ERC-6551 NFT 绑定账户

ERC-6551
NON-FUNGIBLE TOKEN BOUND ACCOUNTS

什么是 ERC-6551

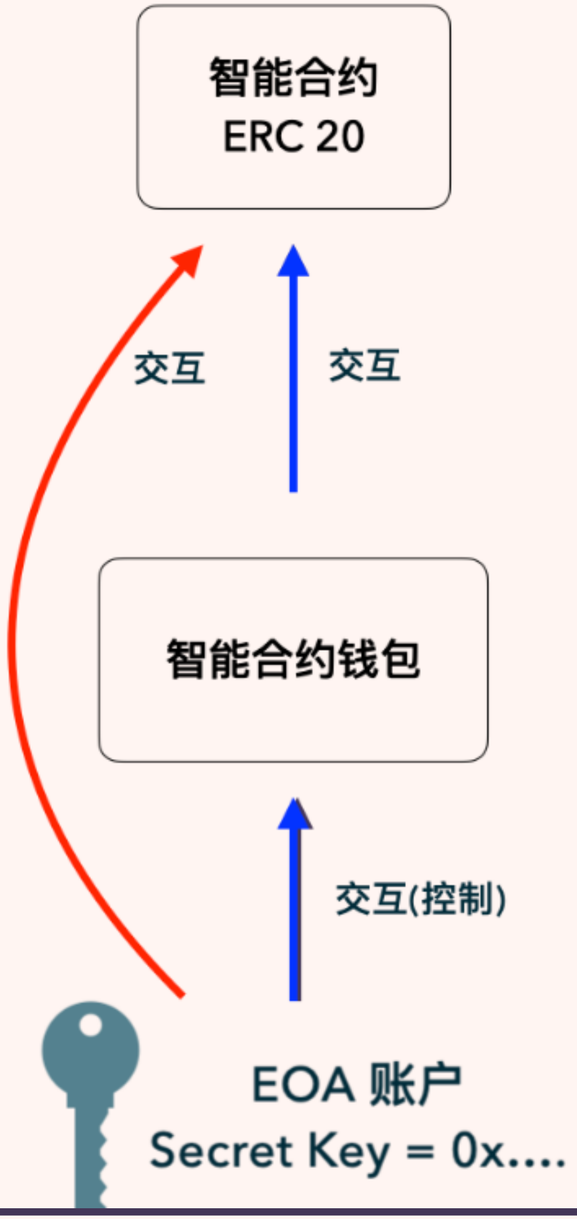
MOTIVATION

- A character in a role-playing game that accumulates assets and abilities over time based on actions they have taken
- An automobile composed of many fungible and non-fungible components
- An investment portfolio composed of multiple fungible assets
- A punch pass membership card granting access to an establishment and recording a history of past interactions

This proposal aims to give every NFT the same rights as an Ethereum user

什么是 ERC 6551

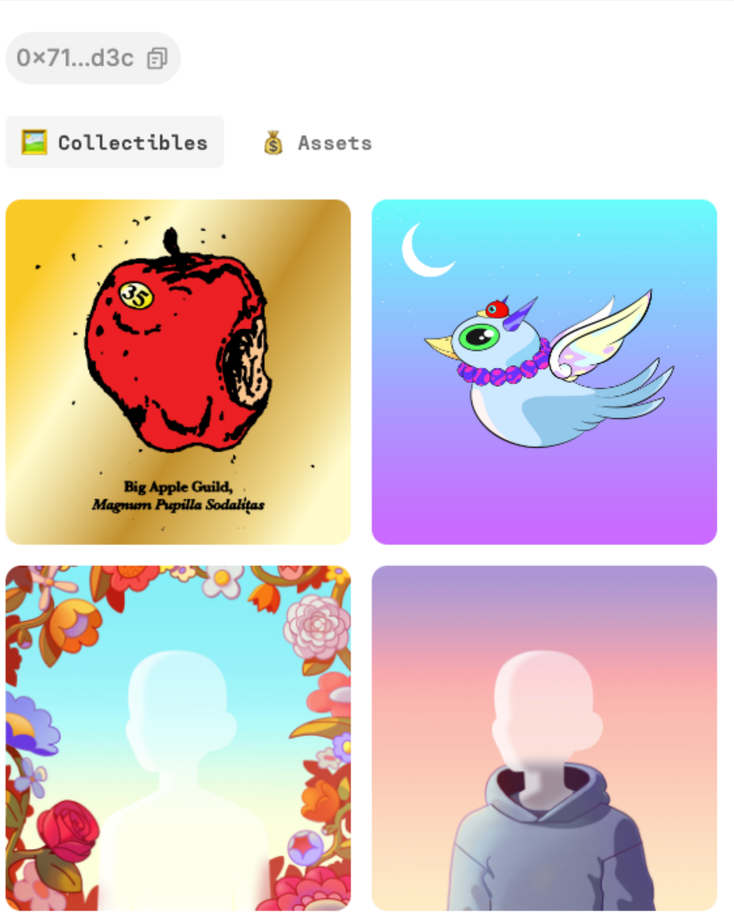
- An interface and registry
  for smart contract accounts owned by non-fungible tokens
- 基于 ERC-721 的协议：
  - 为指定的 NFT 创建关联的智能合约账户(钱包)
  - 创建的账户的调用条件：调用者为 NFT 的所有者
- **无需修改现有 NFT 合约**，增强 NFT 的表达力和可组合性
  - NFT 具有身份 / 为 NFT 创建 Collection / NFT 组合性 / ...



