

跨链兑换之Multichain、Stargate

- [背景](#)
 - [资产跨链的三种方式](#)
 - [1. 中心化交易所CEX](#)
 - [2. 去中心化的跨链桥](#)
 - [3. 多链聚合器](#)
 - [跨链桥接和通信](#)
 - [目前的跨链桥接和通信方式](#)
- [一. 跨链项目1: Multichain](#)
 - [Multichain项目概述](#)
 - [特点](#)
 - [Multichain核心原理](#)
 - [SMPC \(Secure Multi-Party Computation 安全多方计算\)](#)
 - [阈值签名方案TSS](#)
 - [阈值签名和多重签名](#)
 - [跨链桥](#)
 - [跨链兑换流程](#)
 - [跨链兑换AnyswapRouter合约](#)
 - [合约重要方法 \(以v6为例\)](#)
 - [Multichain中跨链兑换交易示例](#)
 - [问题](#)
- [二. 跨链项目2--Stargate](#)
 - [Stargate概述](#)
 - [Stargate核心原理](#)
 - [合约地址 \(以BSC链为例\) :](#)
 - [swap 功能概述](#)
 - [LayerZero协议](#)
- [补充: 加密资产托管的三种方案](#)

背景

资产跨链的三种方式

1. 中心化交易所CEX

- 用户可以先将A链资产充值到CEX中，兑换（交易）为B链所需的资产，然后充值到B链上

2. 去中心化的跨链桥

- 跨链时，资产通常以「映射资产」的形式来实现，需要通过「锁定+铸造+赎回+销毁」等一系列过程

举例：用户想要把A链上的原生资产AToken跨链至B链上

1. 原始A链：用户把将要跨链的资产AToken存入源链智能合约并“锁定”
2. 由预言机告知B链上的智能合约，待矿工验证过后，目标链B链上，1:1“铸造”出源链的封装资产（如AnyUSDC、WBTC），并将其发送给用户
3. 当用户赎回A链的资产（从B链返回A链时），智能合约就会在目标链上1:1“销毁”封装资产（如AnyUSDC、WBTC），并将源链上锁定的原生资产释放给用户。

- 缺点：这种「1对1跨链桥」模式下的A资产从源链跨至目标链时，所得资产并不是原生的A资产了，而是映射后的包装资产。**用户持有的包装资产（如WBTC）在跨链桥发生问题时，可能会变得毫无价值。**往往需要在目标链上将包装资产兑换为原生资产才能更好地使用。

3. 多链聚合器

- 将不同链上资产的流动性集中起来，构建出跨链资产的**流动性资金池**，用户可以在池中完成A链上的X资产兑换成B链上X资产的过程。
- 缺点：「流动资金池」式的聚合跨链工具们，往往由于资金池被部署在不同的区块链网络中，因此**不能统一各条链上的流动性**，造成每条链上的流动性资金池内的资产数量有限，无法满足用户的大额跨链转账需求。

跨链桥接和通信

- 区块链彼此独立，每条链都有自己的架构讯息，不与其他链互通。比如，以太坊链上的ETH无法被Solana辨识，因为ETH并不是Solana架构下的产物。若要转移它，只能使用跨链协议（IBC）
- 跨链token：

balance float: 价值恒定
mint/burn
liquidity swap
wrapped + mint/burn

目前的跨链桥接和通信方式

1. 由中间链验证并转发链上信息：缺点是安全性低。

原始链 -- 中间链 -- 目标链

要依靠中间链来执行你的验证，并相信它的共识。中间链有充分的签署权，可以将自己的交易写到目标链上，而这些目标链则隐含地信任这些交易。

这里明显的问题是有可能被利用/攻击。如果中间链的共识被利用，所有配对网络上的所有流动资金几乎可以立即被盗。考虑到需要确保数十亿美元的安全，这个中间链将不可避免地成为未来攻击的一个关键故障点。

