G0 Group

Security Review of
Hashi

March 28, 2024

Hashi / March 2024

Files in scope

Following files in https://github.com/gnosis/hashi/tree/0d441d83762b2a73168b7d605d5a9f793678997d/packages/evm/

```
contracts/
    Hashi.sol
    Yaho.sol
    Yaru.sol
    utils/
        HeaderStorage.sol
        MessageHashCalculator.sol
        MessageIdCalculator.sol
    ownable/
        GiriGiriBashi.sol
        ShoyuBashi.sol
        ShuSo.sol
```

Current status

All reported issues have been fixed by the developer.

Issues

1. The same message can be relayed multiple time in Yaho

type: security / severity: major

There's no protection from the messages argument of `Yaho.relayMessagesToAdapte` containing duplicit messages. The mappings get reset only after all of the messages are verified.

status - fixed

The issue has been fixed and is no longer present in:
https://github.com/gnosis/hashi/tree/f1a9fdb2998c7024268e9c69777f4dc43d2f775e/packages/evm

2. Two messages in Yaho can end up with the same messageHash

type: security / severity: medium

Using `gasleft(` as part of the salt in `Yaho` doesn't ensure uniqueness, since the messages can be submitted in the same block in two different transactions with the same remaining gas
https://github.com/gnosis/hashi/blob/0d441d83762b2a73168b7d605d5a9f793678997d/packages/evm/contracts/Yaho.sol#L149

status - fixed

The issue has been fixed and is no longer present in:
https://github.com/gnosis/hashi/tree/f1a9fdb2998c7024268e9c69777f4dc43d2f775e/packages/evm

3. Missing authentication in GiriGiriBashi.setChallengeRange

type: security / severity: critical

`GiriGiriBashi.setChallengeRa` function is missing authentication.

status - fixed

The issue has been fixed and is no longer present in:
https://github.com/gnosis/hashi/tree/f1a9fdb2998c7024268e9c69777f4dc43d2f775e/packages/evm

4. _adapters argument in GiriGiriBashi.resolveChallenge is not validated

type: security / severity: critical

In `GiriGiriBashi.resolveChalle`, `_adapter` array is never checked to belong to the domain and for uniqueness.

status - fixed

The issue has been fixed and is no longer present in:
https://github.com/gnosis/hashi/tree/f1a9fdb2998c7024268e9c69777f4dc43d2f775e/packages/evm

5. If the same adapter is used across multiple domains, settings will be overwritten

type: implementation / severity: major

`IAdapter.getHash(domain, i` implies that the same adapter can be used across multiple domains, but in `GiriGiriBashi` `setting` mapping doesn't allow to set separate adapter settings for each domain. This also means that successive calls to `enableAdapter` with the same adapter address but different domain will overwrite settings set for previous domain.

status - fixed

The issue has been fixed and is no longer present in:
https://github.com/gnosis/hashi/tree/f1a9fdb2998c7024268e9c69777f4dc43d2f775e/packages/evm

6. In GiriGiriBashi.resolveChallenge adapter is allowed to be in the _adapters array

type: security / severity: critical

In `GiriGiriBashi.resolveChalle` adapter is allowed to be in the `_adapter` array. So if threshold is `2` and there are `3` adapters in total, it will consider the adapter to behave correctly even if it disagrees with the other two adapters if its address (and just its address) is provided in both adapter and `_adapter`.

status - fixed

The issue has been fixed and is no longer present in:
https://github.com/gnosis/hashi/tree/f1a9fdb2998c7024268e9c69777f4dc43d2f775e/packages/evm

7. Implementation issues in GiriGiriBashi.declareNoConfidence

type: implementation / severity: critical

First problem is that `declareNoConfidenc` doesn't allow to prove no confidence in all cases. Let's say there's `6` adapters and the threshold is `4`, the adapters provide three different answers, each of which is agreed on by two adapters. It's impossible to provide a set of `4` adapters that all provide different answers, yet that's required by the code to prove no confidence. The other issue is this: Let's say threshold is `3`, total count is `5`. We have `2` adapters reporting `hashA`, `1` adapter reporting `hashB` and `2` adapters reporting `bytes32(0)` because they haven't got the answer yet. We're potentially going to converge on one of the two hashes, but `GiriGiriBashi.declareNoConfide` will assume we're in a no confidence state.

status - fixed

The issue has been fixed and is no longer present in:
https://github.com/gnosis/hashi/tree/f1a9fdb2998c7024268e9c69777f4dc43d2f775e/packages/evm

8. In ShuSo threshold can be lower than majority count of the adapters

type: implementation / severity: major

In `ShuSo` `threshol` can be lower than majority of the adapters. This means that `_getHash` can return different answers for the same domain and id and also that return value of `_getThresholdHas` can depend on the order of the adapters. This fundamentally breaks the challenge mechanism in `GiriGiriBash`, since there can be multiple sets of adapters that surpass the threshold but give different answers.

status - fixed

The issue has been fixed and is no longer present in:
https://github.com/gnosis/hashi/tree/f1a9fdb2998c7024268e9c69777f4dc43d2f775e/packages/evm

9. Throwing adapter contracts will prevent adapters from being challenged in GiriGiriBashi

type: security / severity: medium

In `GiriGiriBash` If adapters throw instead of returning an incorrect hash, they can't be successfully challenged.

status - fixed

The issue has been fixed and is no longer present in:
https://github.com/gnosis/hashi/tree/f1a9fdb2998c7024268e9c69777f4dc43d2f775e/packages/evm

10. Forced out of gas exception might allow attacker to manipulate a function's return value

type: security / severity: medium

It might be theoretically possible to manipulate the result of `Hashi.checkHashWithThresholdFromAdapt` by passing just enough gas to force an out of gas exception in the last `adapters[i].getHas` external call but still have enough left over for the function to finish execution. This is enabled by the fact that not all of the gas gets passed to the external call, but only `63/64` of the remaining gas. Whether this is an issue in practice depends on the ratio of gas necessary to execute the `adapters[i].getHas` call and the gas necessary to finish executing the top level function. This would only allow an attacker to flip the return value from true to false for certain arguments.

status - fixed

The issue has been fixed and is no longer present in:
https://github.com/gnosis/hashi/tree/f1a9fdb2998c7024268e9c69777f4dc43d2f775e/packages/evm