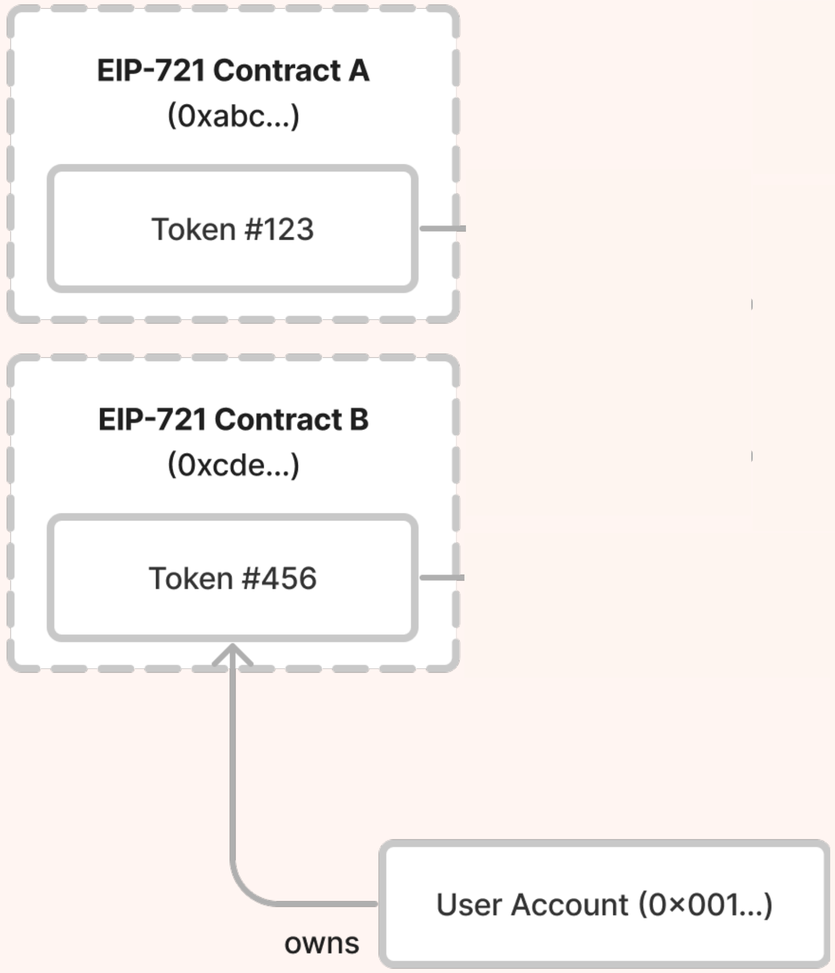
FUTURE PRIMITIVE EXAMPLE

- https://tokenbound.org/assets/ethereum/0x26727Ed4f5BA61d3772d1575Bca011Ae3aEF5d36/1

Sapienz #1



工作原理


EIP-721 Contract A
(0xabc...)
Token #123
EIP-721 Contract B
(0xcde...)
Token #456
User Account (0×001...)
owns

接口：IERC6551REGISTRY

```
// 创建&获取已创建账户
function createAccount(
    address implementation,
    uint256 chainId,
    address tokenContract,
    uint256 tokenId,
    uint256 seed,
    bytes calldata initData
) external returns (address);
function account(
    address implementation,
    uint256 chainId,
    address tokenContract,
    uint256 tokenId,
    uint256 salt
) external view returns (address);
```

