Transfer USDC on testnet from Ethereum to Avalanche

Explore this script to transfer USDC on testnet between two EVM-compatible chains.Suggest Edits

To get started with CCTP, follow the example script providedhere . The example usesweb3.js to transfer USDC from an address on Ethereum Sepolia testnet to another address on Avalanche Fuji testnet.

Interactive Web3 Services Tutorial

If you are new to building smart contracts, check out ourinteractive tutorial where you can transfer USDC using our Web3 Services Smart Contract Platform and Programmable Wallets. The script has 5 steps:

1. In this first step, you initiate a transfer of USDC from one blockchain to another, and specify the recipient wallet address on the destination chain. This step approves Ethereum SepoliaTokenMessenger
2. contract to withdraw USDC from the provided Ethereum Sepolia address.

JavaScript const approveTx = await usdcEthContract.methods.approve(ETH_TOKEN_MESSENGER_CONTRACT_ADDRESS, amount).send({gas: approveTxGas}) 1. In this second step, you facilitate a burn of the specified amount of USDC on the source chain. This step executesdepositForBurn 2. function on the Ethereum Sepolia TokenMessenger contract deployed inSepolia testnet 3. .

JavaScript const burnTx = await ethTokenMessengerContract.methods.depositForBurn(amount, AVAX_DESTINATION_DOMAIN, destinationAddressInBytes32, USDC_ETH_CONTRACT_ADDRESS).send(); 1. In this third step, you make sure you have the correct message and hash it. This step extractsmessageBytes 2. emitted byMessageSent 3. event fromdepositForBurn 4. transaction logs and hashes the retrievedmessageBytes 5. using thekeccak256 6. hashing algorithm.

JavaScript const transactionReceipt = await web3.eth.getTransactionReceipt(burnTx.transactionHash); const eventTopic = web3.utils.keccak256('MessageSent(bytes)') const log = transactionReceipt.logs.find((l) => l.topics[0] === eventTopic) const messageBytes = web3.eth.abi.decodeParameters(['bytes'], log.data)[0] const messageHash = web3.utils.keccak256(messageBytes); 1. In this fourth step, you request the attestation from Circle, which provides authorization to mint the specified amount of USDC on the destination chain. This step polls the attestation service to acquire the signature using themessageHash 2. from the previous step.

JavaScript let attestationResponse = {status: 'pending'}; while(attestationResponse.status != 'complete') { const response = await fetch(https://iris-api-sandbox.circle.com/attestations{messageHash}); attestationResponse = await response.json() await new Promise(r => setTimeout(r, 2000)); } 1. In this final step, you enable USDC to be minted on the destination chain. This step calls thereceiveMessage 2. function on Avalanche FujiMessageTransmitter 3. contract to receive USDC at Avalanche Fuji address.

Note: The attestation service is rate-limited. Please limit your requests to less than 10 per second.

JavaScript const receiveTx = await avaxMessageTransmitterContract.receiveMessage(receivingMessageBytes, signature); Updatedabout 2 months ago