"Multitudes y charlatanes han entrado en los espacios de criptomonedas y contratos inteligentes que no sólo carecen de matices cypherpunk, sino que odian los valores del cypherpunk, incluyendo los valores como la minimización de la confianza que le dan el valor de mercado a ciertas criptomonedas como Bitcoin"

Nick Szabo

ABSTRACT

En un mundo donde la privacidad financiera es cada vez más vulnerable y la libertad económica está bajo amenaza, es crucial tener opciones para proteger nuestros activos y mantener nuestras transacciones seguras. Con la creciente importancia de la privacidad en la era digital, se hace necesario contar con herramientas que permitan a los usuarios tener el control total de sus datos y resguardar su privacidad. Es aquí donde entra en juego el contrato de ofuscación de transacciones de Algorithm X, una solución para aquellos que buscan una mayor privacidad en sus transacciones y una forma de proteger su libertad digital.

Introducción

La introducción de un contrato inteligente debe comenzar con una presentación clara de su propósito y funcionalidad, incluyendo los problemas específicos que resuelve. En términos técnicos, se podría hacer referencia a la implementación de un conjunto de reglas y protocolos programados en un lenguaje de programación para automatizar la ejecución y cumplimiento de un contrato. Además, se puede mencionar que los contratos inteligentes son un tipo de aplicación descentralizada que se ejecuta en una cadena de bloques y se utiliza para garantizar la seguridad, privacidad y transparencia en las transacciones financieras y comerciales. Es importante destacar que estos contratos eliminan la necesidad de intermediarios y terceros confiables en las transacciones, lo que proporciona una mayor eficiencia, seguridad y confianza en el proceso.

Este contrato inteligente, llamado Algorithm X

, ha sido creado con el objetivo de ofrecer una solución a los problemas de privacidad y libertad financiera que enfrentan los usuarios de criptomonedas. En un mundo cada vez más digitalizado, es fundamental contar con herramientas que permitan proteger nuestra información personal y financiera de terceros malintencionados, así como tener control absoluto sobre nuestros activos digitales.

Algorithm X, construido sobre la tecnología de la cadena de bloques Polygon MATIC, ofrece un alto nivel de privacidad al utilizar una función de transferencia que oculta el origen y destino de las transacciones. Además, cuenta con una función para generar códigos aleatorios que permiten a los usuarios retirar sus fondos de manera segura y privada. Todo esto sin comprometer la seguridad y transparencia que caracterizan a la tecnología blockchain.

Creemos firmemente que el acceso a la libertad financiera es un derecho fundamental, y Algorithm X es nuestra contribución a este objetivo. En este contrato, no solo encontrarás una solución a tus necesidades de privacidad y seguridad en el manejo de criptomonedas, sino también una invitación a reflexionar sobre la importancia de la libertad financiera y la necesidad de protegerla.

VENTAJAS

La privacidad es una de las mayores ventajas que ofrece nuestro contrato de ofuscación de transacciones. En la red de Polygon, las transacciones son públicas y cualquier persona puede rastrear las transacciones de cualquier dirección. Sin embargo, nuestro contrato ofrece una solución para aquellos usuarios que desean mantener sus transacciones privadas. Al ofuscar las transacciones, nuestro contrato asegura que las transacciones no se puedan rastrear a una dirección específica. Esto se logra mediante la generación de direcciones aleatorias y la división de los tokens en partes iguales que se transfieren a estas direcciones. De esta manera, no se puede vincular una dirección específica con una transacción determinada.

La seguridad es otra ventaja importante de nuestro contrato. Al dividir los tokens en partes iguales y transferirlos a direcciones aleatorias, nuestro contrato reduce el riesgo de pérdida o robo de tokens. En caso de que un atacante pueda obtener acceso a una de las direcciones aleatorias, solo tendría acceso a una fracción de los tokens, en lugar de la totalidad.

Nuestro contrato también ofrece una solución para aquellos usuarios que desean enviar grandes cantidades de tokens sin revelar la dirección del destinatario final. Al dividir los tokens en partes iguales y transferirlos a direcciones aleatorias, nuestro contrato garantiza que el destinatario final reciba la cantidad total de tokens, sin que se revele su dirección.