值得注意的是，中间链的方法通常比其他解决方案要便宜得多

2. 在链上运行轻节点：费用昂贵

把整个区块头的顺序历史，存储在对面链上（反之亦然）。然后，包含消息的交易证明被转发并在链上对照区块头进行验证。

这是在链之间传输信息的最安全的方法。然而，这也是最昂贵的解决方案

一. 跨链项目1：Multichain

Multichain项目概述

- 官网：<https://multichain.org/>
- 2020年7月20日上线的跨链解决方案提供商。主要优势是：起步时间较早、服务覆盖较广、流动性深度较好、跨链手续费率低。
- 非托管+MPC模式
- Multichain在多条链上部署了「流动性资金池」，来帮助用户完成资产的直接跨链。
- 官方文档：<https://docs.multichain.org/getting-started/introduction>
- 参考：
<https://mirror.xyz/0xd05cFA28Eaf8B4eaFD8Cd86d33c6CeD1a1875417/nW1f2MnTxZgh7vd1QB3t6qrb4id1PbRdC7fO1bUdkOs>

特点

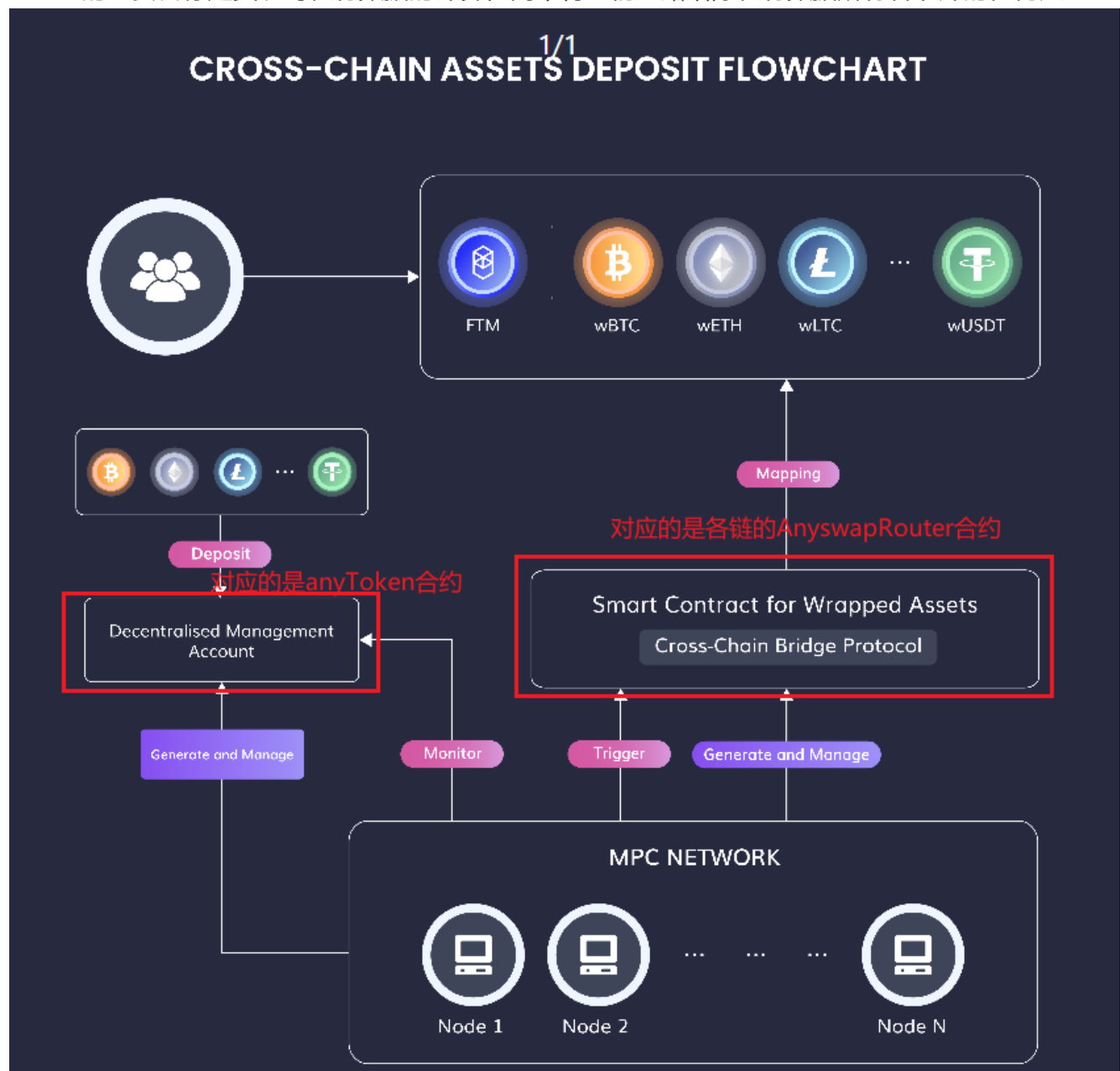
1. 非托管和MPC（多方安全计算）模式。MPC是去中心化的象征，处理多链跨链桥和其他链上的智能合约方法
2. **无滑点，仅需支付手续费**。Multichain 的 1:1 交换使用户能够执行 0 滑点转账并消除与 AMM 相关的隐藏成本。例如，用户想用 12 个 USDT 从 BSC 跨链 Polygon，可以在多链路由功能上进行跨链操作，扣除 gas 费 0.9USDT，实际到 Ploygon 有 11.1USDT
3. 多链路由器。Multichain Router 允许用户在任意两条链之间自由切换，它降低了费用和操作复杂度，并且更容易在链之间移动。

