pEthereum Specifications

Hey guys, Andrey is here

I would like to present you our latest work on the privacy mode for the smart-contracts and integration with major DeFi projects.

Any feedback is valuable. Looking forward for it.

Building privacy-protecting decentralized applications, smart contracts, and cryptoassets.

What is pEthereum?

The Ethereum smart contract platform offers an entirely new programming paradigm. It enables developers all over the world to collectively build a new kind of global financial infrastructure without the need for central authorities. Arguably, privacy concerns play a part in discouraging adoption beyond the crypto niche. Everyday users hesitate to reveal how much DAI they're saving, how profitable their Uniswap trades are, or how often they borrow on Compound.

What is needed is incognito mode for Ethereum smart contracts.

pEthereum is an extension of Ethereum. It enables privacy-protecting Ethereum transactions and privacy-protecting decentralized applications like Uniswap and Compound. Transactions are encrypted using zero-knowledge proofs, allowing users to retain their privacy.

With pEthereum, developers can program smart contracts that are not just decentralized, but also privacy-protecting.

Is pEthereum something people want?

In November 2019, we first proposed the idea to the Ethereum community. Since then, 80+ ERC20 tokens with a total value of $2M+ have been shielded via the Incognito-Ethereum trustless bridge . This is a strong validation that Ethereum users want privacy.

[

Screen Shot 2020-04-22 at 3.58.57 PM

1202×401 37.6 KB

](https://ethresear.ch/uploads/default/original/2X/b/bd17ebfc8a0d677f488f9b8ca093321b6ddf21e0.png)

Source: Etherscan , April 21, 2020

Core concepts

- Cross-Chain Instruction: Incognito communicates with Ethereum via instructions. Instructions are high-level, cross-chain opcodes. An instruction specifies what operation is to be performed by the other chain. There are five instructions: SHIELD, UNSHIELD, DEPLOY, UNDEPLOY, and EXECUTE.
- Bridge: Bridge is a two-way trustless bridge between Incognito and Ethereum. It is responsible for forwarding instructions between the two chains. It consists of multiple relayers. Relayers cannot forge or corrupt the instruction content in any way, because every instruction is cryptographically signed by users and verified on both ends of the bridge.
- Broker: Broker is a smart contract on Ethereum that receives instructions from Incognito, verifies them, and redirects to suitable dApps on Ethereum.
- dApp: A decentralized application (or "dApp") lives on Ethereum. It consists of one or more smart contracts that run exactly as programmed.
- pApp: A privacy-protecting decentralized application (or "pApp") lives on Incognito. It is the privacy-protecting counterpart of an existing dApp on Ethereum.

Cross-Chain Instructions

SHIELD instruction

Shielding is the process of turning a public ERC20 token into its privacy counterpart of the same value. For example, DAI

can be shielded to obtain the privacy coin pDAI. pDAI has the same value as DAI, so 1 pDAI can always be redeemed for 1 DAI and vice versa. Once shielded, privacy coin transactions are confidential and untraceable.

Following is an overview of the SHIELD instruction flow:

[

Screen Shot 2020-04-22 at 6.14.03 PM

785×543 45.8 KB

](https://ethresear.ch/uploads/default/original/2X/c/c03aaa5c7c843f9ee0f3688143ecc2611c39b23a.png)

1. A user deposits some amount of an ERC20 token into the Broker smart contract. Once the transaction is confirmed on Ethereum, the user has a deposit proof.
2. The user sends a SHIELD instruction to Incognito, along with the deposit proof, via the bridge.
3. Incognito validators parse the SHIELD instruction to get the deposit proof, which is verified using Ethereum Simplified Payment Verification (SPV), as well as the minting parameters, which are used to mint the privacy coin counterpart of the same value.

UNSHIELD instruction

Unshielding is the reverse process of shielding: turning privacy coins back into public ERC20 tokens.

Following is an overview of the UNSHIELD instruction flow:

[

Screen Shot 2020-04-22 at 6.14.14 PM

789×543 52.6 KB

](https://ethresear.ch/uploads/default/original/2X/d/d3898e4a93566b398a1cbc0e68ace68f7785c470.png)

1. A user initiates an unshielding transaction on Incognito, with information about which privacy coins they want to unshield and the amount. Once the transaction is confirmed on Incognito, the user has a burn proof.
2. The user sends an UNSHIELD instruction to the Broker smart contract, along with the burn proof, via the bridge.
3. The Broker smart contract parses the UNSHIELD instruction to get the burn proof, which is verified by counting the number of signatures from Incognito validators, as well as the burning parameters, which are used to transfer the public ERC20 tokens back to the user.

DEPLOY instruction

Once shielded, privacy coin transactions are confidential and untraceable. However, they are limited to only basic features like sending and receiving. DEPLOY, EXECUTE, and UNDEPLOY are three instructions that allow users to use their privacy coins in their favorite Ethereum dApps. For example, trade pETH for pDAI on Uniswap, or collateralize pETH to borrow pUSDC on Compound.