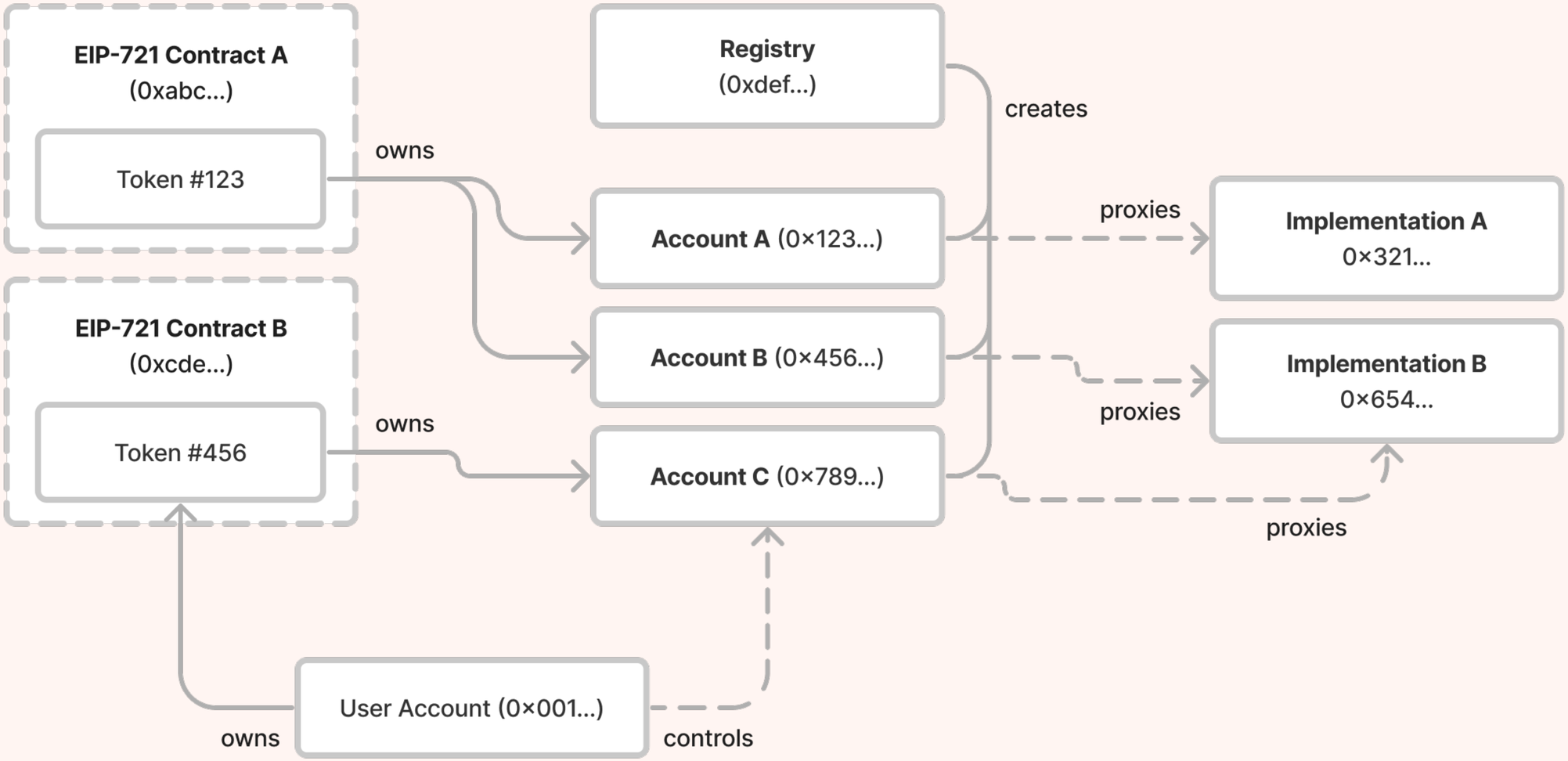
```
// 账户创建 event
event AccountCreated(
    address account,
    address indexed implementation,
    uint256 chainId,
    address indexed tokenContract,
    uint256 indexed tokenId,
    uint256 salt
);
```

接口：IERC6551ACCOUNT

```
receive() external payable;

function state() external view returns (uint256);

function token()
    external
    view
    returns (
        uint256 chainId,
        address tokenContract,
        uint256 tokenId
    );

    * MUST return the bytes4 magic value 0x523e3260
    if the given signer is valid
function isValidSigner(address signer, bytes calldata context)
    external
    view
    returns (bytes4 magicValue);
```

接口：IERC6551EXECUTABLE（非强制）

```
    // 非强制
    * - 0 = CALL
    * - 1 = DELEGATECALL
    * - 2 = CREATE
    * - 3 = CREATE2
    function execute(
        address to,
        uint256 value,
        bytes calldata data,
        uint256 operation
    ) external payable returns (bytes memory);
```


EIP-721 Contract A
(0xabc...)
Token #123
EIP-721 Contract B
(0xcde...)
Token #456
owns
owns
Registry
(0xdef...)
creates
Account A (0×123...)
Account B (0×456...)
Account C (0×789...)
proxies
proxies
proxies
Implementation A
0×321...
Implementation B
0×654...
User Account (0×001...)
owns
controls

简单总结

- Registry 作为工厂合约
  - 可以通过getLogs查询NFT对应的账户
  - 将账户部署到指定地址
- NFT 的 Tokenbound Account（TBA）的控制者为 NFT 的所有者
- NFT 与 TBA 是一对多的关系，允许部署时指定Account的实现合约
- Account 可以具有执行接口，但没有强制的接口约定

扩展空间?