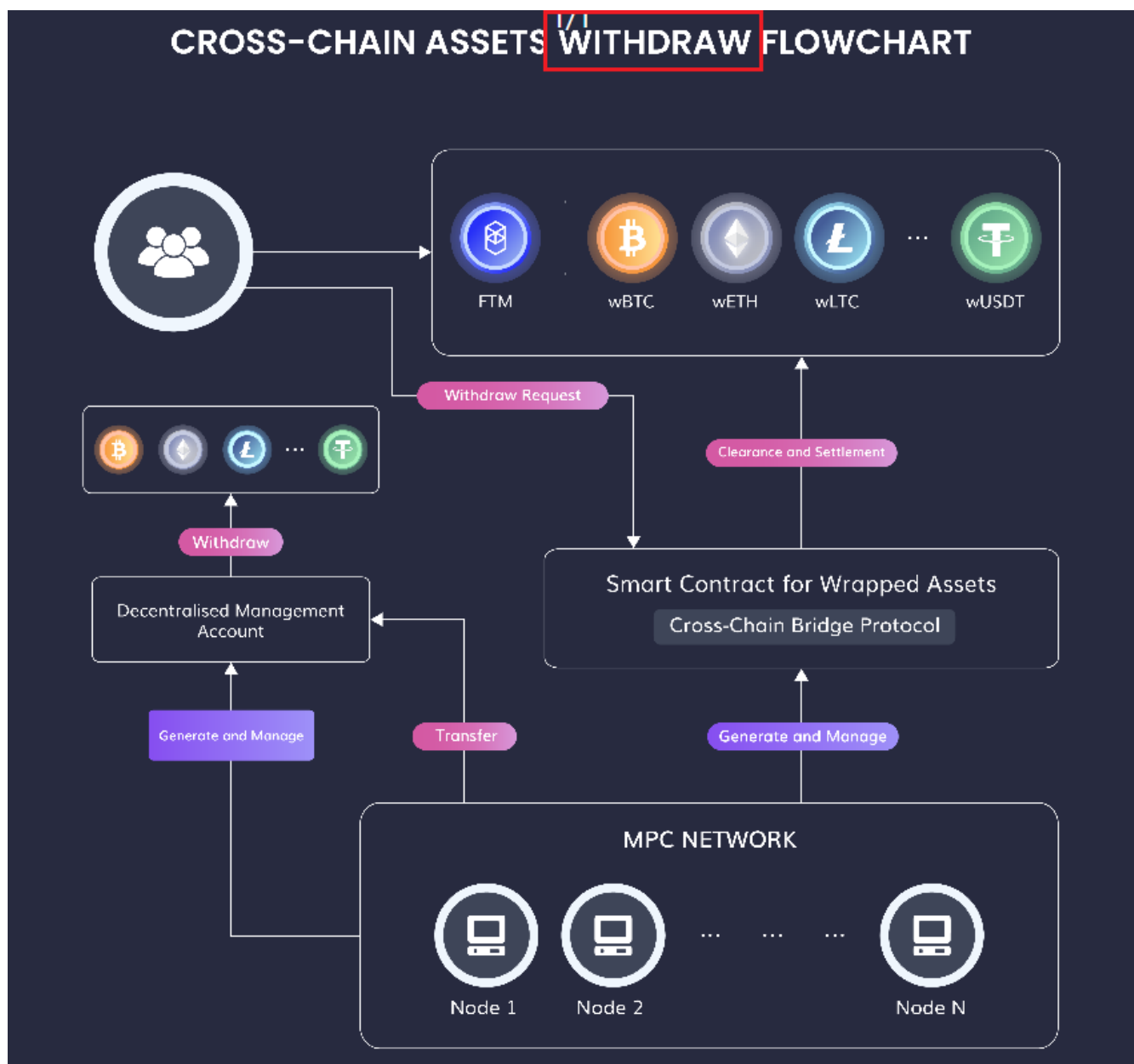
Multichain核心原理

SMPC（Secure Multi-Party Computation 安全多方计算）

- SMPC指在无可信第三方的情况下，多个参与方协同计算一个约定的函数，并且保证**每一方仅获取自己的计算结果**，无法通过计算过程中的交互数据推测出其他任意一方的输入和输出数据。

