN_BRIDGE_L2_TOKEN_CONTRACT
UPGRADE_DELAY_SLOT	0xc21dbb3089fcb2c4f4c6a67854ab4db2b0f233ea4b21b21f912d52d18fc5db1f
FINALIZED_STATE_SLOT	0x7d433c6f837e8f93009937c466c82efbb5ba621fae36886d0cac433c5d0aa7d2
tokenSettings()[token].maxTotalBalance	0x3041252d52563574c35b8d1cd49a52a0cc144f208981902e6ce7a5ba4a6058f4

Storage slots for the Proxy pattern

The table below presents the state variables related to the proxy with their respective values and transaction hashes of the last update.

State Variable	Value	Last Updated at Txn Hash
IMPLEMENTATION_SLOT	0x6ad74d4b79a06a492c288ef66ef868dd981fdc85	0xcd8680...49d026
L2_BRIDGE_TAG	0x0088eedbe2fe3918b69ccb411713b7fa72079d4eddf291103ccbe41e78a9615c	0xe26bb9...ff6734
UPGRADE_DELAY_SLOT	259200	At deployment transaction
FINALIZED_STATE_SLOT	0	It never changed
tokenSettings() [token].maxTotalBalance	115792089237316195423570985008687907853269984665640564039457584007 913129639935	0x1e48c9...4746bc

Values stored in the state variables associated to the proxy

The following two tables provide an overview of role-based access control, identifying the administrators for `SECURITY_ADMIN` and `GOVERNANCE_ADMIN` roles. The first table displays the storage slots where the administrators' addresses are held, specifying the slot associated with each administrator role.

State Variable	Slot
_roles()[SECURITY_ADMIN] .members[MEMBER_ADDR_L1]	0x11f2d9aee0769e146db4b38e9cdfa21aa126ce369acbce687dce5816c9ef09a7
_roles()[SECURITY_AGENT].adminRole	0x133e8fc3bf3b8cbe8741215d94714cce4d113437771c895d8581377f86bc9b7a
_roles()[SECURITY_ADMIN].adminRole	0x2c11a1f9c63817dbb9f0faa966615764d2db5d6e008269e948a99e0b52181c24
_roles()[UPGRADE_GOVERNOR].adminRole	0x3504a568a75815c7d149918f124b8511ce97b33034fa85f36f1c13e849386cce
_roles()[TOKEN_ADMIN].adminRole	0xb92d8dcbe87ba2b1e5aaf88aae12a6f89f52f6ff057743f29c5ffdb55498b6f8
_roles()[OPERATOR].adminRole	0xb30300bed82b2e0ff889fe81a4b0a9ad17f79a0b7b748f3da5b523151d502317
_roles()[GOVERNANCE_ADMIN].adminRole	0xa5fdb349cc4ffac7e8ce7d3b075149d1bc847367d814e69a9beca89ef02db8b1
_roles()[APP_ROLE_ADMIN].adminRole	0xfad5851dfcf56d90e05c22302d339e27ef0eb32d2866743a1bd2461c6925a7f8
_roles()[APP_GOVERNOR].adminRole	0x789965849d43665eb0fa09b289ebd3629b94e51d8eb419502fa992de138405fd
_roles()[GOVERNANCE_ADMIN] .members[MEMBER_ADDR_L1]	0xd84cccb3683fef6766d20d4e907b11b3fa6189200aac33443ea861c67af0dc5d

Storage slots where the administrator's addresses are held

The table below describes an overview of the role-based access control with the values in each state variable. These values were set at transaction 0x4eb13c...7baa6e.

