به نام خدا

https://etherscan.io/address/0x4Dd942bAa75810a3C1E876e79d5cD35E09C97A76#code

dash2trade لینک کد های قرار داد

در ابتدا از سالیدیتی ورژن 0.8.0 استفاده شده و یا بالاتر

یک اینترفیس که داخل آن:

یک ایویت تعریف شده که احتمالا لاگ های تراکنش را ذخیره میکند

و یک ایونت دیگر تعریف شده که دو آدرس و یک اینتجر ذخیره میکند

totalsupply:

تنها عدد برمیگرداند و چیزی دریافت نمیکند احتمالا مجموع توکن ها را برمیگرداند

balanceof:

این تابع متغیر آدرس دریافت میکند و متغیر عددی به ما برمیگرداند تابع به صورت اکسترنال تعریف شده و عددی رو به کاربر نمایش میدهد (ویو). دلیل اکسترنام بودن این تابع این است که در یک اینترفیس تعریف شده است

transfer:

تابعی دیگر از اینترفیس که یک آدرس و یک مقدار از کاربر میگیرد و بولین برمیگرداند و اکسترنال تعریف شده

allowance:

آدرس میگیرد و عدد برمیگرداند

approve:

آدرس میگیرد عدد میگیرد ولی بولین برمیگرداند

transferfrom:

دو آدرس و یک عدد میگیرد و بولین برمیگرداند

درواقع این توابع برای نوشتن یک توکن تقریبا حالت پرتکل دارند که داخل اینترفیس نوشته شده اند

بعد از این اینترفیس یک اینترفیس دیگر داریم که از اینترفیس قبلیوراثت میگیرد

از اسامی و کامنت های کد پیداست که این توابع برای برگرداندن نام و نماد توکن و اعشار توکن استفاده میشود

یک آبسترکت کانترکت تعریف شده

داخل این کانترکت دو فانکشن تعریف شده

برای تعریف هردوی این فانکشن ها از کلمه مجازی استفاده شده که احتمال این را نشان میدهد که از این دو فانکشن وراثت گرفته شده یا به اصطلاح اور راید شده

یک کانترکت به نام تووکن که از دو اینترفیس قبلی وراثت گرفته تعریف شده است

اولین خط کانترکت یک مپینگ تعریف شده که به صورت آرایه داده هارا ذخیره میکند

_allowance بعد از آن یک مپ تو در تو تعریف شده به نام

ایندکس مپینگ با دیتا تایپ آدرس و مقدار ذخیره عدد میباشد و به خاطر خاصیت پرایوت آن خارج از کانترکت قابل دسترسی نیست

یک متغیر عددی تعریف شده به نام

_totalsupply

دو متغیر رشته ای و دو عددی تعریف شده

uintعدد جلوی

نشانه مقدار حافظه ای که برای ذخیره اختصاص میابد میباشد تا از هزینه گس کمتری بهره ببرد

کانستراکتور یا سازنده تابعی که به محض دیپلوی کانتکت ران میشود تعریف شده

سه آرگومان میگیرد دو تا رشته ای که به خاطر کلید واژه ی مموری داخل رم ذخیره میشود(هزینه گس کمتر) و یک آدرس

شرط اجرای کانستراکتور این است که آدرس آرگومان دارای یک مقدار باشد درواقع یک آدرس واقعی باشد

داخل کانستراکتور نام و سمبل را در متغیر هایی که قبل کانستراکتور معرفی شد ذخیره میکند و با فراخوانی تابع مینت و ارسال دو پارامتر تعداد توکن ها ضرب میشود به آدرسی که به عنوان پارامتر ارسال شده

فانکشن های پایین تر شامل کلمه اورراید میباشند که از فانکشن هایی که بالاتر توضیح دادم وراثت میگیرند

و اطلاعات ثبت شده توکن را به ما برمیگردانند

تفاوت فانکشن های ترنسفر و ترنسفر فرام در این است که ترنسفر توسط کاربر انجام میشود اما ترنسفر فرام درواقع به یک کانترکت اجازه عمل انتقاد داده میشود

در فانکشن

_transfer

که از کتاب خانه اوپن زپلینک میباشد با دو شرط شروع شده که جفت آدرس های ورودی باید حقیقی باشند

ابتدا موجودی حساب را در یک متغیر ذخیره کرده و با شرط اینکه مقدار انتقالی که کاربر وارد کرده از حسابش نباید بیشتر باشد ادامه فانکشن را تعریف کرده اند.

حالا مقدار مورد نیاز از حساب انتقال دهنده کم و به حساب دریافت کننده زیاد میشود.

در آخر ایونت ترنسفر ما شلیک میشود تا اطلاعات تراکنش ذخیره شود.