- 多链（空间）支持
  - chainId 参数 & isValidSigner
- 权限支持
  - isValidSigner 返回值
- ERC20 / ERC 1155 / ERC 3525 支持
  - isValidSigner 判定逻辑 -> DAO / 多签 / ...
  - 1155 任何一个持有者都能够调用合约，2个以上持有者同意可以撤销其他人的所有权

```
function token()
    external
    view
    returns (
        uint256 chainId,
        address tokenContract,
        uint256 tokenId
    );
```

```
 * MUST return the bytes4 magic value 0x523e3260
 if the given signer is valid
function isValidSigner(address signer, bytes calldata
    external
    view
    returns (bytes4 magicValue);
```

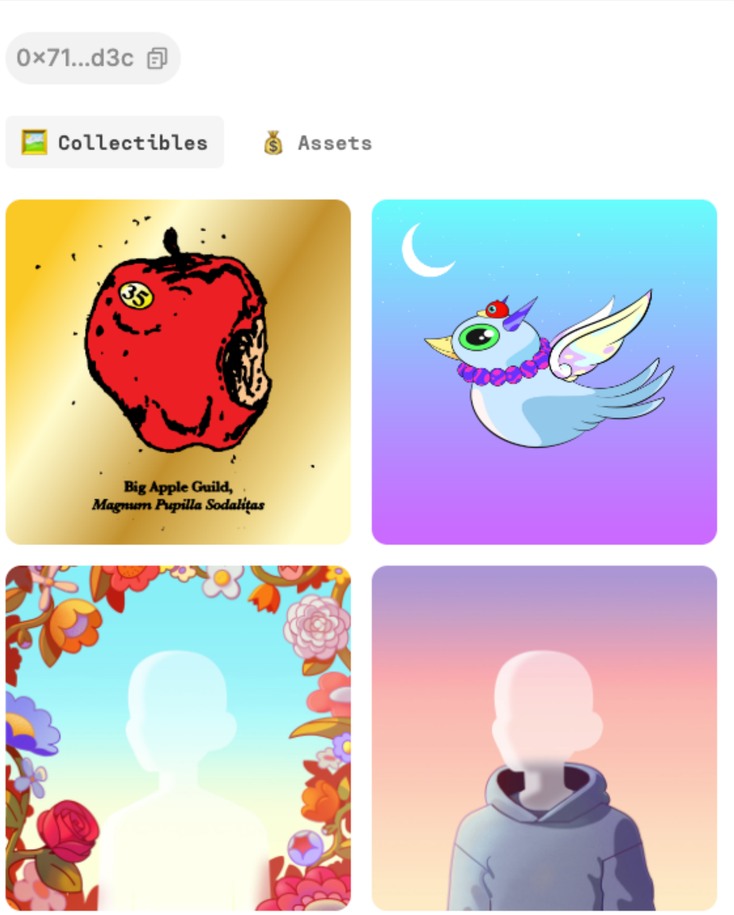
用例

以太坊

FUTURE PRIMITIVE EXAMPLE

- https://tokenbound.org/assets/ethereum/0x26727Ed4f5BA61d3772d1575Bca011Ae3aEF5d36/1

Sapienz #1



ETHGLOBAL WATERLOO 2023

- **AquaNet (https://ethglobal.com/showcase/aquanet-dzz2c)**
  AquaNet 利用非同质化 Token 绑定账户（ERC-6551）来实现 NFT 在基于人工智能驱动的社交媒体网络中的创建和记录活动。每个符合条件的 NFT 都可以在「myPuddle」网络上创建自己的账户，使 NFT 能够建立和发展独特的 AI 身份——主要由其现有特征生成。它们将自主地发布消息到自己的账户、列出自己的兴趣，并分享关于自己的信息。

ETHGLOBAL WATERLOO 2023

- **Fukuro(https://ethglobal.com/showcase/fukuro-3cdwv)**
  该项目利用 EIP-6551 创建了一个拍卖市场。用户可以对 Fukuro 捆绑包进行列出和竞价。

  「Fukuro」的名字灵感来自于日本的福袋，也称为「幸运袋」。它们是神秘的密封袋子，以固定价格出售。EIP-6551 使得 ERC-721 能够控制自己的「钱包地址」并持有资产。

  每个 Fukuro 可以包含任何以太坊地址可以包含的东西，例如其他 NFT、ERC20 等等。这使得交易各种资产成为可能，例如艺术收藏品、交易卡牌套装甚至策划好的投资组合。

ETHGLOBAL WATERLOO 2023

- **Piggybank 6551 NFT (https://ethglobal.com/showcase/piggybank-6551-nft-e2ai5)**
  Piggybank 6551 是一个有趣的 NFT 储蓄账户。用户可以铸造 Piggybank NFT，然后通过可支付函数或直接向 NFT 的 6551 账户地址发送 ETH 来将其装载到 NFT 中。随着每个 Piggybank 中的 ETH 累积，NFT 元数据的颜色、文本标签和属性立即更新。一旦 ETH 锁定在 NFT 中，提取它的唯一方法是销毁 NFT，要么调用 burn 函数，要么手动将 NFT 发送到自己的 6551 账户。当 6551 账户收到 NFT 后，它立即将包含的 ETH 返回给销毁 NFT 的账户，并将元数据更新为「已销毁」

ETHGLOBAL WATERLOO 2023

- **Tokenbound Titans (https://ethglobal.com/showcase/tokenbound-titans-5w6oq)**
  这是一个将 ERC-6551 与动态生成的 NFT 结合在一起的 NPC 游戏实现。玩家是 NFT 的所有者，每个 NFT 拥有不同的能力和特征。