For simplicity, the values for role_admin are presented with the respective role name in the table. Their values are listed below:

- **GOVERNANCE_ADMIN**: 0x03711c9d994faf6055172091cb841fd4831aa743e6f3315163b06a122c841846
- **APP_ROLE_ADMIN**: 0x03e615638e0b79444a70f8c695bf8f2a47033bf1cf95691ec3130f64939cee99
- **SECURITY_ADMIN**: 0x026bd110619d11cfdfc28e281df893bc24828e89177318e9dbd860cdaedeb6b3

State Variable	Value	Last Updated at Txn Hash
_roles()[SECURITY_ADMIN] .members[MEMBER_ADDR_L1]	1	0x4eb13c...7baa6e
_roles()[SECURITY_AGENT].adminRole	SECURITY_ADMIN	0x4eb13c...7baa6e
_roles()[SECURITY_ADMIN].adminRole	SECURITY_ADMIN	0x4eb13c...7baa6e
_roles()[UPGRADE_GOVERNOR].adminRole	GOVERNANCE_ADMIN	0x4eb13c...7baa6e
_roles()[TOKEN_ADMIN].adminRole	APP_ROLE_ADMIN	0x4eb13c...7baa6e
_roles()[OPERATOR].adminRole	APP_ROLE_ADMIN	0x4eb13c...7baa6e
_roles()[GOVERNANCE_ADMIN].adminRole	GOVERNANCE_ADMIN	0x4eb13c...7baa6e
_roles()[APP_ROLE_ADMIN].adminRole	GOVERNANCE_ADMIN	0x4eb13c...7baa6e
_roles()[APP_GOVERNOR].adminRole	APP_ROLE_ADMIN	0x4eb13c...7baa6e
_roles()[GOVERNANCE_ADMIN] .members[MEMBER_ADDR_L1]	1	0x4eb13c...7baa6e

Role-based access control overview

5.2 L2 Bridge

- **L2 Bridge address - Mainnet**: 0x0088eedbe2fe3918b69ccb411713b7fa72079d4eddf291103ccbe41e78a9615c.
- **L2 Bridge class hash - Mainnet**: 0x0358663e6ed9d37efd33d4661e20b2bad143e0f92076b0c91fe65f31ccf55046 corresponding to the commit 5a10fd263d29cd032b7229691d043520edae0737

The following table shows the slots for the most relevant state variables related to the bridge.

State variable	Slot
l1_bridge	0x00c88ee7a00e0b95f1138ef53d396c4327eeed7f9677bbd02ce82a663537b1cf
erc20_class_hash	0x03e4b2efa9f3dc5a5ca304578e8f83c116445fc36a80508f598770ab9d0ba8fa
l2_token_governance	0x03acd88ba6181ba1fb253286b276575767587130253bbbed7f7270cd11e8be7d
l1_l2_token_map(0x7f39C581F59 5B53c5cb19bD0b3f8dA6c935E2Ca0)	0x54ff3721ea40be0e7c25189adedd9b4c064814f232a61fa25fc4158d5aafbf7
l2_l1_token_map(0x042b8f0484674ca266ac5 d08e4ac6a3fe65bd3129795def2dca5c34ecc5f96d2)	0x4ab3d9efee6d7e88294d6bb5bd78219cff02c9dcad28d271c118b937d75b1e5
daily_withdrawal_limit_pct	0x031b6e2ac42e0e554bf70a1d3d890fda8a700de6068f723a0ec603cac893a1e4
l2_token	0x01dc79e2fd056704ede52dca5746b720269aaa5da53301dff546657c16ca07af
upgrade_delay	0x003fc801c47df4de8d5835f8bfd4d0b8823ba63e5a3f278086901402d680abfc
finalized	0x00cfc0e4c73ce8e46b07c3167ce01ce17e6c2deaaa5b88b977bbb10abe25c9ad

Storage slots for the most state variables for the L2 bridge

The following table shows the values for the most relevant state variables related to the bridge.

