DeFi
分享

在 Solidity 中破解私有变量的快速指南

概述

每个人都多次听说过，智能合约中的一切都是开放的，每个人都可以看到。这意味着，不能在合约中隐藏敏感信息。在这篇简短的文章中，我们将学习如何从另一个合约中读取私有变量。

简单回顾一下：当声明一个变量(或函数)为“private”时，并不意味着它是“private”。因为没有人能够读懂它(因此才有了这个令人困惑的名称)。Private只是指“谁”可以使用它，换句话说，就是变量(或函数)的作用域。

Private只能从特定的合约中调用。

Internal可以从该合约和派生的合约中调用。

External 可以从外部源调用。

Public可以从任何地方调用。

在这篇文章中，我们不会详细讨论作用域，但是更重要的一点是，当声明一个状态变量public时，Solidity会自动为这个特定的变量创建一个getter函数，因此用户可以直接调用它。当一个变量被声明为“private”时，我们需要做更多……让我们开始吧!

想象一下，有人写了这样一份合约:

```
// SPDX-License-Identifier: MIT
pragma solidity >= 0.8.0 < 0.9.0;

contract Secret {

  address public owner;
  string private secretPassword;

  constructor(string memory _secretPassword) {
    owner = msg.sender;
    secretPassword = _secretPassword;
  }

  function getSecretPassword() public view returns (string memory) {
    require(owner == msg.sender, "You are not the owner!");
    return secretPassword;
  }
}
```

乍一看，似乎只有所有者可以获得分配给secretPassword的值。但是，正如我们之前看到的，所有内容都可以通过智能合约访问。

为了获得secretPassword的值，我们首先需要了解存储。

存储

默认情况下，Solidity中的所有状态变量都存储在storage中。这意味着在函数之外声明的所有变量都由EVM保存。有一个例外，当声明它为“常量”时。当将一个变量声明为常量时，它不会占用一个槽，而是放在编译后的代码中。

每个插槽的长度为32字节(32字节== 256位== 64十六进制(或半字节))

变量按照从0到n (n =最大容量)的顺序存储。如果某个变量超过了该特定槽的空间，那么它将被传递到下一个槽。

多个变量可以存储在一个槽下，如果它们合适的话。

Struct创建一个新的槽，结构的元素的行为与上面描述的相同。

固定大小的数组创建一个新的插槽。

动态数组为每个元素创建一个插槽。

映射存储在哈希(key, slot)

让我们应用这个。

如果我们回到合约，我们会看到secretPassword在第二个插槽(槽1)中。

```
// SPDX-License-Identifier: MIT
pragma solidity >= 0.8.0 < 0.9.0;

contract Secret {

  address public owner; // slot 0
  string private secretPassword // slot 1;

  constructor(string memory _secretPassword) {
    owner = msg.sender;
    secretPassword = _secretPassword;
  }

  function getSecretPassword() public view returns (string memory) {
    require(owner == msg.sender, "You are not the owner!");
    return secretPassword;
  }
}
```

(我把合约部署到Rinkeby，密码是“不是那么秘密!”)

为了检索该值，我们将使用ethers .js:

```
const { ethers } = require("ethers");
const provider = ethers.providers.getDefaultProvider("rinkeby");
const contractAddress = "0x62f9aA64Af88Ad57B0a6bdb59a8172d3a0897eCF";

const hex_to_ascii = _hex => {
  const hex  = _hex.toString();
  let str = '';
  for (let i = 0; i < hex.length; i += 2) {
    str += String.fromCharCode(parseInt(hex.substr(i, 2), 16));
  }
  return str;
 }

const decodePassword = async () => {
    // result in hex
    const storage1 = await provider.getStorageAt(contractAddress, 1);
    // convert it to ascii
    const result  = hex_to_ascii(storage1);

    console.log("password -->", result);

}

decodePassword();
```

如你所见，我们调用了**provider.getStorageAt(contractAddress, storageSlot)**函数。这将以十六进制返回结果。然后，我们只是将其转换为 ascii。就可以随意使用该代码和地址，它部署在Rinkeby上!

Source：https://medium.com/coinmonks/a-quick-guide-to-hack-private-variables-in-solidity-b45d5acb89c0

关于

ChinaDeFi- ChinaDeFi.com 是一个研究驱动的DeFi创新组织，同时我们也是区块链开发团队。每天从全球超过500个优质信息源的近900篇内容中，寻找思考更具深度、梳理更为系统的内容，以最快的速度同步到中国市场提供决策辅助材料。

Layer 2进发- 欢迎对Layer 2感兴趣的区块链技术爱好者、研究分析人与Gavin（微信：chinadefi）联系，共同探讨Layer 2带来的落地机遇。敬请关注我们的微信公众号**“去中心化金融社区”**。

评论 0

文明上网理性发言，请遵守《新闻评论服务协议》

请先登录后发表评论~

已显示所有评论