Permit DOS Vulnerability

On 15 January 2024 Linus (oot2k) disclosed a low severity DOS vulnerability in the EVM implementation of the EUROe stablecoin to Membrane Finance.

The Team reacted quickly, and after a short discussion the issue was confirmed. Membrane already had security measurements in place, no user funds were in danger at any point in time.

Description

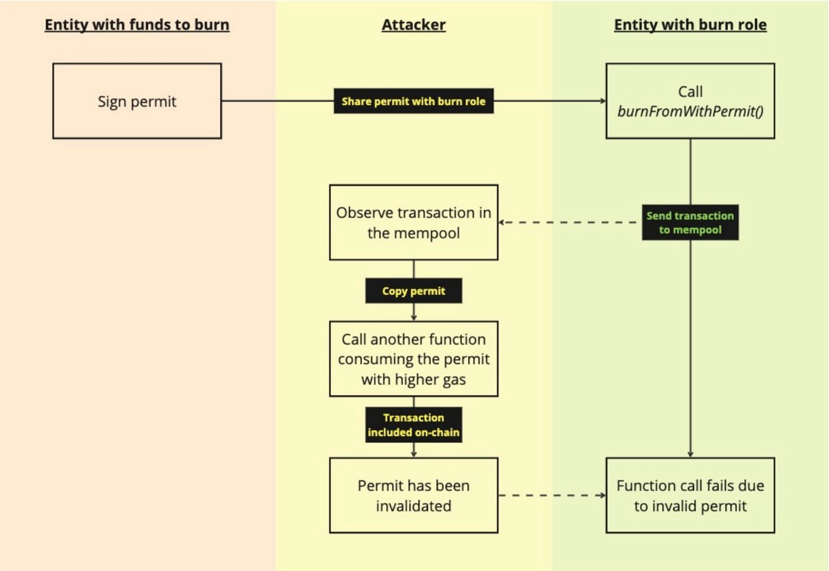
The EUROe ERC20 token incorporates the EIP-2614 permit, enabling users to off-chain sign permits for their tokens. These permits can then be handed to a service for execution, facilitating gasless transfers. The permit() function, accepting a user's signature as input, is published in the blockchain mempool when the transaction is initiated.

Due to the open Zeppelin design choice allowing third-party services to execute user token transfers, any entity can execute the permit() function, potentially leading to frontrunning. Frontrunning involves reading a transaction in the Ethereum mempool, duplicating it, and executing it ahead of the original transaction.

While frontrunning itself does not pose a direct threat, as it doesn't alter the final state, issues arise when calling permit() within another function that modifies the state post-permit. If an attacker successfully frontruns the permit and executes it before the original transaction, the permit in the original transaction is already consumed, resulting in a revert. This causes the entire original transaction to revert.

The EUROe token introduces the burnFromWithPermit() function, which initiates the permit first and then performs the burnFrom() operation. This setup creates vulnerability to frontrunning, as an attacker can exploit the burnFromWithPermit() call, causing it to revert. This leads to a brief Denial of Service (DOS) situation and potential off-chain complications.

Refer to this graph for a visual representation of the steps involved in causing the burnFromWithPermit call to revert.


Entity with funds to burn
Attacker
Entity with burn role
Sign permit
Share permit with burn role
Call burnFromWithPermit()
Observe transaction in the mempool
Send transaction to mempool
Copy permit
Call another function consuming the permit with higher gas
Transaction included on-chain
Permit has been invalidated
Function call fails due to invalid permit

(made by Membrane Finance after disclosure)

Impact

Although this attack enables a Denial of Service (DOS) on the burnFromWithPermit() call, alternative functions like burn() and burnFrom() provide manual token burning options. Incorporating a private RPC adds an extra layer of security, and the implementation of fallback loops further mitigates potential issues in case burnFromWithPermit() encounters a revert.

Fortunately, there was no risk to user funds, and EUROe has already integrated fallback loops, minimizing the current impact of this issue.

It remains crucial to be mindful of this, particularly when deploying your own EUROe.b tokens and creating bridged tokens.

Bug bounty

Based on Immunefi severity standards, this attack falls under the low category.

Membrane Finance reacted fast and professionally, and resolved the issues 2 days after disclosure.
They added a detailed explanation and warning in their documentation.

References

The article that acted as inspiration:
https://www.trust-security.xyz/post/permission-denied

EUROe documentation:
https://dev.euroe.com/docs/BridgedEUROe/developers