State variable	Value	Last updated at Txn Hash
l1_bridge	1092735609972394726528730534548720965203717757019	0x067f58...5140c3
erc20_class_hash	0	It never changed
l2_token_governance	0	It never changed
l1_l2_token_map(0x7f39C581F59 5B53c5cb19bD0b3f8dA6c935E2Ca0)	1886212889629631188189497155848883534738756148921111726686756987927630157522	0x434be2...c0cb72 0x434be2...c0cb72
l2_l1_token_map(0x042b8f0484674ca266ac5 d08e4ac6a3fe65bd3129795def2dca5c34ecc5f96d2)	726330175714135941764069406682033110407748398240	
daily_withdrawal_limit_pct	5	0x434be2...c0cb72
l2_token	1886212889629631188189497155848883534738756148921111726686756987927630157522	0x75a942...dc6f05
upgrade_delay	0	At deployment transaction
finalized	0	It never changed

Values Stored in the State Variables Associated to the Bridge

The following table outlines the storage slots used in the L2 endpoint for the role model. The storage variable `role_admin` stores the admin ID for each role ID in the corresponding slots listed in the table. Moreover, the table also presents the slots where the `role_members` variable stores a value of `1`, confirming the specified role for the account **`MEMBER_L2_BRIDGE`**=`0x45653a585ec600d7f050279376d353f84c7d6d09cf225aefbcb83bfaf4abb07`.

State variable	Slot
role_admin(APP_GOVERNOR)	0x55f3055e2322652ee76c141ee039dd148f75ca61fd2419f0b15e898fe58ebbb
role_admin(APP_ROLE_ADMIN)	0x61f2cbdd29b41cddc7201657f27649d1b4f82f3a902b74c423e85cd873ba383
role_admin(GOVERNANCE_ADMIN)	0x51ff9d8e3b9d9153881c6d021c07e41e820360c3abbff54b127a8bc6d613574
role_admin(OPERATOR)	0x3357edccf7145387e0368a3ae6bc29ac2faed2e804582512b065eaec608d8f9
role_admin(TOKEN_ADMIN)	0x2db6a3aac3da6dac69be905507cfea3f35f35d16e4f8724a63fbcd4f1b7b28d
role_admin(UPGRADE_GOVERNOR)	0x67eebb8a37da29031c921351ac6573033c0e3db1d2f205e01a4339afd272a14
role_admin(SECURITY_ADMIN)	0x37ab142dd955dadabd954354586ff943a15e7f4c5ae4eaaac1dee2539377465
role_admin(SECURITY_AGENT)	0x2d2fdbaf84c6c5ba492118bd3c1fb52ddbc6801331cb9ff654f7684398cec56
role_members(GOVERNANCE_ADMIN, MEMBER_L2_BRIDGE)	0x6178675768bb1b18f5efbe571ca463249bcbbc56f6389b56ecbd862a7363205
role_members(UPGRADE_GOVERNOR, MEMBER_L2_BRIDGE)	0x6ba59b7522f54b6ae9459b798b25f5a2e372360dafbd4dba776947c716bc709
role_members(SECURITY_ADMIN, MEMBER_L2_BRIDGE)	0x151991a0cf89dd4cb2a2cfac842779667cb01c79e3673c7bfc567ff8174919b

Slots for Role Model Implementation in the L2 Endpoint of the Bridge

The table below reveals the current values for `role_admin` and `role_members` for a specified role and their last update. As can be noted, **`MEMBER_L2_BRIDGE`** is the only member for the three roles: `GOVERNANCE_ADMIN`, `UPGRADE_GOVERNOR`, and `SECURITY_ADMIN`. For convenience, the values for `role_admin` are presented with the respective role name in the table. Their values are listed below:

- **`GOVERNANCE_ADMIN`**: 1556789761824654631941645422192241267775035395971414146468438977535375841350
- **`APP_ROLE_ADMIN`**: 176346099131589563331632914670641295067070635456104341766362342424175214146 5
- **`SECURITY_ADMIN`**: 109512123972335277077216659171873768696620434525311902747795259873638421061 1

