THREE SIGMA

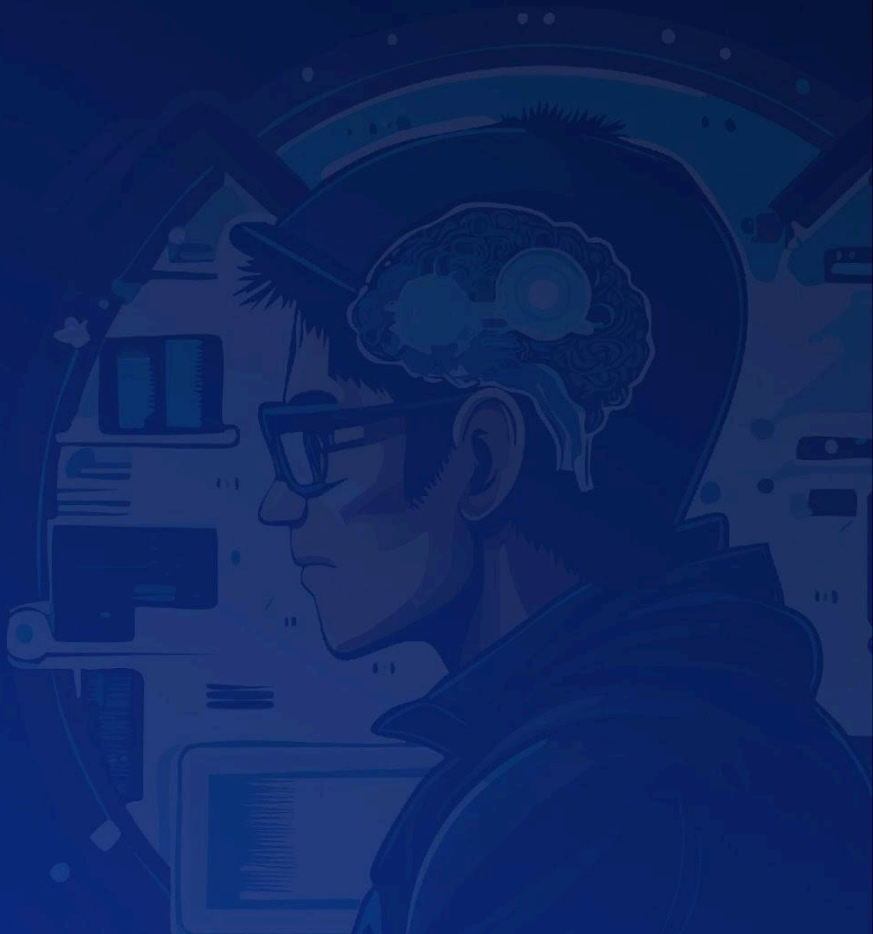
Keyring Sidepocket hook

Security Review

Disclaimer

Security Review

Keyring Sidepocket hook



Disclaimer

The ensuing audit offers no assertions or assurances about the code's security. It cannot be deemed an adequate judgment of the contract's correctness on its own. The authors of this audit present it solely as an informational exercise, reporting the thorough research involved in the secure development of the intended contracts, and make no material claims or guarantees regarding the contract's post-deployment operation. The authors of this report disclaim all liability for all kinds of potential consequences of the contract's deployment or use. Due to the possibility of human error occurring during the code's manual review process, we advise the client team to commission several independent audits in addition to a public bug bounty program.