- SMPC的主要目标是实现对私有数据的计算，而不将它们透露给除私有数据所有者以外的任何人





- Multichain 的 SMPC Network 由一些**独立运行和维护的节点**组成，这些节点在需要**初始化生成公钥或进行签名时执行 TSS 算法**。
- 为实现加密资产的跨链交互，要求 SMPC Network 是一个**分布式网络**，实时处理链与链之间的跨链请求。即 **SMPC Network 实时检测原始链上的状态，然后将其转化为目标链上的行为**。当前的 MPC 网络上的每个节点将独立验证原链的状态，并在所有节点之间使用 TSS 算法对验证结果达成共识

阈值签名方案TSS

- TSS是密码学中安全多方计算(SMPC)领域的一部分
- TSS=分布式密钥 + 分布式签名（每个参与者都只持有部分私钥，公钥是根据所有用户的私钥计算得出的）
- TSS允许灵活的定义阈值策略，例如，三个用户共同管理一个私钥，每个人都只持有部分私钥碎片，为了进行交易签名，需要将至少两个用户的签名数据整合在一起才能构建有效签名。
- TSS的主要步骤：
 1. Vault初始化：该步骤将建立一个端到端加密的通信渠道，对参与各方进行初始化。例如需要三方参与

2. 密钥生成：在本步骤中，我们需要确定签名人数的阈值策略并共享私钥。例如，若采用2/3政策，这意味着私钥将被分割成3个私钥碎片，并且3个参与者中的任何2个都可以签署交易。
3. 签名：从各自的私钥份额中生成数字签名，且私钥不会泄露
4. Vault重组：如果有人丢失了他保管部分的私钥，则有必要重新分享私钥。重组会生成新的密钥碎片，同时之前的碎片将失效

阈值签名和多重签名

- 多重签名方案提供了与阈值签名相似的功能，必须要N个人中的M个人共同签名才有效，不同点是：
 1. 多重签名中每个参与者都有自己的完整私钥/公钥对，阈值签名中所有参与者对应的是同一个公钥、部分私钥
 2. 多重签名上的每次签名都需要一个单独的交易，就会导致gas费用很高。MPC 签名是在链下计算的，因此链上的交易费用将与单个签名钱包差不多

跨链桥

- 跨链桥是实现跨链兑换的基础，桥允许将A链的资产“发送”到B链
- 每个 Bridge 都是两个区块链之间的链接，Multichain中跨链桥的组成：资产来源链上Router智能合约 + 资产来源链上分散管理账户（即AnyswapToken合约，作为资金池）+ SMPC网络 + 目标链上Router智能合约+目标链分散管理账户。

跨链兑换流程

- 跨链兑换是基于跨链桥实现的
- 当用户将 XYZ 从链 A 转移到链 B 时，跨链流程是：
 1. XYZ 被添加到链 A 上的anyXYZ池中，触发事件
 2. 【v4中存在该步骤、v6中无】在链 A 上为用户铸造相同数量的 anyXYZ、然后再销毁掉

Overview Logs (5) Comments

Transaction Hash: 0xb38a636c6854ee1ad3eb8fe5eb108619aad9d9f83e56e0957172c0daff609052

Status: Success

Block: 19531803 276 Block Confirmations

Timestamp: 13 mins ago (Jul-14-2022 02:48:07 AM +UTC)

From: 0xbbbb456add87e8157fe1d977f5f72ce00ae50d

Interacted With (To): Contract 0xd1c5966f9f5ee6881ff6b261bbda45972b1b5f3

Tokens Transferred: 3

- From 0xbbbb456add87... To 0x8965349fb649a... For 439.944200374479903034 (\$440.00) Binance-Peg... (USDC)
- From Null Address: 0x00... To 0xbbbb456add87... For 439.944200374479903034 USDC (anyUSD...)
- From 0xbbbb456add87... To Null Address: 0x00... For 439.944200374479903034 USDC (anyUSD...)