State variable	Value	Last Updated at Txn Hash
role_admin(APP_GOVERNOR)	APP_ROLE_ADMIN	0x434be2...c0cb72
role_admin(APP_ROLE_ADMIN)	GOVERNANCE_ADMIN	0x434be2...c0cb72
role_admin(GOVERNANCE_ADMIN)	GOVERNANCE_ADMIN	0x434be2...c0cb72
role_admin(OPERATOR)	APP_ROLE_ADMIN	0x434be2...c0cb72
role_admin(TOKEN_ADMIN)	APP_ROLE_ADMIN	0x434be2...c0cb72
role_admin(UPGRADE_GOVERNOR)	GOVERNANCE_ADMIN	0x434be2...c0cb72
role_admin(SECURITY_ADMIN)	SECURITY_ADMIN	0x434be2...c0cb72
role_admin(SECURITY_AGENT)	SECURITY_ADMIN	0x434be2...c0cb72
role_members(GOVERNANCE_ADMIN, MEMBER_L2_BRIDGE)	1	0x434be2...c0cb72
role_members(UPGRADE_GOVERNOR, MEMBER_L2_BRIDGE)	1	0x434be2...c0cb72
role_members(SECURITY_ADMIN, MEMBER_L2_BRIDGE)	1	0x434be2...c0cb72

Current Values and Last Updates for `role_admin` *and* `role_members` *in Specified Roles*

5.3 StarkGate: wstETH Token

- **StarkGate: wstETH Token address - Mainnet**: `0x042b8f0484674ca266ac5d08e4ac6a3fe65bd3129795def2dca5c34ecc5f96d2`
- **StarkGate: wstETH Token class hash - Mainnet**: `0x05ffbcfeb50d200a0677c48a129a11245a3fc519d1d98d76882d1c9a1b19c6ed` corresponding to the commit `5a10fd263d29cd032b7229691d043520edae0737`

The following table shows the slots for the most relevant state variables related to the `L2 token`.

State variable	Slot
permitted_minter	0x1390569bb0a3a722eb4228e8700301347da081211d5c2ded2db22ef389551ab
upgrade_delay	0x003fc801c47df4de8d5835f8bfd4d0b8823ba63e5a3f278086901402d680abfc
finalized	0x00cfc0e4c73ce8e46b07c3167ce01ce17e6c2deaaa5b88b977bbb10abe25c9ad

Slots for the L2 wstETH token

The table below reveals the values stored in the state variables `permitted_minter`, `upgrade_delay`, and `finalized`.

State variable	Value	Last updated at at Txn Hash
permitted_minter	2419397445738757360752830461762744704477102451845266111460970951396416146 84	0x6ee3c9...c86acf
upgrade_delay	0	At deployment transaction
finalized	0	At deployment transaction

Current Values and Last Updates for the L2 wstETH token

The following table shows the slots and current values for the role model implemented in the `L2 StarkGate: wstETH Token`.

State variable	Slot
role_admin(GOVERNANCE_ADMIN)	0x51ff9d8e3b9d9153881c6d021c07e41e820360c3abbff54b127a8bc6d613574
role_admin(UPGRADE_GOVERNOR)	0x67eebb8a37da29031c921351ac6573033c0e3db1d2f205e01a4339afd272a14
role_members(GOVERNANCE_ADMIN, MEMBER_L2_BRIDGE)	0x6178675768bb1b18f5efbe571ca463249bcbbc56f6389b56ecbd862a7363205
role_members(UPGRADE_GOVERNOR, MEMBER_L2_BRIDGE)	0x6ba59b7522f54b6ae9459b798b25f5a2e372360dafbd4dba776947c716bc709

Slots for the Role Model Implementation in the L2 wstETH token

As indicated in the table above, **`MEMBER_L2_BRIDGE`** in `role_members` is the only member for `GOVERNANCE_ADMIN` and `UPGRADE_GOVERNOR` roles. The address for `MEMBER_L2_BRIDGE` is

```
0x45653a585ec600d7f050279376d353f84c7d6d09cf225aefbcb83bfaf4abb07
```