Además, nuestro contrato es fácil de usar y se puede acceder a él desde cualquier billetera compatible con la red de Polygon. Los usuarios simplemente tienen que generar un código de transferencia y enviar sus tokens a la dirección proporcionada. Una vez que se han confirmado las transacciones, los usuarios pueden reclamar sus tokens utilizando el mismo código de transferencia. Los tokens se enviarán a su dirección de billetera especificada.

En resumen, nuestro contrato de ofuscación de transacciones ofrece una solución para aquellos usuarios que desean mantener sus transacciones privadas y seguras. Además, ofrece una solución para aquellos usuarios que desean enviar grandes cantidades de tokens sin revelar la dirección del destinatario final. Nuestro contrato es fácil de usar y se puede acceder desde cualquier billetera compatible con la red de Polygon.

Funciones Especiales

Transferencia clásica ofuscada

```
//funcion de tranferencia ofuscada
//Obfuscated transfer function
function transfer(address to, uint256 amount) public override returns (bool) {
require(balanceOf(msg.sender) >= amount, "Not enough tokens");

//Generar 10 direcciones aleatorias
//Generate 10 random addresses
uint256[] memory addresses = new uint256[](10);
for (uint256 i = 0; i < 10; i++) {
    addresses[i] = uint256(keccak256(abi.encodePacked(block.timestamp, i, msg.sender))) % 2**160;
}

//Dividir la cantidad de tokens en 10 partes iguales
//Divide the amount of tokens into 10 equal parts
uint256 numChunks = 10;
uint256 chunkSize = amount / numChunks;

//Enviar los tokens a las direcciones aleatorias fantasmas
//Send the tokens to the phantom random addresses
for (uint256 i = 0; i < numChunks; i++) {
    address recipient = address(uint160(addresses[i]));
    _transfer(msg.sender, recipient, chunkSize);
}

//Transferir los tokens de las direcciones fantasmas al usuario real
//Transfer the tokens from the phantom addresses to the real user
for (uint256 i = 0; i < numChunks; i++) {
    address recipient = address(uint160(addresses[i]));
    _transfer(recipient, to, chunkSize);
}

return true;
}
```

En primer lugar, esta función se encarga de transferir una cantidad de tokens desde la dirección del remitente hacia la dirección del destinatario, pero lo hace de manera ofuscada para aumentar la privacidad de la transacción.

Para lograr esto, la función utiliza dos pasos principales:

1. Generar direcciones aleatorias: Se generan 10 direcciones aleatorias utilizando el hash de una combinación de la marca de tiempo del bloque, un índice y la dirección del remitente. Estas direcciones son "fantasmas" y no pertenecen a ninguna persona física o jurídica.

2. Transferir tokens a las direcciones aleatorias y luego al destinatario: La cantidad de tokens se divide en 10 partes iguales y se envía cada una a una de las direcciones aleatorias generadas anteriormente. Después, los tokens son transferidos de las direcciones aleatorias al destinatario final en partes iguales.

Este proceso crea múltiples transacciones que son difíciles de rastrear,en la cadena de bloques la cuenta A de origen nunca se vinculara con la B de destino aquí se produce un enmascaramiento de datos, solo se reflejara en el polygonscan 20 tranferencias aumentando así la privacidad de la transacción. Además, la división de tokens en partes iguales hace que sea más difícil para los atacantes obtener información sobre el monto total de los tokens transferidos

Transferencia por Código

```
//funcion generada por codigo aleatorio Paso N1
//function generated by random code Step N1
function generateTransferCode(uint256 amount) public returns (bytes32) {
    require(balanceOf(msg.sender) >= amount, "Insufficient balance");
    bytes32 code = keccak256(abi.encodePacked(msg.sender, block.timestamp, amount));
    transferCodes[msg.sender] = code;
    transferAmounts[code] = amount;
    _burn(msg.sender, amount);
    return code;
}


//funcion para generar reclamar codigo aleatorio Paso N2
//function to generate claim random code Step N2
function getCode() public view returns (bytes32) {
    return transferCodes[msg.sender];
}


//funcion para retirar el saldo por medio del codigo Paso N3
//function to withdraw the balance through the code Step N3
function withdrawWithCode(bytes32 code) public {
    require(transferAmounts[code] > 0, "Invalid code");
    uint256 amount = transferAmounts[code];
    transferAmounts[code] = 0;
    _mint(msg.sender, amount);
}
```

En cuanto a la función de transferencia por códigos, esta función permite a los usuarios generar códigos de transferencia para compartir con otros usuarios, en lugar de transferir los tokens directamente a la dirección del destinatario. Esto puede ser útil en situaciones donde el destinatario no desea compartir su dirección o para transacciones

privadas. El receptor del código de transferencia puede luego canjearlo para recibir los tokens transferidos.