Table of Contents

Security Review

Keyring Sidepocket hook

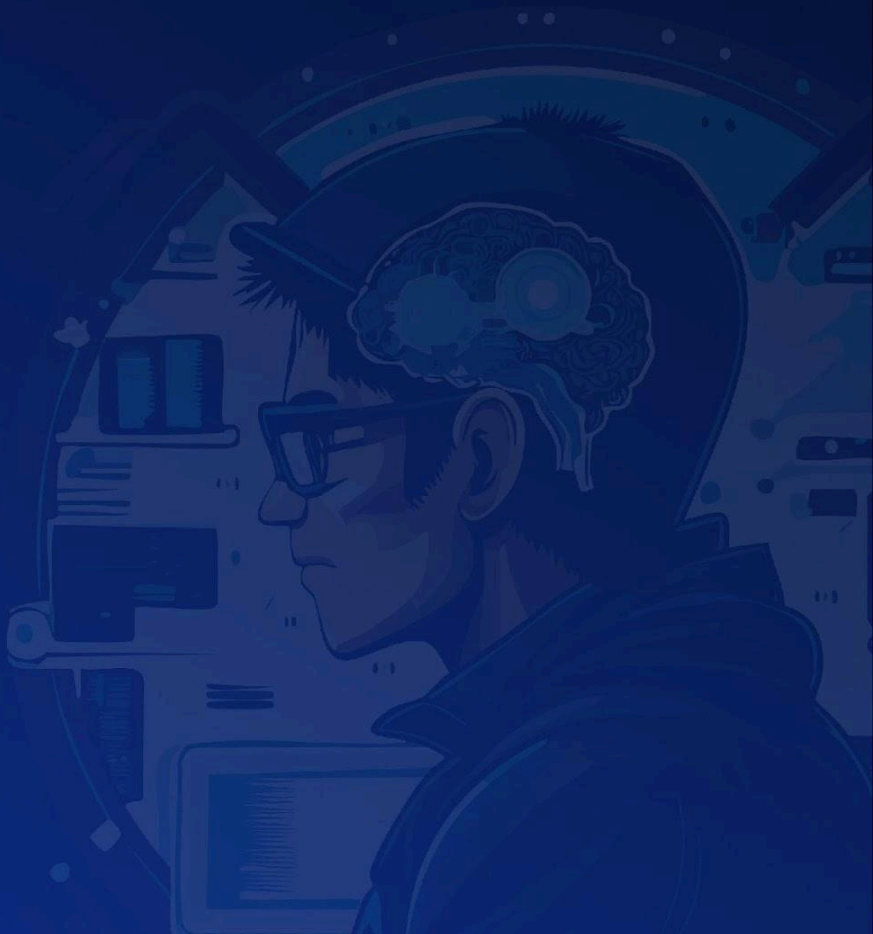
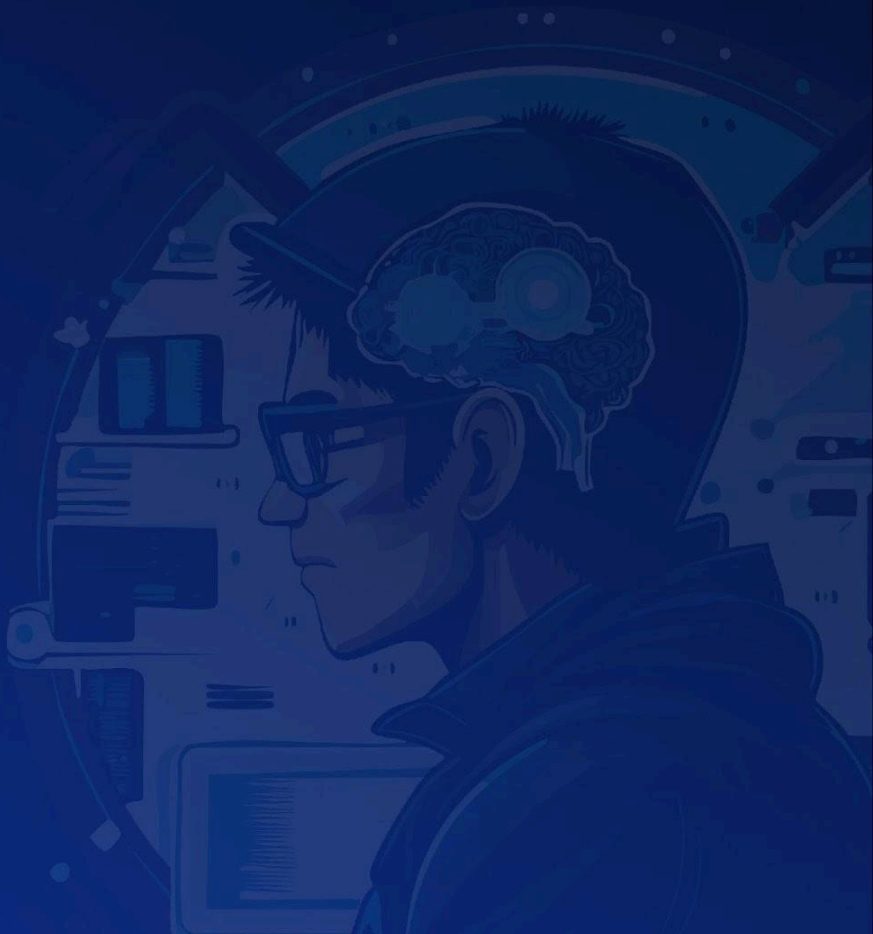


Table of Contents

Summary

Security Review

Keyring Sidepocket hook



Summary

Three Sigma audited Keyring in a 0.4 person week engagement. The audit was conducted from 11/11/2025.

