Gasless Transaction Forwarder – Team seventhsense

1. Overview

The Gasless Transaction Forwarder is a dApp that allows users to send ERC-20 and ERC-721 tokens without the need for ETH in gas fees. It employs a meta-transaction pattern whereby a relayer pays the gas fees on behalf of the user. *(Refer the youtube Video for more exact details - https://youtu.be/HVf_lA9WEp0)*

2. System Architecture

Components

1. *Smart Contract (Forwarder):*
    - Handles gasless transactions.
    - Verifies user signatures.
    - Executes token transfers.
2. *Frontend Interface:*
    - Web-based UI for user interaction.
    - Connects to MetaMask for signing.
    - Displays transaction history.
3. *Relayer:*
    - Will submit signed transactions to the blockchain.
    - Will pay gas fees for users.

Workflow

1) User signs a transaction request off-chain.
2) Relayer submits the signed request to the Forwarder contract.
3) Contract verifies the signature and executes the transfer.
4) Relayer pays the gas fees.

3. Smart Contract

Contract Details

- **Name: GaslessForwarder**

Solidity Version: ^0.8.0

- *Features:*
    - Supports ERC-20 and ERC-721 transfers.
    - EIP-712 compliant signature verification.
    - Replay protection using nonces.
    - Deadline-based request expiration.

Key Functions

1. *execute:*
    - Verifies the user's signature.
    - Will execute the token transfer.
    - Then update the user's nonce.
2. *nonces:*
    - Tracks the number of transactions per user to prevent replay attacks.

4. Frontend Interface

Technologies Used

- ✓ HTML/CSS: For UI structure and styling.
- ✓ JavaScript: For interaction logic.
- ✓ Ethers.js: For blockchain interaction.
- ✓ MetaMask: For wallet connection and signing.

Key Features

1. *Wallet Connection:*
    - Users connect their MetaMask wallet.
    - Automatically detects network and account changes.
2. *Transaction Form:*
    - Dynamic form for ERC-20/ERC-721 selection.
    - Input validation for addresses and amounts.
3. *Transaction History:*
    - Displays recent transactions with links to Etherscan (or SapoliEtherscan).
4. *Error Handling:*
    - Displays user-friendly error messages.
    - Handles network and signature errors.

5. Relayer

Role

- Has to submit signed transactions to the blockchain.
- Pays gas fees on behalf of users.

Implementation

- Can be implemented as a backend service or a trusted third party.

- Requires ETH balance to cover gas fees (The relayer here will act as the account who will cover the gas fees, and not the depositor or depositee)

6. Security

Smart Contract Security

- Replay Protection: Nonces prevent reuse of signed messages.
- Signature Verification: ECDSA ensures only valid requests are executed.
- Deadline Enforcement: Transactions expire after a set time.

Frontend Security

- HTTPS: Required for secure communication with MetaMask.
- Input Validation: Prevents invalid or malicious inputs.
- Content Security Policy (CSP): Protects against XSS attacks.

7. Deployment

Smart Contract

1. Compile the contract using solc or Hardhat.
2. Deploy to Ethereum mainnet or testnet.
3. Verify the contract on Etherscan.

Frontend

1. Host on a static web server (e.g., GitHub Pages, Netlify).
2. Ensure HTTPS is enabled.
3. Update the contract address and ABI in the frontend code.

8. Usage Guide

1. Connect your MetaMask wallet.
2. Select the token type (ERC-20 or ERC-721).
3. Enter the token address, amount/token ID, and recipient address.
4. Sign the transaction request.
5. Wait for the relayer to submit the transaction.

A1. Troubleshooting

Common Issues

1. *Wallet Not Connecting:*
    - *Ensure MetaMask is installed and unlocked.*
    - *Check network connection.*
2. *Transaction Fails:*
    - *Verify token approval.*
    - *Ensure the relayer has sufficient ETH.*

3. *Invalid Signature:*
    - *Check the signing process.*
    - *Ensure the correct contract address and chain ID.*

A2. References

1. EIP-712: Typed Structured Data Hashing
2. Ethers.js Documentation
3. MetaMask Developer Docs
4. OpenZeppelin Contracts

Team Lead –

Adwait Bhardwaj (adwait.mitblr2024@learner.manipal.edu *OR* cdtadwaitbhardwaj@outlook.in)

Team Members –

Rushil Bakori (rushil.mitblr2024@learner.manipal.edu)
Arjit Menon (arjit.mitblr2024@learner.manipal.edu)
Shreyansh Avasthi (shreyansh1.mitblr2024@learner.manipal.edu)
Raghav Dubey (raghav2.mitblr2024@learner.manipal.edu)