  游戏的核心是这些 NPC 之间的战斗，玩家可以在数字竞技场上押注自己的 Token。计算战斗结果的算法是链下的，并且可以使用 ZK-proof 进行验证。

  赢得战斗可以让玩家的 NPC 升级，增强血量、攻击、速度和护甲等现有属性。自然的延伸是还能解锁新的技能和组合，让玩法更加多样化。随着 NFT 的升级，它们变得更强大、更灵活。

CORE

适合国内的AA / 账户托管 / DID

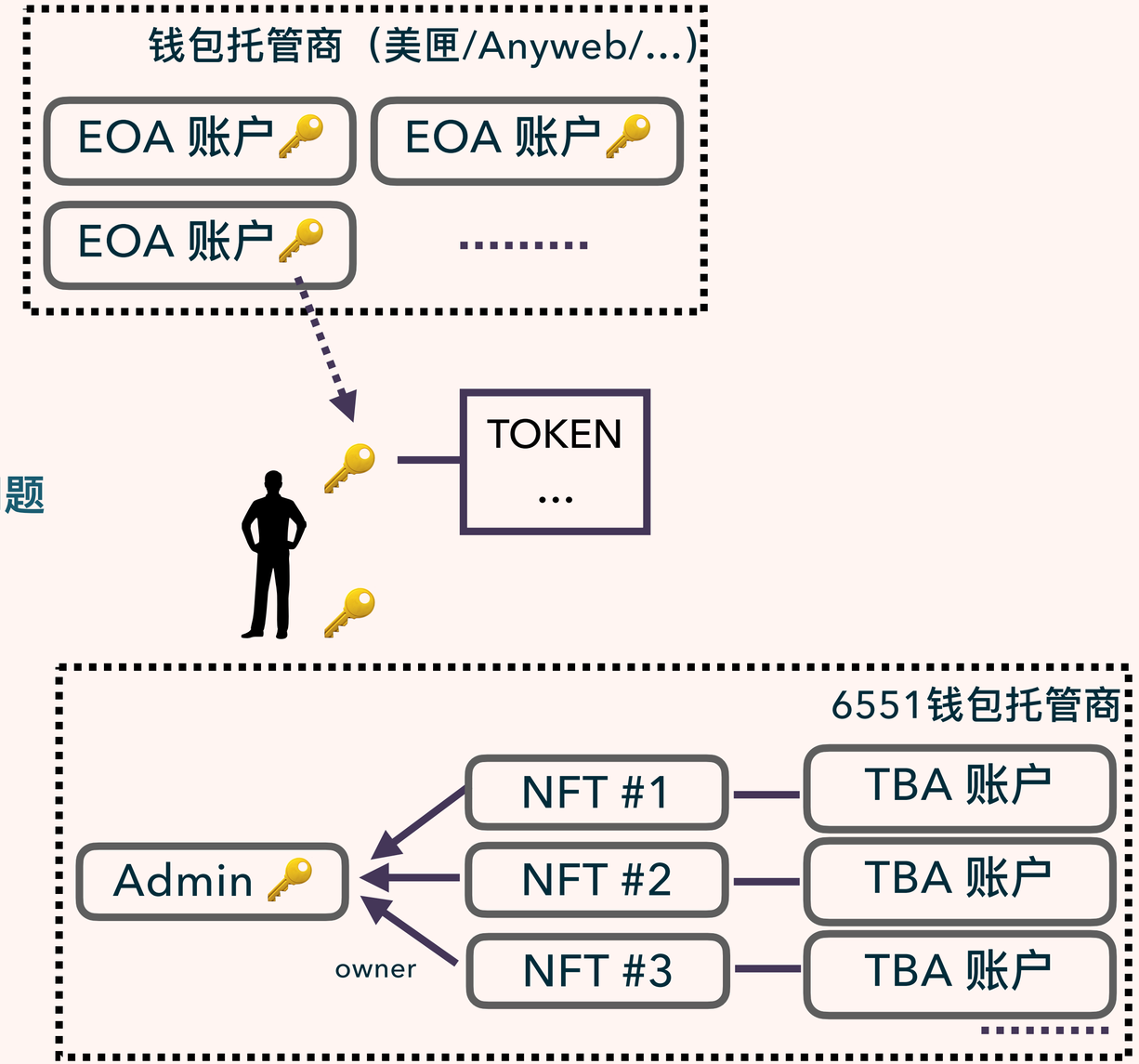
现状

- Core Space 不需要解决 4337 的代付问题
- Core 目前的生态主要基于 NFT
  且 NFT 相关商业活动容易合规展开

优势

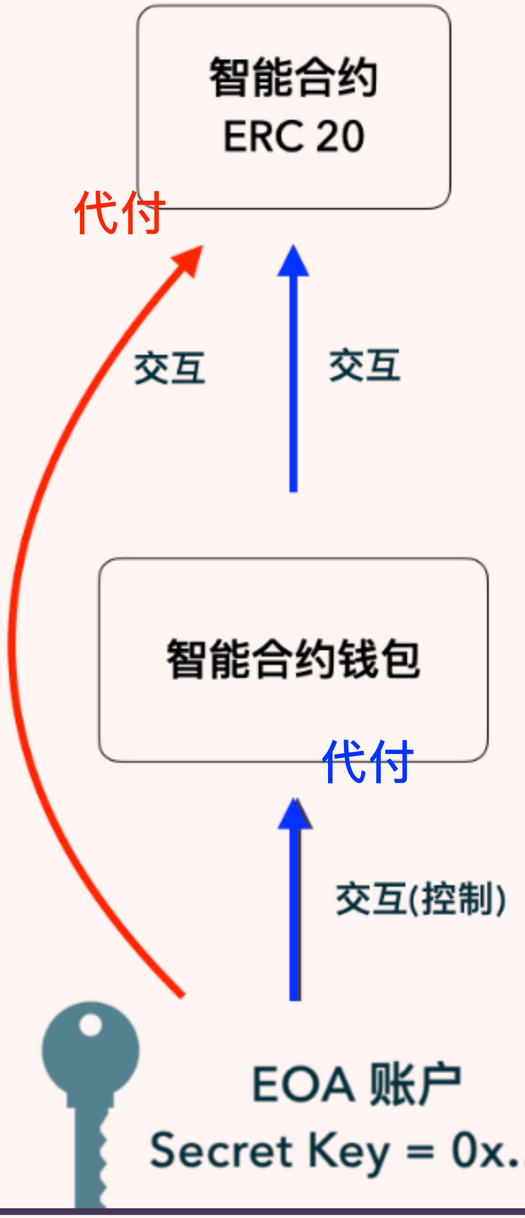
- （大量）私钥的安全存储与使用是一个复杂的问题
- 安全的账户迁移非常复杂

- **AA 钱包的固有优势：表达能力是 EOA 的超集**
  - **更无缝的 web2 -> web3 transition**
  - **绕过私钥托管问题**
- 一切都可以代付



可以开展的服务

- TBA 账户托管 / (D)ID 业务 / CNS 服务