State variable	Value	Last updated at Txn Hash
role_admin(GOVERNANCE_ADMIN)	GOVERNANCE_ADMIN	0x6c86a5...de61a5
role_admin(UPGRADE_GOVERNOR)	GOVERNANCE_ADMIN	0x6c86a5...de61a5
role_members(GOVERNANCE_ADMIN, MEMBER_L2_BRIDGE)	1	0x6c86a5...de61a5
role_members(UPGRADE_GOVERNOR, MEMBER_L2_BRIDGE)	1	0x6c86a5...de61a5

Current Values and Last Updates for `role_admin` and `role_members` - L2 wstETH token

The value stored for `GOVERNANCE_ADMIN` is:

```
1556789761824654631941645422192241267775035395971414146468438977535375841350
```

5.4 Starknet governance forwarder

StarkGate: Starknet governance forwarder - Mainnet

Address: 0x07ba4bb6a9ec398598c9c08424af6bdb83f56e78ffc8f07f0da0dfae8deca432

Class hash: `0x0611bf8ac1d6e31031e37e428c48547b8be4179d3d047d95fde9452d6c7a6a15`, corresponding to the commit a4965a1663beb28ca.

The following two tables show the slots and current values for state variables that were set at transaction 0x07d925...eacd71.

State variable	Slot
grace_period	0x00c087fc777ac0972310ae93465273c06e090a0a854ff345ff505e9026f74aab
guardian	0x03bcb4375b910093bcf636b6b2f26b26eda2a29ef5a8ee7de44b5743c3bf9a28
minimum_delay	0x02e4019d18ae4a4b56836bf8a3d2640c37c5e30eeedc078039977e2077e73576
delay	0x004f5c4824d6fe4ecb5e00033e433ffbfd69fc213c4f1e41b731944f129687e5
maximum_delay	0x02886b30d4a1385ad9f9fcace02b5c03a634c32385436d5d94ed4fe49b71fee0
ethereum_governance_executor	0x01dfe3f85d5823443b30a33c01355b7d453bc48eee894156fc9e9fba2e01292b

Slots for the Starknet Governance Forwarder

The `GUARDIAN` is the only role that can *cancel* actions and the currently assigned account is:

```
1628889469176486951891799519361523237603513148881369604904406283515871767630
```

State variable	Value	Last updated at Txn Hash
grace_period	36000	0x07d925...eacd71
guardian	GUARDIAN	0x07d925...eacd71
minimum_delay	480	0x07d925...eacd71
delay	600	0x07d925...eacd71
maximum_delay	7200	0x07d925...eacd71
ethereum_governance_executor	403953306733026111984514834281377691541756724567	0x07d925...eacd71

Current Values and Last Updates for the State Variables - Starknet Governance Forwarder

6 Deployment Script Check

6.1 StarkGate: wstETH Bridge

No deployment scripts are provided for the `Ethereum` or `Starknet` contracts. However, scripts for compiling the contracts in the expected version are provided. In the case of `Starknet`, the class hash is already declared, so only a `DEPLOY` transaction using the class hash is needed.

6.2 Governance Forwarder

Declare and deployment scripts are provided for the `GovernanceForwarder` contract.

7 Documentation Check

7.1 StarkGate: wstETH Bridge

This section checks the public documentation available for the bridge.

Documentation explaining the bridge architecture and flows can be found in this link. However, there is no documentation specifically detailing the deployment, initialization, and multiple updates that have been done to the `wstETH` bridge.

7.2 Governance Forwarder

The README file in the contract's repository provides information about the contract. Instructions on how to execute the provided scripts can also be found there. This information can be complemented with the Aave Governance Crosschain Bridges documentation, which inspired this contract.

8 wstETH Rollup Bridging Guide

This section explores how the `StarkGate: wstETH` bridge adheres to the baseline recommendations encouraged in the bridging guide created by the Network Expansion Workgroup.

8.1 R-1: Audited Code and Verifiable Deployment

A third party audited the contracts, and the relevant issues discovered were fixed. However, the deployed code does not correspond to the specific commit where the code was audited or the fixes were applied. The contracts on the Ethereum Mainnet are verified, and their code can be checked from block explorers. However, contracts on `Starknet` have not been verified.

