It is quite simple: you deposit using a hash, and later withdraw to any address by providing the original message of the hash. On classic EVM chains, this simple method doesn't guarantee privacy because everything is public. We can easily link depositors and withdrawers by hashing the message provided during withdrawal and finding the corresponding deposit transaction.

However, with SUAVE, we can perform confidential computation and storage, allowing us to secretly verify if the provided original message matches a deposit transaction without leaking the message.

code at blindlycash-toliman/src/BlindlyCash.sol at main · DiveInto/blindlycash-toliman · GitHub

a working demo at https://toliman.blindly.cash

PS1: an implementing detail worth noting is that: when doing withdraw, the address you want to send coin to often has no ETH to begin with, so it can't send the withdraw tx itself. We introduce a middleman to do the task, but of course, we don't want to naively give the origin message to the middleman. Instead we give the encrypted version of the message. That's why the smart contract holds an RSA key pair(we keep the private key in the confidential store). We do the decryption in an off-chain function and checks if the provided message has a corresponding deposit. If it does, we call the on-chain function to redeem.

PS2: on Ethereum, I implemented a similar service using blind signature, The TLDR is that this trick let you make people sign things without knowing the content. So I made a signing service do this blind signing when you deposit, and you can withdraw by showing the service the content and signature. Since the signer service doesn't know what it has signed, it cannot link the deposit and the withdrawal, hence privacy achieved(for more tech details, you can check this tweet which inspired me in the first place), it feels like TornodoCash without the zk, and since it is simply ETH transfer with some calldata, it is way cheaper than TornodoCash, the down side? It is a centralized service since it relies on a central role to do the signing. When I implemented this, I wish so badly that Ethereum can have some way to store secrets so I can make this centralize signing role a smart contract, but in no way this can be true, unless we applied ZK tech, which would basically turn it into TornadoCash. and when I read about SUAVE, I suddenly realized my wish had come true. That is what inspired me to create this demo.