- SCAN 索引服务 （针对复数Registry）
- 代付一切：To C 的代付服务
- NFT with 账户恢复



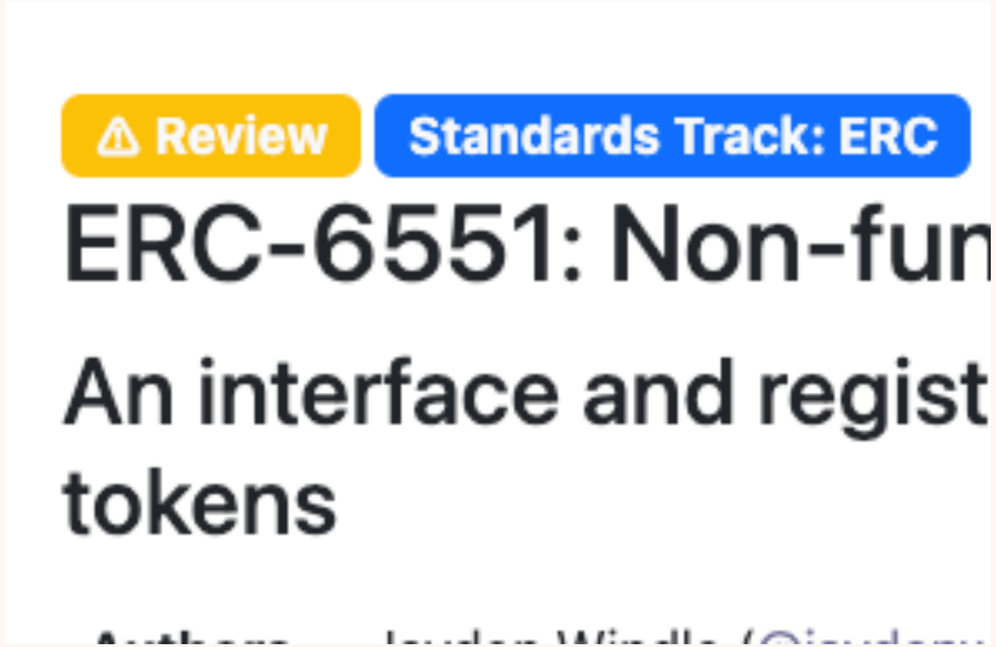
BSIM 卡?

- 初始化：mint 不可手动转移的手机号 NFT + 创建6551账户
- 赚米：电信分销代付
- SIM卡遗失：身份证件补办SIM卡，已铸造NFT的ByAdmin转移 / ...
- 密钥管理：简单

需要解决的问题

6551 仍在REVIEW

- 接口存在调整的可能
- 6551 协议中存在潜在的需要应用层解决的问题
  - Fraud Prevention
  - Ownership Cycles



REGISTRY

- **Singleton （但应该不是问题）**
- **Core 上无法部署到相同地址**

Registry

The ERC-6551 Registry contract has been deployed to the same address across

Note: until ERC-6551 is finalized, the registry address is subject to change.