فانکشن مینت هم آدرس میگیرد مقدار میگیرد و به کل مقدار یا توتال ساپلای و آدرس مورد نظر اضافه میکند

همچنین یک ایونت دیگر با دستور امیت اجرا شده و تراکنش آن ذخیره میشود

فانکشن برن هم دقیقا برای سوزاندن توکن استفاده میشود و بر عکس فانکشن مینت عمل میکند

تنها فاهم این فانکشن با فانکشن قبل امیت میباشد که تراکنش را ذخیره کند

و در آخر با فانکش اپرو دو آدرس و یک متغیر عددی دریافت میکند با دو شرط صحت آدرس ها شروع شده و به مپینگ تو در توی الاونس کار دارد که ایندکس اول این مپینگ یک آدرس میباشد و ایندکس دوم نیز آدرس و مقدار ذخیره شده عدد میباشد که ما تمام این هارا به در این فانکشن به مپینگ پاس میدهیم

و امیت ایونت اپروال رو شاهد هستیم برای ذخیره اطلاعات

تفاوت توکن پری سیل و جنرال سیل:

درواقع پریسیل نوعی بازاریابی حساب میشود که توکن ها ارزون تر از حالت عادی به فروش میرسد(معمولا)

یعنی قبل از اراعه عمومی و برای جذب سرمایه بیشتر یا کمک مالی بشتر این اتفاق می افتد

تقریبا مانند اتفاق عرضه اولیه داخل بورس :)

ضعف این استرتژی در این میباشد که بسیاری از خریداران به محض عرضه عمومی تمام توکن هارا میفروشند و این اتفاق معمولا باعث افت قیمت در حالت عادی و چیزی که واقعا توکن شاید لایق آن باشد میشود

d2t:دید سالیدپروف نسبت به

https://github.com/solidproof/projects/tree/main/Dash2Trade

درواقع سالیدپروف یک سری تست ها برای اسکم نبودن و ایمن بودن پروژه ها و بلاکچین ها تعریف کرده و درصورت درخواست آنهارا تست میکند

پروژه ی دی 2 تی نیز از این ماجرا بهره برده

گـواهی انطباق

گزارش آسیب‌پذیری و توصیه‌های کد

(، اسکریپت های سفارشیMythX، Slither) تست آسیب پذیری خودکار

(SWC-Registryبررسی کد دستی،) تست امنیت دستی

presale:اطلاعات مربوط به فروش توکن در حالت

همچنین برای تایید اینکه آیا این پروژه موفق یا محکوم به شکست میباشد یک سری علایم رو بهتر است چک کنیم

مثلا حقیقی بودن افراد تیم و تحقیق درباره آن ها مثل پیدا کردن لینکدین و صفحات مجازی آن ها

وجود تفکر پشت پروژه و اضافه کردن یک قابلیت به طور خلاصه هدفمند بودن پروژه

چک کردن کدهای پروژه

رودمپ پروژه و به موقع رسیدن به مراحل تعیین شده

کامیونیتی پروژه و پاسخگو بودن ادمین ها که اکثرا در دیسکورد توییتر و تلگرام فعال هستند

طرز کار متامسک و تراست ولت با توکن های جدید:

برای اینکه بتوانیم از توکن های جدید داخل متامسک یا تراست ولت بهره ببریم باید آن ها را در لیست دارایی های خودمان اضافه کنیم

اول باید شبکه ی ت.کن را چک کنیم اگر درلیست شبکه ها نبود آن را اضافه میکنیم و سپس آدرس قرارداد توکن را وارد میکنیم

که آدرس توکن ها عموما در وایت پیپر و صفحه اصلی سایت پیدا میشود

سیستم کار ایردراپ و کدنویسی آن:

اگر ما بخواهیم ایردراپ داشته باشیم ابتدا یک لیست باید از آدرس هایی که میخواهیم وارد مرحله ایردراپ های ما بشوند ایجاد بکنیم

این کار به شکل های مختلفی صورت میگیرد مثلا یکی از آن ها فالوور های توییتر میباشد

یا عمل ریتوییت کردن که بعضی پروژه ها با این روش کامیونیتی خود را تبلیغ میکنند به این شکل که از کاربر میخواهند یک توییت را ریتوییت کنند و آدرس ولت خود را نیز زیز در همان توییت وارد کنند

کد های آن به این شکل خواهد بود :

```
contract ERC 20{

function transfer(address _recipient, uint 256_value) public returns (bool success);

}


contract Airdrop {

function drop(ERC 20token, address[] recipients, uint256[] values) public {

for (uint 256i =  0; i < recipients.length; i++) {

token.transfer(recipients[i], values[i]);

}

}
```

}