v4: 铸造后再销毁
v6中直接省略该步骤

3. SMPC 节点网络检测到1.中的事件后，会在链 B 上为用户铸造anyXYZ
4. 接下来分为两种情况：
 - 如果链 B 上的 XYZ 的数量大于为用户创建的 anyXYZ数量，则将链 B 上的用户的 anyXYZ 烧毁，并将 XYZ 发送到链 B 上的用户钱包。
 - 如果 XYZ 的数量小于 为用户铸造的anyXYZ数量，则表示现在没有足够的余额可以提取。用户 swap最终接收到的就是anyXYZ（不会被转化为XYZ），这表示他们的池份额稍后将在再次有足够的时候由用户再自行赎回XYZ。

跨链兑换AnyswapRouter合约

- polygon链兑换合约 V6:
<https://polygonscan.com/address/0x2eF4A574b72E1f555185AfA8A09c6d1A8AC4025C>
- BSC链的兑换合约V4:
<https://bscscan.com/address/0xd1c5966f9f5ee6881ff6b261bbeda45972b1b5f3>
- BSC链anyUSDT: <https://bscscan.com/address/0xedf0c420bc3b92b961c6ec411cc810ca81f5f21a>

合约重要方法（以v6为例）

1. `event LogAnySwapOut(address indexed token, address indexed from, address indexed to, uint amount, uint fromChainID, uint toChainID);`
 - 用户进行跨链兑换时，会在**原始链**上触发该事件
 - 目的：Multichain SMPC网络会一直监听该事件
2. `event LogAnySwapIn(bytes32 indexed txhash, address indexed token, address indexed to, uint amount, uint fromChainID, uint toChainID);`
 - 用户进行跨链兑换时，会在**目标链**上触发该事件（由项目方调用时触发）
 - 目的：Multichain SMPC网络会一直监听该事件
3. `anySwapOutXxxx(address token, address to, uint amount, uint toChainID)external:`
 - 参数：token标识用户swap in的真实token对应的anyToken；to表示swap out出来的token要转给谁；amount表示兑换数量（还未扣除手续费）；toChainID表示swap的目标链的chainId
 - 用户进行跨链dex时直接交互的方法，在原始链上被用户触发
 - 该方法的作用是接收用户swap in的token，并触发LogAnySwapOut 事件（目的是通知 Multichain SMPC网络有跨链兑换请求）
 - 如下图所示，用户swap in的token的种类有：anyToken、ERC20 token、主网token三种