EVM Network	Registry Address
Goerli	0x02101dfB77FDE026414827Fdc604ddAF224F0921
Sepolia	0x02101dfB77FDE026414827Fdc604ddAF224F0921
Mumbai	0x02101dfB77FDE026414827Fdc604ddAF224F0921
Mainnet	0x02101dfB77FDE026414827Fdc604ddAF224F0921
Polygon	0x02101dfB77FDE026414827Fdc604ddAF224F0921
Optimism	0x02101dfB77FDE026414827Fdc604ddAF224F0921
Arbitrum	0x02101dfB77FDE026414827Fdc604ddAF224F0921
Gnosis Chain	0x02101dfB77FDE026414827Fdc604ddAF224F0921

ACCOUNT IMPLEMENTATION

- 新的 ACCOUNT 实现添加多链（空间）支持

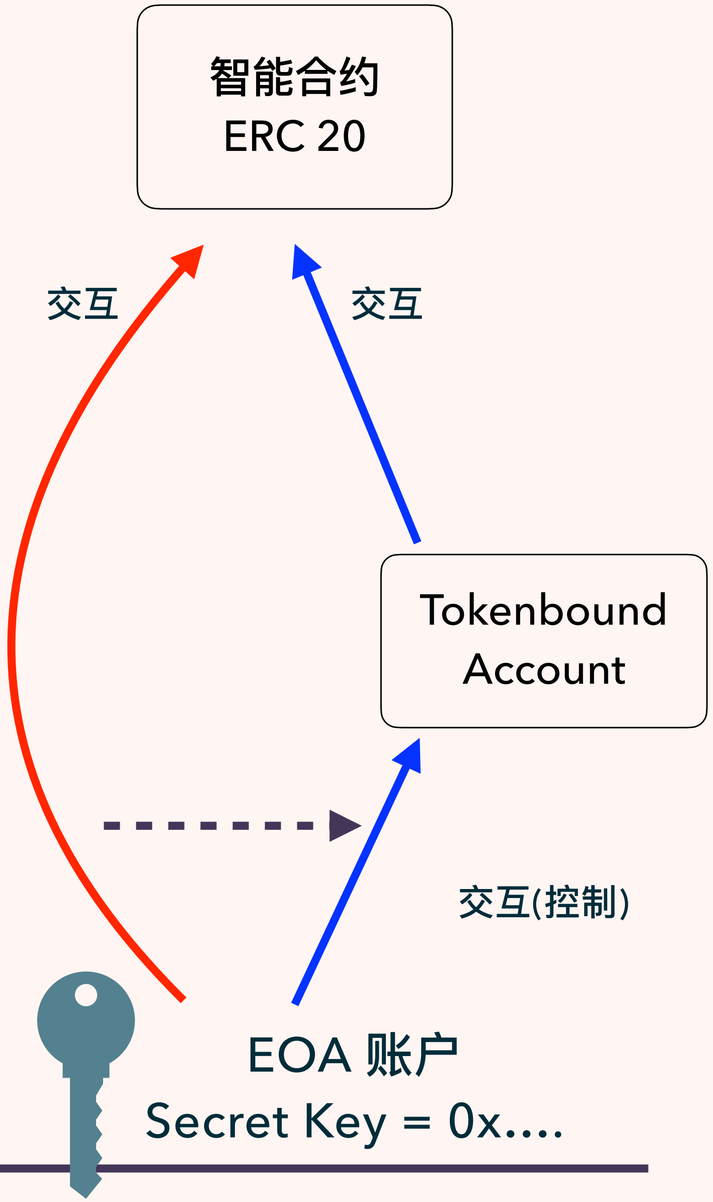
```
// 创建&获取已创建账户
function createAccount(
    address implementation,
    uint256 chainId,
    address tokenContract,
    uint256 tokenId,
    uint256 seed,
    bytes calldata initData
) external returns (address);
```

```
* MUST return the bytes4 magic value 0x523e3260
if the given signer is valid
function isValidSigner(address signer, bytes calldata context)
    external
    view
    returns (bytes4 magicValue);
```

基础设施兼容路线

直接交互
to: (address of ERC20)
data: 0x123... (
  method: transfer
  params:
    to: address A
    value: 10**18
)

经由智能合约钱包交互
to: (address of TBA)
data: 0x456 ... (
  method: execute
  params:
    to: (address of ERC20)
    value: 0
    data: 0x123...
    operation: 0
)

