KESTROVA

Professional Security Report | Solidity static analysis

Report ID SAMPLE Generated 2026-10-27T00:00:00Z Slither, compiler 0.8.38

Executive summary

Risk level: High

Slither flagged 2 high-impact and 2 medium-impact patterns that should be reviewed before deployment.

2	2	1	2	1
High	Medium	Low	Informational	Optimization

Priority findings

[High] ETH can be sent to an arbitrary address (arbitrary-send-eth) - Vault.sol:20-23, Vault.sol:22

[High] Possible reentrancy (ETH can be withdrawn repeatedly) (reentrancy-eth) - Vault.sol:13-18, Vault.sol:15

[Medium] tx.origin used for authorisation (tx-origin) - Vault.sol:20-23, Vault.sol:21

[Medium] Return value of send() is not checked (unchecked-send) - Vault.sol:20-23, Vault.sol:22

Scope

Vault.sol - 25 lines, 826 bytes

`sha256 2b2ce1fb932f8eaa4adf4dec1ad3ad4e75a2a929a31b9ac8467b9e009b5e969a`

`Combined code hash (sha256): acb623731b502ec9bc225ac50c2427f00d2a9098110b376b83380ca71b62b68d`

Payment receipt

1 USDC on Solana mainnet to HUCBiR2pqkQBjDKq3Nf5xaV4GqjCKwJ9PZNJssi5H8Qn

`Transaction sample report - no payment`

Limitations

Slither detectors only: known code patterns such as reentrancy, unchecked calls, tx.origin auth, arbitrary sends, shadowing. Not covered: business logic, economic/oracle/MEV attacks, access-control intent, upgrade and admin-key risk, contracts or libraries not included in the sources, deployment configuration. Never describe a contract as safe or audited based on this result.

Automated static analysis with Slither. Findings are possible issues, not verified vulnerabilities, and may include false positives. A report with no findings does not mean the code is secure. This report is not a substitute for a professional security audit.

Findings

1. ETH can be sent to an arbitrary address

High impact, Medium confidence | detector arbitrary-send-eth

`Location: Vault.sol:20-23, Vault.sol:22`

What it means: A function sends the contract's ETH to an address chosen by the caller. If access control is missing or weak, anyone may be able to drain funds.

Remediation: Restrict who can call the function (for example onlyOwner or a role check using msg.sender), and send only to trusted or pre-approved addresses.

```
Vault.sweep(address) (Vault.sol#20-23) sends eth to arbitrary user?    Dangerous calls:?    -
to.send(address(this).balance) (Vault.sol#22)
```

2. Possible reentrancy (ETH can be withdrawn repeatedly)

High impact, Medium confidence | detector reentrancy-eth

`Location: Vault.sol:13-18, Vault.sol:15, Vault.sol:17`

What it means: A function sends ETH or calls another contract before it finishes updating its own balances. A malicious contract can call back in during that call and repeat the withdrawal.

Remediation: Update state before making external calls (checks-effects-interactions), and consider a reentrancy guard such as OpenZeppelin ReentrancyGuard.

```
Reentrancy in Vault.withdraw() (Vault.sol#13-18):?    External calls:?    - (ok,None) =
msg.sender.call{value: amt}() (Vault.sol#15)?    State variables written after the call(s):?    -
balances[msg.sender] = 0 (Vault.sol#17)?    Vault.balances (Vault.sol#6) can be used in cross function
reentrancies:?    - Vault.balances (Vault.sol#6)?    - Vault.deposit() (Vault.sol#11)?    -
Vault.withdraw() (Vault.sol#13-18)
```

3. tx.origin used for authorisation

Medium impact, Medium confidence | detector tx-origin

`Location: Vault.sol:20-23, Vault.sol:21`

What it means: Checking tx.origin lets a malicious contract act on behalf of a user who calls it, for example by tricking the owner into a transaction.

Remediation: Use msg.sender for access control.

```
Vault.sweep(address) (Vault.sol#20-23) uses tx.origin for authorization: require(bool,string)(tx.origin ==
owner,not owner) (Vault.sol#21)
```

4. Return value of send() is not checked

Medium impact, Medium confidence | detector unchecked-send

`Location: Vault.sol:20-23, Vault.sol:22`

What it means: send() returns false on failure instead of reverting. If the result is ignored, the contract may continue as if the payment succeeded.

Remediation: Check the returned bool, or use call{value: ...}("") and require success.

```
Vault.sweep(address) (Vault.sol#20-23) ignores return value by to.send(address(this).balance)
(Vault.sol#22)
```

5. Address not checked for zero

Low impact, Medium confidence | detector missing-zero-check

`Location: Vault.sol:20, Vault.sol:22`

What it means: An address parameter is used without checking it is not address(0).

Remediation: Add require(addr != address(0)) where a zero address would be a mistake.

```
Vault.sweep(address).to (Vault.sol#20) lacks a zero-check on :?        - to.send(address(this).balance)
(Vault.sol#22)
```

6. Low-level call used

Informational impact, High confidence | detector low-level-calls

`Location: Vault.sol:13-18, Vault.sol:15`

What it means: A low-level call bypasses some compiler checks; this is informational.

Remediation: Make sure the result is checked and the call is necessary.

```
Low level call in Vault.withdraw() (Vault.sol#13-18):?    - (ok,None) = msg.sender.call{value: amt}()
(Vault.sol#15)
```

7. Compiler version note

Informational impact, High confidence | detector solc-version

`Location: Vault.sol:3`

What it means: The pragma allows compiler versions with known issues, or is very broad.

Remediation: Pin a recent, well-tested compiler version.

```
Version constraint ^0.8.20 contains known severe issues
(https://solidity.readthedocs.io/en/latest/bugs.html)?    - VerbatimInvalidDeduplication?    -
FullInlinerNonExpressionSplitArgumentEvaluationOrder?    - MissingSideEffectsOnSelectorAccess.?It is used
by:?    - ^0.8.20 (Vault.sol#3)
```

8. Variable could be immutable

Optimization impact, High confidence | detector immutable-states

`Location: Vault.sol:7`

What it means: A state variable is only set in the constructor and could be declared immutable to save gas.

Remediation: Declare it immutable.

```
Vault.owner (Vault.sol#7) should be immutable
```