Deploying is the process of moving funds from Incognito to Ethereum so that users can spend them in Ethereum dApps.

Following is an overview of the DEPLOY instruction flow:

[

Screen Shot 2020-04-22 at 6.14.25 PM

790×545 54.4 KB

](https://ethresear.ch/uploads/default/original/2X/6/62d5761eff5d5bf56dea2e682ceeec40850fa485.png)

1. A user confidentially initiates a deploy transaction on Incognito with information about which privacy coins they want to deploy and the amount. Once the transaction is confirmed on Incognito, the user has a deploy proof, which is similar to a burn proof.
2. The user sends a DEPLOY instruction to the Broker smart contract, along with the deploy proof, via the bridge.
3. The Broker smart contract parses the DEPLOY instruction to get the deploy proof, which is verified by counting the number of signatures from Incognito validators, and the deploy parameters, which is used to increase the user’s

currently deployed balances.

EXECUTE instruction

Executing is the process of running a function call of an Ethereum smart contract anonymously. For example, running swap(pETH, pDAI) on Uniswap anonymously or borrow(pUSDC) on Compound anonymously.

The following is an overview of the EXECUTE instruction flow:

[

Screen Shot 2020-04-22 at 6.14.41 PM

635×579 47.3 KB

](https://ethresear.ch/uploads/default/original/2X/e/e04a192d76e9b608fb329a6af207549ad7f5ef81.png)

1. A user confidentially signs and sends an EXEC instruction from a pApp on Incognito, with information about which counterpart dApp on Ethereum they want to run and the parameters.
2. The bridge forwards the EXEC instruction to the Broker smart contract.
3. The Broker smart contract parses the EXEC instruction without revealing the user identity, verifies the parameters (especially the amount the user wants to spend against the user balance), and finally sends a message to a suitable smart contract via the encoded ABI.

An EXECUTE instruction contains the following parameters:

- The input token to spend on this transaction
- The input amount of input token to spend on this transaction, which should not exceed the user balance in the Broker smart contract
- The output token if the execution returns one
- The dApp contract address
- The encoded ABI of the target function of the dApp
- The timestamp of the transaction
- The signature on the combined data of the above parameters

UNDEPLOY instruction

Undeploying is the reverse process of deploying: moving funds from Ethereum to Incognito.

Following is an overview of the UNDEPLOY instruction flow:

[

Screen Shot 2020-04-22 at 6.14.56 PM

786×543 71.5 KB

](https://ethresear.ch/uploads/default/original/2X/4/4c332f9eb324e8f582a5c29be77661607d7109a7.png)

1. A user confidentially creates an UNDEPLOY instruction, with information about which privacy coins they want to undeploy and the amount.
2. The bridge forwards the UNDEPLOY instruction to the Broker smart contract.
3. The Broker smart contract parses the UNDEPLOY instruction, verifies the user's signature, and subtracts the user's currently deployed balances. Once the transaction is confirmed on Ethereum, the user has an undeploy proof.
4. The bridge forwards an ACK instruction to Incognito, along with the undeploy proof.
5. Incognito validators parse the ACK instruction to get the undeploy proof, which is verified using Ethereum Simplified Payment Verification (SPV), as well as the undeploy parameters, which are used to mint the privacy coin counterpart and send it to the user.

Timeline

The core team has designed the product strategy around the most popular dApps on Ethereum. We believe that implementing the counterpart pApps for these dApps will provide the most value to Ethereum users.

Date

pApp

Developer Tools

NOV 2019

Incognito Wallet

SHIELD instruction

UNSHIELD instruction

MAY 2020

pKyber

DEPLOY instruction

UNDEPLOY instruction

EXEC instruction

Aug 2020

pCompound

pUniswap

pEthereum SDK

Delayed

pAragon / pMolochDAO

In parallel, we're developing the pEthereum SDK that allows developers to build their own pApps on top of their existing dApps.

Source code

All Incognito development is public. The code is open-source on Github . All progress is viewable via weekly updates on incognito.org . Also, find incognito's monthly updates here.

Conclusion

Crypto's lack of privacy threatens our new economy and slows the adoption of new financial products. We believe that privacy is the missing piece for many everyday users.

We have proposed a way to build privacy-protecting decentralized applications on top of Ethereum. It doesn't make sense to build a new EVM from scratch. Ethereum already has a large developer and user base. By leveraging what Ethereum has done, we can focus on solving the privacy problem. Developers can continue to build dApps on Ethereum using Solidity, and utilize the pEthereum SDK to add incognito mode for their dApps.

We're working on the pEthereum Developer Guide. It will show you how to build a privacy-protecting decentralized application – both from scratch and on top of your existing dApp. Stay tuned!

QR code for the Privacy Quest