

SafeWallet

Wallet for everyone, not for the geeks

課題：Web3のセキュリティ

Web3では自己責任の世界であり、初心者へのハードルが高い

ウォレットとWeb3アプリの仕組み

Wallet 

シードフレーズ

秘密鍵

EOAアドレス

lorem ipsum
dolor ... sit
amet



P2cRtN...
.UnX



0xEE3....0
EED



3hPy...9A
bn



0xDE1....1
389



4bX...RtN
RBP



0x30A....b
AAD

トランザクション

サイン

DApps  OpenSea

Contract code

```
contract StandardToken is ERC20, BasicToken {
    mapping (address => mapping (address => uint256)) internal allowed;

    /*
     * @dev Transfer tokens from one address to another
     * @param _from address The address which you want to send tokens from
     * @param _to address The address which you want to transfer to
     * @param _value uint256 the amount of tokens to be transferred
     */
    function transferFrom(address _from, address _to, uint256 _value) public {
        require(to != address(0));
        require(_value == balanceOf(_from));
        require(_value == allowed[_from][_to]);
        balances[_from] = balanceOf(_from) - _value;
        balances[_to] = balanceOf(_to) + _value;
        allowed[_from][_to] = allowed[_from][_to] - _value;
    }
}
```

Web3の特徴①: シードフレーズ/秘密鍵

シードフレーズや秘密鍵がアドレスの全権限を管理するが、**漏洩すると全資産がリスクに晒される**

Web3の特徴②: トランザクション

Web3の取引はトランザクションとサインを活用するが、**間違えた形式で送ると資産が盗まれることも**

Web3のハッキングの事例

ハッキング手法は巧妙で、小さなミスが全財産の喪失に繋がる

	狙い	詳細
メタマスク サポート詐欺	①シード フレーズ/ 秘密鍵	公式サポートを語った詐欺師が シードフレーズや秘密鍵を言葉巧みに聞き出す 。安易に教えてしまうと、すぐさま全財産を引き抜かれる
マルウェア	①シード フレーズ/ 秘密鍵	「プロジェクトのWhitepaperである」や「BCGのα版のテストプレイをしてくれ」等と ファイルのダウンロードを誘導 。実行してしまうと PC上の秘密鍵が抜かれ 、資産が盗まれる
偽サイト	②ラン ザクション /署名	詐欺師が公式サイトに似せた 偽サイトを作成 。Wallet操作をすると、 悪意のあるランザクションやサインをさせられ、資産が盗まれる（次ページ詳細）
フロントエンド ハッキング	②ラン ザクション /署名	フロントエンド自体がハッキングされ 、そこでWallet操作をすると、 悪意のあるランザクションやサインをさせられ、資産が盗まれる（次ページ詳細）

トランザクション・署名による詐欺の詳細

悪意のあるコントラクトにsetApprovalForAllをする

```
function setApprovalForAll(address operator, bool approved)
```

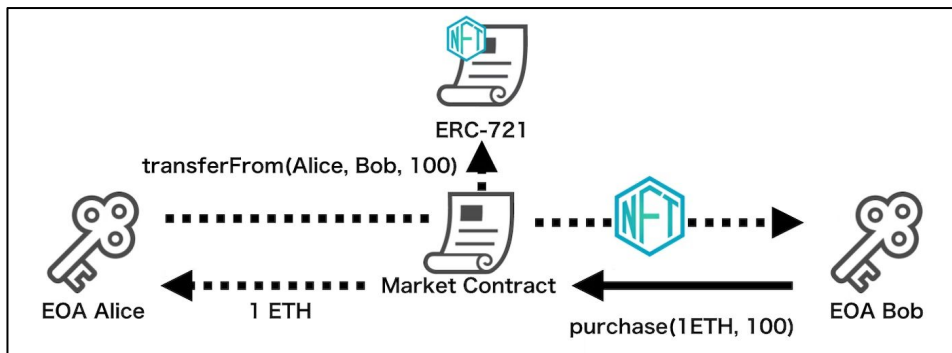
どのアドレスに承認するか

承認可否(0,1)

通常はNFT Marketに対して全権限許可をして…



…売買発生時にNFT Marketが取引処理を実施



→一般的に使うFunctionでありながら、**一つ目の引数（「どのアドレスに承認をするか」）を間違えて、悪意のあるコントラクトに権限許可してしまうと、NFTを盗まれてしまう**

(例: <https://etherscan.io/tx/0x6d380cc5616137464c6111526fae131bde890a4e4c1aacfe965d7724468429be>)

トランザクション・署名による詐欺の詳細

DEXで交換したトークンの送り先が別アカウントに設定

```
function swapExactTokensForTokens(uint amountIn, uint amountOutMin,  
address[] calldata path, address to, uint deadline)
```