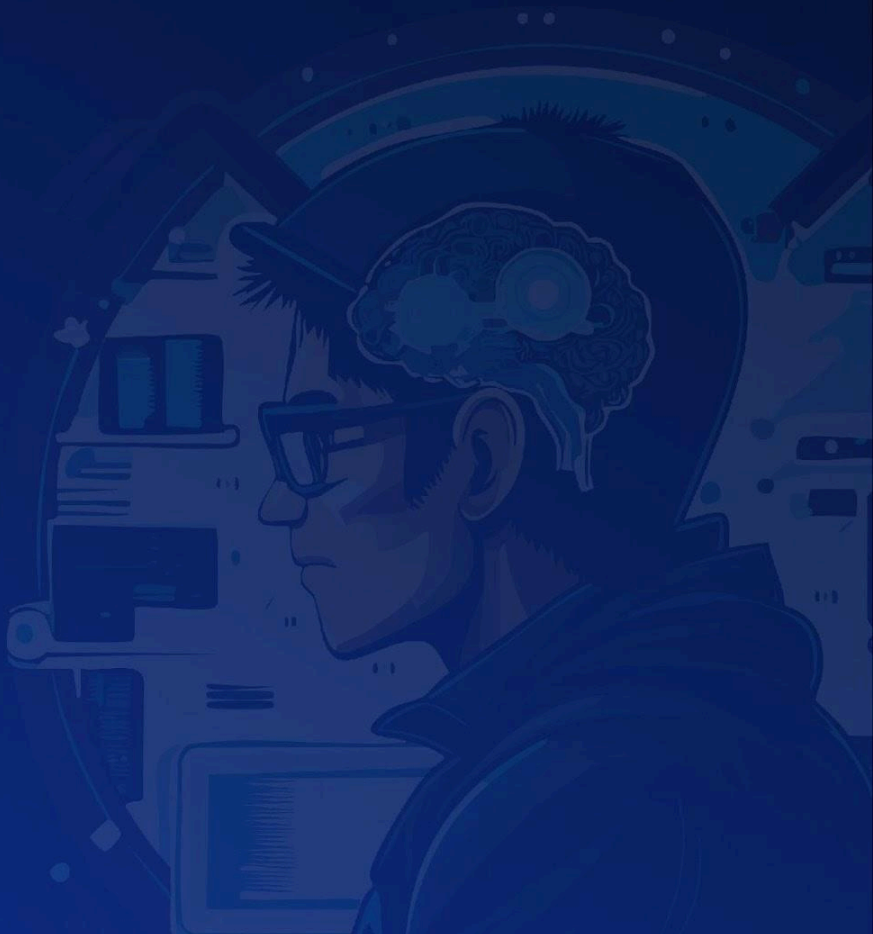
Protocol Description

Euler Keyring Side Pocket Hook is a hook contract for Euler vaults that adds side-pocket functionality with pro-rata withdrawal limits. It enforces keyring-based access control and computes per-user withdrawal allowances via a cumulative withdrawal liquidity index set by administrators. The hook intercepts withdraw and redeem vault operations to authenticate users and enforce limits, tracks cumulative user withdrawals, and disables share transfers to preserve position integrity.