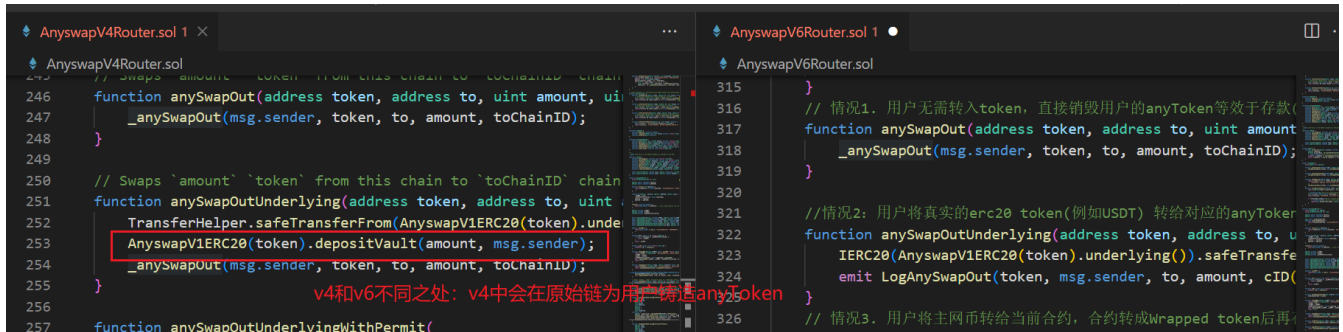
```
function _anySwapOut(address from, address token, address to, uint amount, uint toChainID) internal {
    // 销毁from用户的amount数量的anyToken、不会再有其他真实token的转移，只是会触发事件：Transfer(account, address(0),
    AnyswapV1ERC20(token).burn(from, amount);
    emit LogAnySwapOut(token, from, to, amount, cID(), toChainID);
}

// 情况1. 用户无需转入token，直接销毁用户的anyToken等效于存款(anyToken是之前存过款的凭证，与真实token的兑换比率是1: 1)
function anySwapOut(address token, address to, uint amount, uint toChainID) external {
    _anySwapOut(msg.sender, token, to, amount, toChainID);
}

// 情况2: 用户将真实的erc20 token(例如USDT) 转给对应的anyToken合约(例如anyUSDT);并触发事件LogAnySwapOut
function anySwapOutUnderlying(address token, address to, uint amount, uint toChainID) external {
    IERC20(AnyswapV1ERC20(token).underlying()).safeTransferFrom(msg.sender, token, amount);
    emit LogAnySwapOut(token, msg.sender, to, amount, cID(), toChainID);
}

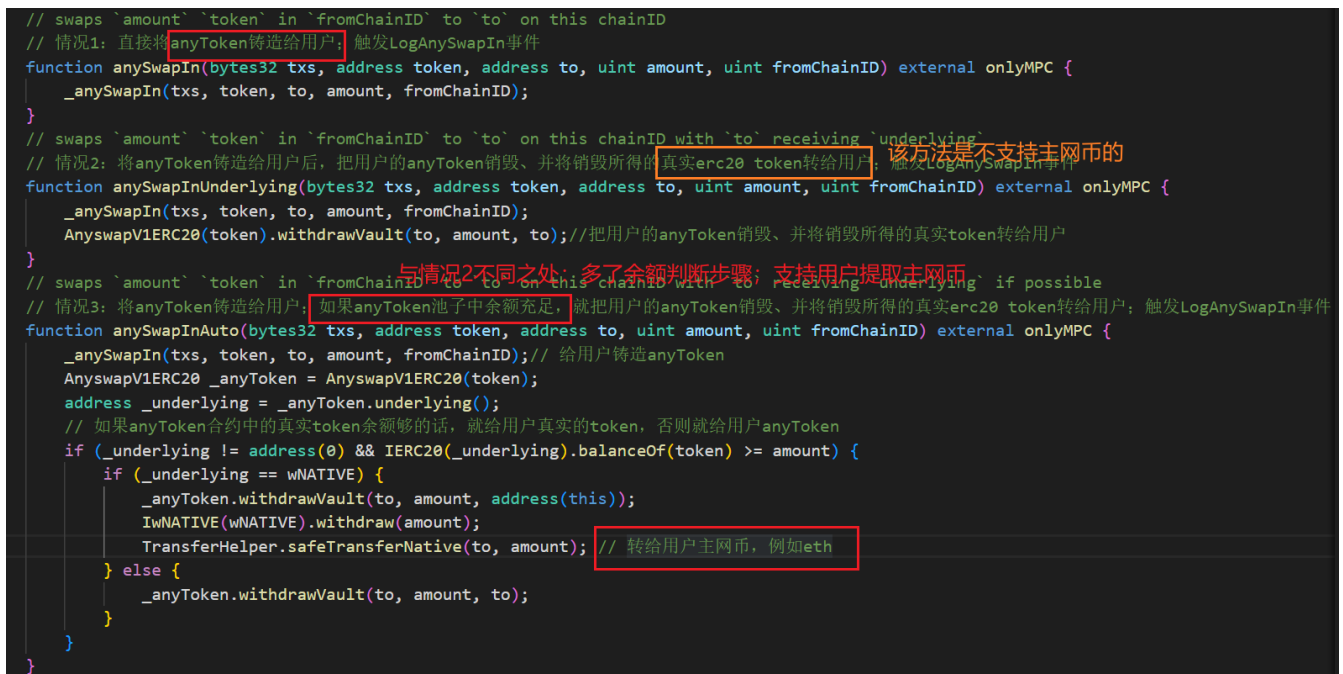
// 情况3. 用户将主网币转给当前合约，合约转成Wrapped token后再存入anyToken合约;并触发事件LogAnySwapOut
function anySwapOutNative(address token, address to, uint toChainID) external payable {
    require(AnyswapV1ERC20(token).underlying() == wNATIVE, "AnyswapV3Router: underlying is not wNATIVE");
    IwNATIVE(wNATIVE).deposit{value: msg.value}();
    assert(IwNATIVE(wNATIVE).transfer(token, msg.value));
    emit LogAnySwapOut(token, msg.sender, to, msg.value, cID(), toChainID);
}
```