1. Generar un código de transferencia: el usuario debe llamar a la función generateTransferCode en el contrato y especificar la cantidad de tokens que desea transferir y la dirección del destinatario. El contrato generará un código único (0x1234561234656123456123456123456123456)
2. Una vez generada la función generateTransferCode el remitente debe llamar a la función getCode y se le proporcionara el un código de tranferencia aleatorio que solo será utilizable una vez no tendrá fecha de vencimiento los tokens estarán guardados en el contrato Ejemplo (0x1234561234656123456123456123456123456) de transferencia generado
3. El destinatario una vez que haya recibido el código puede retirar los tokens: después de llamar a la función withdrawWithCode. Esto transferirá los tokens a la dirección del destinatario y los eliminará del contrato.En este caso las cuentas A remitente y B destinatario jamas se cruzaran en polygonscan solo figurara que una dirección 0x0000000000000000000000000000000000 hizo una transferencia al destinatario

En resumen, la transferencia por códigos permite a los usuarios transferir tokens de forma segura sin tener que revelar sus direcciones públicas o privadas a la otra parte.

Tokenomics

Nota:el equipo renuncio a los tokens enviándolos a un contrato de distribución que se encargara del manejo de los mismos

El contrato DistributeAlgorithmX es un contrato inteligente escrito en Solidity que se utiliza para administrar los tokens del proyecto Algorithm X. En este contrato, los desarrolladores del proyecto no manejan los fondos, sino que el contrato lo hace.

El contrato tiene en su poder un total de 21 millones de tokens disponibles, y el porcentaje de cada categoría se divide de la siguiente manera:

- 30% para la venta pública (inmediato)
- 10% para airdrops (180 días de bloqueo después del deploy)
- 8% para el equipo (con un bloqueo de un año)
- 12% para la reserva del ecocistema
- 10% para el marketing
- 30% para una segunda venta pública después de un año

El contrato tiene las direcciones de las billeteras de airdrop, equipo, reserva y marketing, que son las destinatarias de los tokens según su porcentaje.

Además, el contrato tiene un bloqueo de 180 días para los tokens de airdrop y un bloqueo de un año para los tokens del equipo. El tiempo de la segunda venta pública se establece para un año en el futuro.

El contrato también incluye una función llamada "distributeTokens", que se utiliza para distribuir los tokens según los porcentajes establecidos. Los tokens se envían a las direcciones correspondientes después de que se cumplan los bloqueos de tiempo.

En resumen, este contrato garantiza que los tokens del proyecto sean administrados de manera justa y transparente, y que los desarrolladores no puedan acceder a los fondos de manera deshonesta.

Conclusión

En resumen, el contrato de ofuscación de transacciones proporciona una solución innovadora y efectiva para aquellos usuarios que desean mantener su privacidad y seguridad mientras realizan transacciones con criptomonedas. A través de la generación de direcciones aleatorias y la división de tokens en partes iguales, este contrato brinda una mayor seguridad y privacidad a los usuarios, permitiéndoles realizar transacciones sin preocuparse por la exposición de su información personal o financiera.

Además, la función de transferencia por código proporciona una forma adicional para que los usuarios transfieran tokens de manera segura y privada, evitando la necesidad de compartir direcciones públicas o privadas.

En conjunto, el contrato de ofuscación de transacciones y la función de transferencia por código ofrecen a los usuarios una mayor seguridad y privacidad al realizar transacciones con criptomonedas, abriendo nuevas posibilidades y oportunidades en el espacio de las criptomonedas.

Dedicatoria:

Este proyecto está dedicado a Nick Szabo, un pionero en el campo de las criptomonedas y la tecnología blockchain. Sus contribuciones y visiones han sido fundamentales para el desarrollo y la evolución de este campo, inspirando a muchos a seguir adelante en la exploración de las posibilidades y el potencial de esta tecnología revolucionaria.

AlgorithmX@gmx.com