Scope

Security Review

Keyring Sidepocket hook



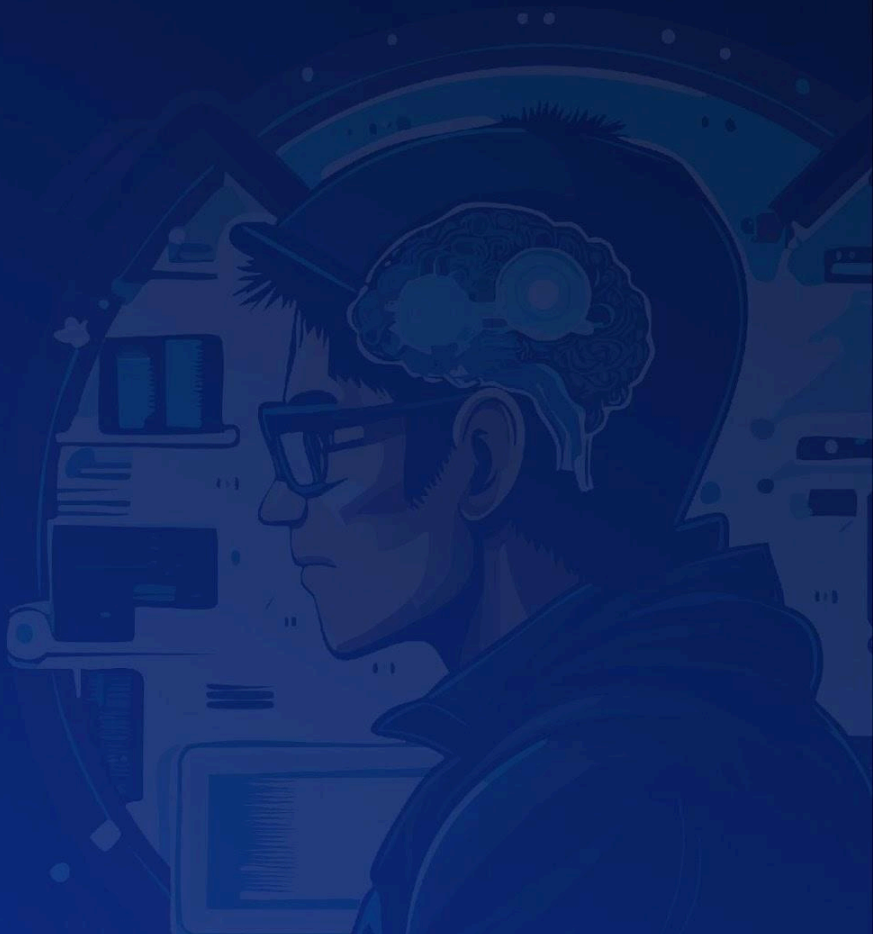
Scope

Filepath	nSLOC
src/HookTarget/HookTargetAccessControlKeyringSidePocket.sol	80
src/HookTarget/HookTargetAccessControlKeyring.sol	74
Total	**154**

Methodology

Security Review

Keyring Sidepocket hook



Methodology

To begin, we reasoned meticulously about the contract's business logic, checking security-critical features to ensure that there were no gaps in the business logic and/or inconsistencies between the aforementioned logic and the implementation. Second, we thoroughly examined the code for known security flaws and attack vectors. Finally, we discussed the most catastrophic situations with the team and reasoned backwards to ensure they are not reachable in any unintentional form.

Taxonomy

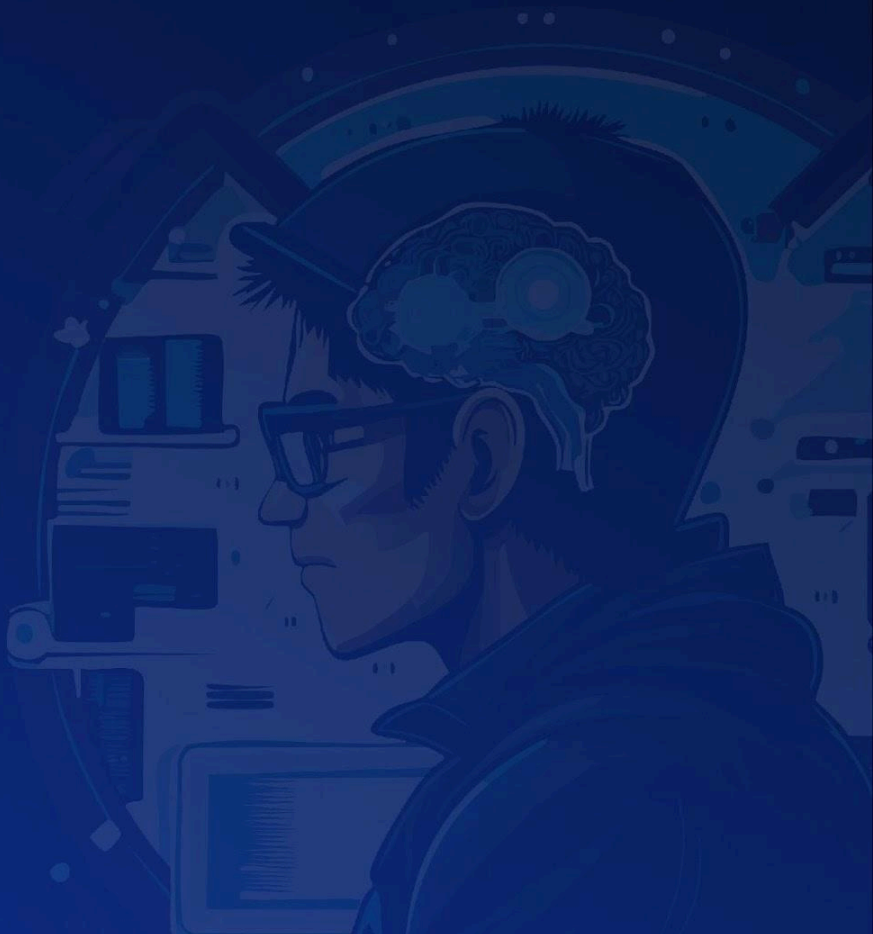
In this audit, we classify findings based on Immunefi's Vulnerability Severity Classification System (v2.3) as a guideline. The final classification considers both the potential impact of an issue, as defined in the referenced system, and its likelihood of being exploited. The following table summarizes the general expected classification according to impact and likelihood; however, each issue will be evaluated on a case-by-case basis and may not strictly follow it.