8.2 R-2: "Lock and Mint" Bridge Mechanics

The `StarkGate: wstETH` uses a "Lock and Mint" mechanism.

8.3 R-3: Usage of Canonical Bridge

`StarkGate` has been developed by `Starkware` and is considered the canonical bridge for `Starknet`.

8.4 R-4: L2 wstETH Token Upgradable

The proxy architecture is not widely used in `Starknet` due to the existence of the class hash concept. Accounts can change their class hash through system calls. The `wstETH` token deployed on `Starknet` includes an upgradability mechanism that allows changing its behavior if needed.

8.5 R-5: Bridging L1 Lido DAO Decisions

At the current moment, there is no `Governance Forwarder` contract. The `Starknet` contract is managed by the account:

```
0x45653a585ec600d7f050279376d353f84c7d6d09cf225aefbcb83bfaf4abb07
```

8.6 R-6: Dedicated Upgradable Bridge Instances

The reviewed contracts are solely used for bridging `wstETH`. Other tokens have their own bridge or use the `MultiToken StarkGate` bridge. Both endpoints were designed with upgradable capabilities.

8.7 R-7: Pausable Deposits and Withdrawals

The bridge does not implement direct functionality for pausing/unpausing. However, similar results could be achieved through the "limits" functionality.

8.8 R-8: Support of ERC-2612 Permit Enhanced with EIP-1271

The `StarkGate: wstETH` token does not implement the `ERC-2612` or `EIP-1271`.

9 About Nethermind

Nethermind is a Blockchain Research and Software Engineering company. Our work touches every part of the web3 ecosystem - from layer 1 and layer 2 engineering, cryptography research, and security to application-layer protocol development. We offer strategic support to our institutional and enterprise partners across the blockchain, digital assets, and DeFi sectors, guiding them through all stages of the research and development process, from initial concepts to successful implementation.

We offer security audits of projects built on EVM-compatible chains and Starknet. We are active builders of the Starknet ecosystem, delivering a node implementation, a block explorer, a Solidity-to-Cairo transpiler, and formal verification tooling. Nethermind also provides strategic support to our institutional and enterprise partners in blockchain, digital assets, and decentralized finance (DeFi). In the next paragraphs, we introduce the company in more detail.

Blockchain Security: At Nethermind, we believe security is vital to the health and longevity of the entire Web3 ecosystem. We provide security services related to Smart Contract Audits, Formal Verification, and Real-Time Monitoring. Our Security Team comprises blockchain security experts in each field, often collaborating to produce comprehensive and robust security solutions. The team has a strong academic background, can apply state-of-the-art techniques, and is experienced in analyzing cutting-edge Solidity and Cairo smart contracts, such as ArgentX and StarkGate (the bridge connecting Ethereum and StarkNet). Most team members hold a Ph.D. degree and actively participate in the research community, accounting for 240+ articles published and 1,450+ citations in Google Scholar. The security team adopts customer-oriented and interactive processes where clients are involved in all stages of the work.

Blockchain Core Development: Our core engineering team, consisting of over 20 developers, maintains, improves, and upgrades our flagship product - the Nethermind Ethereum Execution Client. The client has been successfully operating for several years, supporting both the Ethereum Mainnet and its testnets, and now accounts for nearly a quarter of all synced Mainnet nodes. Our unwavering commitment to Ethereum's growth and stability extends to sidechains and layer 2 solutions. Notably, we were the sole execution layer client to facilitate Gnosis Chain's Merge, transitioning from Aura to Proof of Stake (PoS), and we are actively developing a full-node client to bolster Starknet's decentralization efforts. Our core team equips partners with tools for seamless node set-up, using generated docker-compose scripts tailored to their chosen execution client and preferred configurations for various network types.