- v6与v4版本的区别:



4. anySwapInXxxx(bytes32 txs, address token, address to, uint amount, uint fromChainID) external onlyMPC:

- 参数: txs表示原始链上的交易哈希; token标识用户swap out的真实token对应的anyToken; to表示swap out出来的token要转给谁; amount表示兑换数量(已扣除手续费); fromChainID表示swap的原始链的chainId
- 在目标链上被项目方触发
- 如下图代码所示, 所有的方法都有 onlyMPC 修饰符, 即只有项目方的固定MPC账户可以调用(可以通过调用 function changeMPC(address newMPC) public onlyMPC 来修改)
- 如下图所示, 用户swap out的token的种类有: anyToken、ERC20 token、主网token三种(和swap in相同)



5. depositNative(address token, address to)

- 参数: token标识某一个anyToken, to标识铸造出的anyToken要转给谁
- 函数作用: 向对应的anyToken资金池添加流动性(即用token换取anyToken)
- 代码作用: 对anyToken合约的deposit()进行了封装调用。用户msg.sender存入主网币, 获得对应数量的anyToken(例如, 在eth链, 存入eth, 得到anyWETH)

- 对应于官网中的：



5. `withdrawNative(address token, uint amount, address to)`

- 参数：token标识某一个anyToken，to标识提取出的真实Token要转给谁，amount标识要销毁的anyToken的数量
- 函数作用：从对应的anyToken资金池中移除流动性
- 对anyToken合约的withdraw()进行了封装调用。用户msg.sender销毁amount数量的anyToken，获得对应数量的主网币(例如，在eth链，销毁anyWETH，得到eth)
- 在官网中的调用位置：如上图中的“移除”按钮

6. 其余方法：是对sushiswap中的各种swap()的封装调用，帮助用户将anyToken通过某一条兑换路径进行兑换。具体参考代码逻辑

Multichain中跨链兑换交易示例

- 用户将polygon链的USDT 兑换为BSC链的USDT时，由polygon链、BSC链上的两条交易记录组成：

1. 用户调用polygon链AnyswapRouter合约的 `anySwapOutXxxx()`，将USDT转给项目

- 交易详情：

<https://polygonscan.com/tx/0x943679fbc86e5c823cabd9f0e56c4d697b96648bede558c9b9032ce9addf2341>

- 交易内容：用户将真实的token(例如USDT) 转给对应的anyToken合约(例如anyUSDT)

2. 项目方的MPC会调用BSC链的AnyswapRouter合约的 `anySwapInXxxx()`，将USDT转给用户

- 交易详情：

<https://bscscan.com/tx/0x42924d18c560936e94738d336ebe0e89881d425827058afd2fc6beac32ff746>

- anyToken和真实token的兑换比例：1：1

问题

1. anyToken和真实token的兑换比率是1: 1，那要如何激励用户向anyToken资金池中添加流动性呢？
TODO

二. 跨链项目2--Stargate

Stargate概述

- <https://www.defidaonews.com/article/6736820>
- stargate是第一个在LayerZero协议之上发布的dApp，它允许在不同链之间transfer原生稳定币(USDC、USDT)
- 官网: <https://stargate.finance/pool>
- 开发者文档: <https://stargateprotocol.gitbook.io/stargate/>
- 用户手册: <https://stargateprotocol.gitbook.io/stargate/v/user-docs/>
- stargate提供了Swap、Pool、Farming、Stake四种功能：
 - Pool中各个USDC、USDT池子就是Swap功能所用的资金池
 - 在pool中添加单币的流动性后，stargate会返给用户LP token，然后用户就可以将LP token质押到Farming中赚取STG。STG又可以锁定到Stake中，赚取治理代币veSTG。