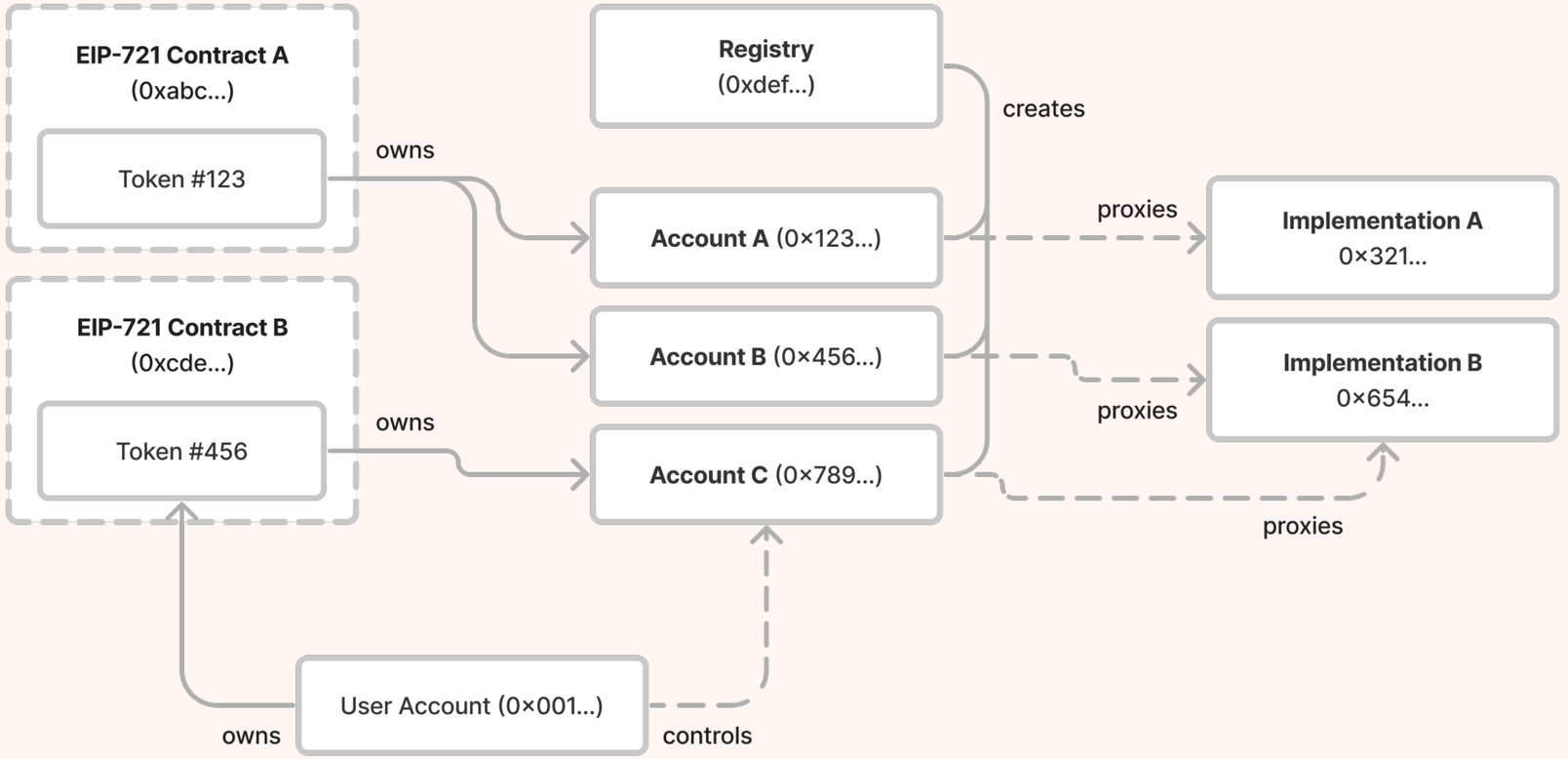


基础设施兼容路线

- 钱包侧：用户使用的钱包本身增加 ERC-6551 支持
  - 现有 dApp 无需修改即可兼容
  - 用户必须使用特定钱包
- dApp 侧：使用SDK封装交易（当前官方提供的参考方法）
  - 无需使用特定钱包
  - 未修改的 dApp 无法支持
- 用户侧：添加额外的适配层（如额外的浏览器插件对现有钱包进行封装）
  - 特定环境下适配所有钱包和dApp
  - 用户需要主动安装；某些环境下不可使用（如移动端）

参考实现

- **Registry**
- **Account (implementation)**



REGISTRY

- ~~createAccount~~ account

```
function account(
    address implementation,
    uint256 chainId,
    address tokenContract,
    uint256 tokenId,
    uint256 salt
) external view returns (address) {
    bytes32 bytecodeHash = keccak256(
        ERC6551BytecodeLib.getCreationCode(
            implementation,
            chainId,
            tokenContract,
            tokenId,
            salt
        )
    );

    return Create2.computeAddress(bytes32(salt), bytecodeHash);
}
```

REGISTRY

- getCreationCode

```
library ERC6551BytecodeLib {
    function getCreationCode(
        address implementation_,
        uint256 chainId_,
        address tokenContract_,
        uint256 tokenId_,
        uint256 salt_
    ) internal pure returns (bytes memory) {
        return
            abi.encodePacked(
                hex"3d60ad80600a3d3981f3363d3d373d3d3d363d73",
                implementation_,
                hex"5af43d82803e903d91602b57fd5bf3",
                abi.encode(salt_, chainId_, tokenContract_, tokenId_)
            );
    }
}
```

其他略

参考资料

主要参考资料

- EIP-6551
  https://eips.ethereum.org/EIPS/eip-6551
  https://ethereum-magicians.org/t/erc-6551-non-fungible-token-bound-accounts/13030
- EIP-1667 https://eips.ethereum.org/EIPS/eip-1167
- eth waterloo https://www.theblockbeats.info/news/42973
- Tokenbound https://docs.tokenbound.org/sdk/methods#tokenboundclient