DevOps and Infrastructure Management: Our infrastructure team ensures our partners' systems operate securely, reliably, and efficiently. We provide infrastructure design, deployment, monitoring, maintenance, and troubleshooting support, allowing you to focus on your core business operations. Boasting extensive expertise in Blockchain as a Service, private blockchain implementations, and node management, our infrastructure and DevOps engineers are proficient with major cloud solution providers and can host applications in-house or on clients' premises. Our global in-house SRE teams offer 24/7 monitoring and alerts for both infrastructure and application levels. We manage over 5,000 public and private validators and maintain nodes on major public blockchains such as Polygon, Gnosis, Solana, Cosmos, Near, Avalanche, Polkadot, Aptos, and StarkWare L2. Sedge is an open-source tool developed by our infrastructure experts, designed to simplify the complex process of setting up a proof-of-stake (PoS) network or chain validator. Sedge generates docker-compose scripts for the entire validator set-up based on the chosen client, making the process easier and quicker while following best practices to avoid downtime and being slashed.

Cryptography Research: At Nethermind, our Cryptography Research team is dedicated to continuous internal research while fostering close collaboration with external partners. The team has expertise across a wide range of domains, including cryptography protocols, consensus design, decentralized identity, verifiable credentials, Sybil resistance, oracles, and credentials, distributed validator technology (DVT), and Zero-knowledge proofs. This diverse skill set, combined with strong collaboration between our engineering teams, enables us to deliver cutting-edge solutions to our partners and clients.

Smart Contract Development & DeFi Research: Our smart contract development and DeFi research team comprises 40+ world-class engineers who collaborate closely with partners to identify needs and work on value-adding projects. The team specializes in Solidity and Cairo development, architecture design, and DeFi solutions, including DEXs, AMMs, structured products, derivatives, and money market protocols, as well as ERC20, 721, and 1155 token design. Our research and data analytics focuses on three key areas: technical due diligence, market research, and DeFi research. Utilizing a data-driven approach, we offer in-depth insights and outlooks on various industry themes.

Our suite of L2 tooling: Warp is Starknet's approach to EVM compatibility. It allows developers to take their Solidity smart contracts and transpile them to Cairo, Starknet's smart contract language. In the short time since its inception, the project has accomplished many achievements, including successfully transpiling Uniswap v3 onto Starknet using Warp.

- **Voyager** is a user-friendly Starknet block explorer that offers comprehensive insights into the Starknet network. With its intuitive interface and powerful features, Voyager allows users to easily search for and examine transactions, addresses, and contract details. As an essential tool for navigating the Starknet ecosystem, Voyager is the go-to solution for users seeking in-depth information and analysis;
- **Horus** is an open-source formal verification tool for StarkNet smart contracts. It simplifies the process of formally verifying Starknet smart contracts, allowing developers to express various assertions about the behavior of their code using a simple assertion language;
- **Juno** is a full-node client implementation for Starknet, drawing on the expertise gained from developing the Nethermind Client. Written in Golang and open-sourced from the outset, Juno verifies the validity of the data received from Starknet by comparing it to proofs retrieved from Ethereum, thus maintaining the integrity and security of the entire ecosystem.

Learn more about us at nethermind.io.

General Advisory to Clients

As auditors, we recommend that any changes or updates made to the audited codebase undergo a re-audit or security review to address potential vulnerabilities or risks introduced by the modifications. By conducting a re-audit or security review of the modified codebase, you can significantly enhance the overall security of your system and reduce the likelihood of exploitation. However, we do not possess the authority or right to impose obligations or restrictions on our clients regarding codebase updates, modifications, or subsequent audits. Accordingly, the decision to seek a re-audit or security review lies solely with you.

Disclaimer

This report is based on the scope of materials and documentation provided by you to Nethermind in order that Nethermind could conduct the security review outlined in **1. Executive Summary** and **2. Audited Files**. The results set out in this report may not be complete nor inclusive of all vulnerabilities. Nethermind has provided the review and this report on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. Blockchain technology remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. This report does not indicate the endorsement of any particular project or team, nor guarantee its security. No third party should rely on this report in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. To the fullest extent permitted by law, Nethermind disclaims any liability in connection with this report, its content, and any related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. Nethermind does not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and Nethermind will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.