Impact / Likelihood	LOW	MEDIUM	HIGH
NONE	None		
LOW	Low		
MEDIUM	Low	Medium	Medium
HIGH	Medium	High	High
CRITICAL	High	Critical	Critical

Project Dashboard

Security Review

Keyring Sidepocket hook



Project Dashboard

Application Summary

Name	Keyring
Repository	https://github.com/Keyring-Network/euler-keyring-sidepocket-hook
Commit	ad19951
Language	Solidity
Platform	Avalanche

Engagement Summary

Timeline	11/11/2025
№ of Auditors	2
Review Time	0.4 person weeks

Vulnerability Summary

Issue Classification	Found	Addressed	Acknowledged
Critical	0	0	0
High	0	0	0
Medium	0	0	0
Low	0	0	0
None	1	1	0

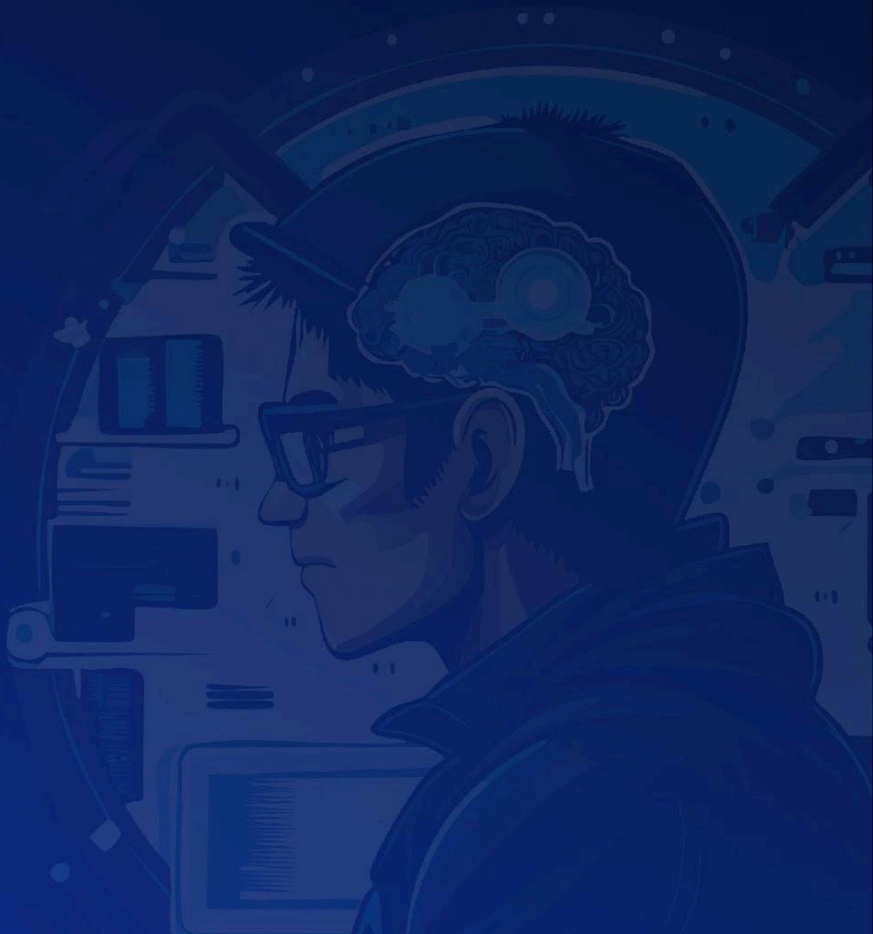
Category Breakdown

Suggestion	1
Documentation	0
Bug	0
Optimization	0
Good Code Practices	0

Risk Section

Security Review

Keyring Sidepocket hook



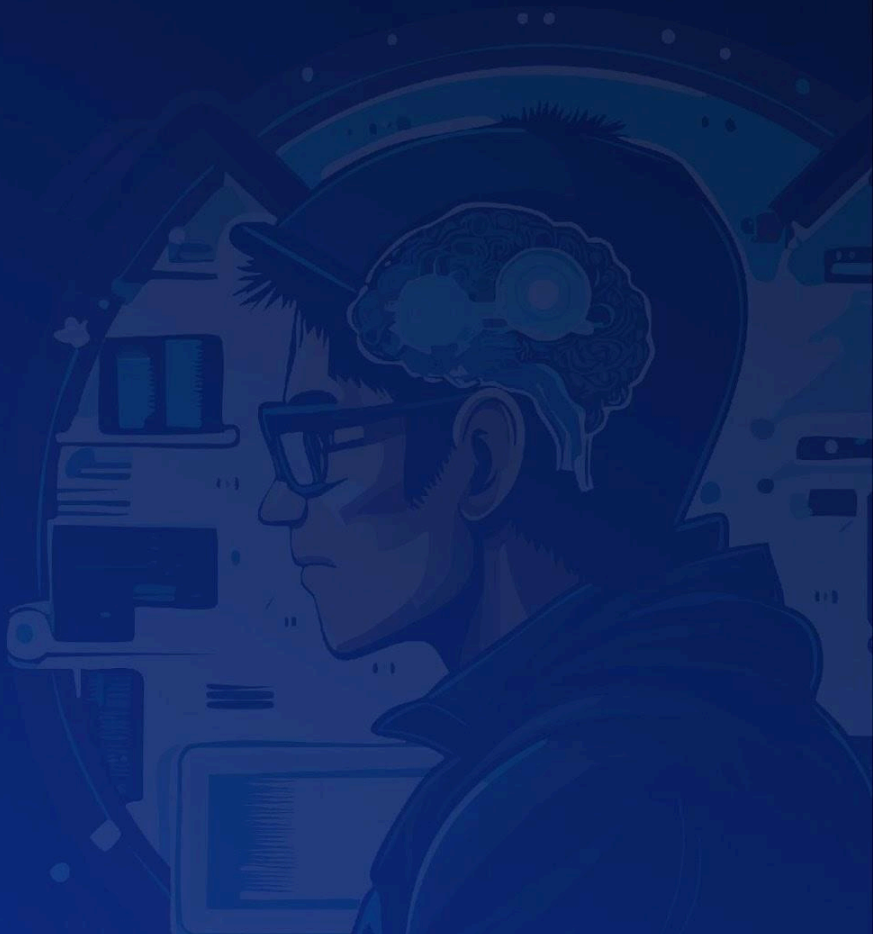
Risk Section

No risks were identified.

Findings

Security Review

Keyring Sidepocket hook



Findings

3S-Keyring-N01

Missing only-vault gating leads to direct calls consuming the caller's withdrawal allowance

Id	3S-Keyring-N01
Classification	None
Category	Suggestion
Status	Addressed in #f2f462f.

Description

HookTargetAccessControlKeyringSidePocket::withdraw and **HookTargetAccessControlKeyringSidePocket::redeem** are intended to be invoked by an EVault via the Euler hook, enforcing Keyring access control and pro-rata withdrawal limits. Currently these entrypoints are externally callable and do not restrict **msg.sender** to a **GenericFactory** proxy. Since **BaseHookTarget::_msgSender** returns the raw **msg.sender** when the caller is not a factory proxy, a legitimate, credentialed user can mistakenly call the hook directly, pass **_authenticateCallerAndAccount**, and increment their own **userWithdrawnAmounts** without any corresponding vault asset movement, because the call did not originate from the vault. The state change persists and reduces their remaining allowance, causing subsequent legitimate withdrawals via the vault to revert. This is a self-DoS caused by user misinteraction rather than an adversarial exploit.

Recommendation

Restrict **withdraw** and **redeem** functions to be callable only by EVault proxies by enforcing **eVaultFactory.isProxy(msg.sender)**.