Stargate核心原理




- 作为LayerZero上第一个产品，Stargate认为当前的跨链桥存在「不可能三角」，即「资产跨链的到账即时性」、「跨链流动性资金池的统一性」、「跨链转移后的资产原生性」这三个要素，在当前的跨链设施中只能满足一个或两个。
- Stargate号称可以解决这个「不可能三角」——即时的交易确认，即保障用户资产在交易确认时就可以跨到目标链上；统一的流动池，即针对同一资产部署在A、B、C等不同链时，可以共享一个流动性；跨链交互资产的都是原生资产
 - 1. 在交易即时性的实现上，LayerZero采用了超轻客户端技术，即在A链和B链上都植入一个客户端应用，通过预言机和中继器（验证数据）可以传递并即时验证A和B链之间所传递信息的真伪，无需任何中间件，即可保障交易的及时性和无误。
 - 2. 在确保原生资产和流动性的统一性方面，Stargate通过在**每个链上建立原生资金池**来解决，且**允许同一资产之间可以共享不同链的流动性（如何实现的？？）**，以保障资产在跨链时的流动性深度。但这也会遇到风险，如多条链同时从流动池中提取资产时会不会出现兑付危机，或者某条链大额提取造成了流动性枯竭等。为此，Stargate采用了**Delta资源分配平衡算法**来保障原生资产池的平衡，即通过借贷或者套利补充流动性，或者提高提取者的费用，来限制大额提取。

合约地址（以BSC链为例）：

- Router(BSC链上的swap、pool功能的入口):
<https://bscscan.com/address/0x4a364f8c717caad9a442737eb7b8a55cc6cf18d8>

swap 功能概述

- 如下所示，swap功能目前只支持各条链上的USDC、USDT、STG间的兑换，暂不支持其他类型的token。


Select a Token			×
Token Name		Balance	
	USDT	1.95	
	USDC	1.10	
	STG	0.00	

- 每次兑换时，from token和 to token必须一致

Transfer


From

Token


 USDT

▼

Network


 AVAX

▼




To

Token

 USDT

▼

Network

 ETH

▼

Total Amount

Balance: 0.00 USDT

0.028520880692230005

MAX

- 每次非STG 的swap，stargate收取0.6%的手续费（0.045%分配给流动性提供者，0.015%分配给金库）
- 此外，Stargate 转移可能会产生池重新平衡费用

LayerZero协议

- 官网：<https://layerzero.network/>
- 白皮书：https://layerzero.network/pdf/LayerZero_Whitepaper_Release.pdf
- LayerZero是一个Omnichain（全链）互操作性协议，设计用于跨链的轻量级消息传递。LayerZero提供了无需信任、且真实的、有保证的、可配置的消息传递。**LayerZero是由一套低费用（gas-efficient）、不可升级的智能合约实现。**
- 运行机制：
LayerZero在每个支持网络（以太坊、Solana、Polygon等）上部署了一套智能合约，用户必须通过LayerZero合约与各链的合约互动。LayerZero信息传递合约与用户部署的合约通信来传输相应的行动。用户的原始合约基于其选择的网络，比如Polygon，用户可以从任何地方与它互动。这一概念为web3创造了新机会——新的跨链互操作解决方案。（LayerZero信息传递合约通过预言机和中继器

(验证数据)可以传递并即时验证A和B链之间所传递信息的真伪, 无需任何中间件, 即可保障交易的及时性和无误)

- 使用方式: 要发送跨链消息, 用户的合约在源链调用端点 (endpoint) 的send()方法, 然后在目标链调用IzReceive()方法接收消息。为了使用端点合约, 我们需要从 LayerZero 库中导入接口
- endpoint.estimateFees(...), 该函数将返回跨链消息的所需的费用
- 示例项目: <https://learnblockchain.cn/article/3984>

补充: 加密资产托管的三种方案

- 参考:<https://finance.sina.com.cn/blockchain/roll/2022-06-28/doc-imizirav0936161.shtml>
1. 通过热钱包或冷钱包自我保管, 只有一组公私钥, 保存在用户手中
 - 热钱包: 私钥保存在联网的软件中, 如Metamask、TrustWallet
 - 冷钱包: 私钥保存在不联网的设备中
 2. 多重签名钱包: 是一种智能合约, 有多组公私钥, 保存在多个用户手中。在执行前要求多个私钥来签署交易来提高安全性
 3. 第三方托管: 通常是为对冲基金等机构提供存储和管理私钥的安全解决方案。它们通常结合多重签名钱包、HSM 或多方计算 (MPC) 的解决方案。**用户必须相信托管人会保护私钥, 而且相信他们不会恶意使用我们的资金等。**
 - 使用 MPC, 私钥会被共享、加密并分发给多方。这些各方将独立计算他们持有的私钥的一部分以生成签名, 而不会向其他方透露加密。由于整个私钥不会在任何特定的时间停留在任何特定的设备上, 因此 MPC 可以为私钥提供额外的一层保护。