トークンの交換ルート

送付先

期限

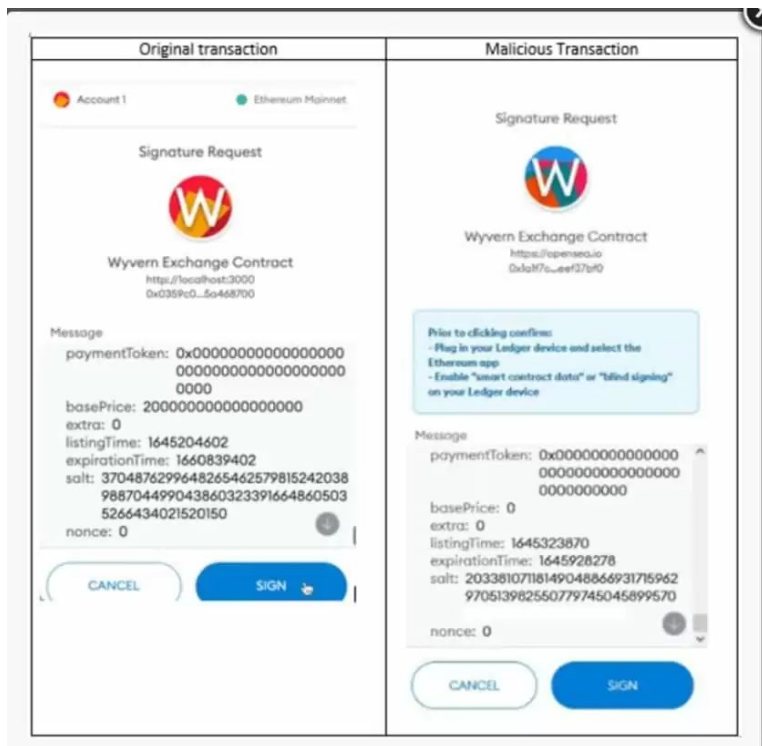


→一般的に使うFunctionだが、**四つ目の引数（「送付先」）が異なると、交換した先のトークンがハッカーのアドレスに送られてしまう**

(例: <https://polygonscan.com/tx/0x178200b693884968aaeb9e3b5955fec5b5c5913b6f3b3dac14a8a29a8f3a725f>)

トランザクション・署名による詐欺の詳細

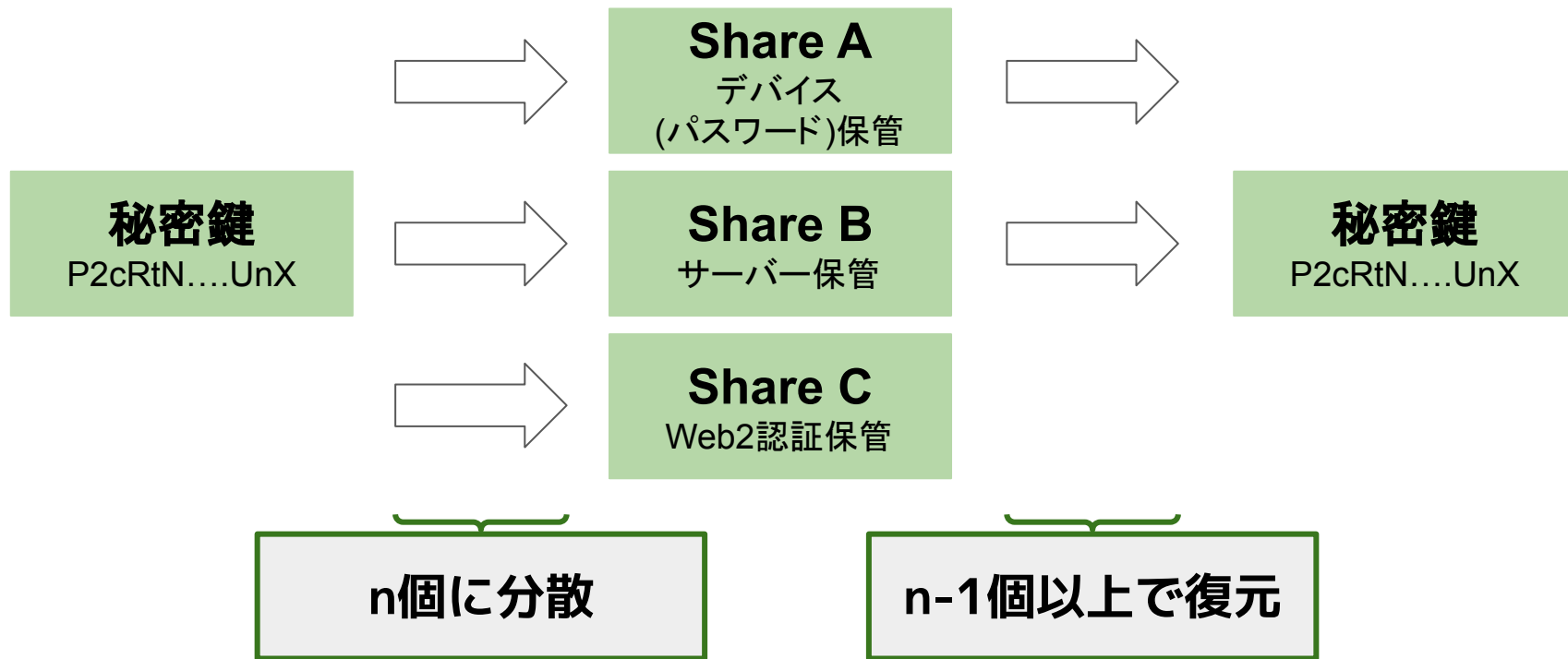
「Openseaで0WETHで売買成立した」という偽サインの署名



→Opensea等の一部サイトは、データのやり取りにSignを使用。中身を精査せずにSignを行うと、**意図していない取引が成立し NFTが盗まれる**ことがある
(例：0WETHで売買成立、等)

解決先：秘密鍵のセミカストディアル管理

秘密鍵を分散させユーザーに秘密鍵を意識させない



解決先：トランザクション/サインのWL化

安全なトランザクション以外の操作を禁止

