5分钟教你看懂Solana交易

欢迎来到《玩转Solana》系列的第3期。在第1期文章中，我们学习了Solana区块链的运行机制、账户模型和交易结构；在第2期文章中，我们学习了如何使用Rust来编写一个用于发布和展示文章的Solana程序并利用TypeScript脚本对其进行测试。



作为本系列的最后一期，本文将带领大家使用Phalcon Explorer来分析一笔简单的Solana交易。不过在此之前，让我们先来看看Solana上的Token是如何实现的。

# 关于Tokens

Solana上的Token可以分为两类：Native Token和其它Token。

Native Token其实就是Solana Token (SOL)。我们在之前的文章里曾提到，Solana中的每一个账户都拥有一个Lamports字段，Lamports实际上是SOL Token的最小单位（1 SOL = 10亿Lamports），它记录了当前账户Solana Token的余额。

而对于其它Token，Solana使用了一个程序账户（Token Program）以及两种数据账户（Mint Account和Token Account），来实现Token需要的所有功能。

### TokenProgram

TokenProgram结构示意图

Token Program是由Solana Program Library（SPL）提供的一个程序账户，因此AccountInfo的Executable字段显示为True。

正如我们在第一篇文章中介绍的那样，所有程序账户的拥有者都是BPF Loader，Token Program也不例外。此外，和System Program一样，Token Program也被部署在Solana链上的一个固定地址。

作为一个程序账户，它实现了数个指令来提供不同的功能。比如，一些指令被用于创建Mint Account以初始化一种新的Token，或是创建Token Account用于记录某一个地址持有的Token数量；而另一些指令则提供了使用Token需要的所有功能，比如增加Token数量的MintTo指令，或是用于在一对地址之间转移Token的Transfer指令。

需要注意，由于Token Program负责创建Mint Account和Token Account，它还是这两种账户的拥有者。

### MintAccount

Mint Account的结构和关系如下所示：Mint Account是一种数据账户，这意味着其AccountInfo的Executable字段为False。

在Solana上，每一种Token都和一个Mint Account对应。Mint Account记录了该种Token的总供应量，拥有Mint该Token权限的账户地址等信息。

### TokenAccount

Token Account用于记录个体持有某种特定Token的数量。针对某一种Token，每一个持有该Token的账户都拥有一个Token Account。如果某用户拥有5种Token，那么TA将拥有5个Token Account。

Token Account作为一种数据账户，其AccountInfo的Data部分由三个字段组成：

Mint:该Token Account对应的Mint Account的地址；

Owner:有权将Token Account中Token转出的账户，即该Token真正的“owner”；

Amount：当前Token Account持有的Token数量。

TokenAccount结构图

需要强调的是，AccountInfo中的owner字段和Data字段中的owner是完全不同的。前者是Solana中每一个账户都拥有的一个字段，它指明了哪一个地址拥有直接修改当前账户数据的权限；后者则指明了该Token Account所记录的Token实际上是属于谁的，该字段本身是AccountInfo Data字段的一部分。

将上述的内容串联起来，我们便能得到下面这张账户关系的示意图。不管是System Program还是Token Program，它们在区块链上都部署在唯一的地址，以库的形式存在；一个钱包账户可能同时持有多个Token Account，它是这些Token Account真正的“owner”；同一类Token Account的Mint字段指向了该种Token唯一的Mint Account，而该账户则记录了Token的总供应量等信息。

## 在Solscan查看Token账户变化情况

我们可以使用Solscan来查看一笔交易中Token账户的变化情况：

https://solscan.io/tx/byRn8qtNAYSdvgaGCK4kmZV1m89b7uuFuy1cn96W6femp7WgwymLqJ2MP9hPbegqN9EPe7NvghWpqDFqoCDjKph#tokenBalanceChange



Address一栏列出了该交易涉及到的所有Token Account；

Owner一栏则标注了该Token真正的“拥有者”，也就是Token Account Data字段中的owner；

Token一栏则对应了当前Token的MintAccount，我们可以点进第一行的$SON进一步查看：



可以看到，Profile Summary中的Owner Program指明了当前Mint Account的拥有者是Token Program，至此三类账户我们都在Solscan中进行了对应。

不过，Solscan这种展示方法可能会给人带来一些误解。比如，在上一张图中第一行增加的Balance到底属于地址CHS9WajyFfuaAZRk2JC7hRJvPHXmG5fC94gtAPbnLjuY，还是Raydium Authority V4？这种展示方式无疑增加了理解的成本。

除此之外，读者也许还会发现，Solscan中的Solana Token和其它Token在两个板块分开展示。尽管这种分割方式在技术上是合理的，但如果能在展示时将它们统一视作Token放置在相同板块下则更加容易理解。

Phalcon Explorer不仅解决了上述问题，还针对Solscan做了很多其它创新，接下来让我们再使用Phalcon Explorer来看看同一笔交易。

? 建议打开链接，跟随我们的步骤一起分析，这样可以更好地了解交易细节，感受PhalconExplorer的强大功能?

https://app.blocksec.com/explorer/tx/solana/byRn8qtNAYSdvgaGCK4kmZV1m89b7uuFuy1cn96W6femp7WgwymLqJ2MP9hPbegqN9EPe7NvghWpqDFqoCDjKph

# 使用PhalconExplorer查看和分析交易



在Phalcon Explorer的正上方可以看到，当前交易被识别成了JITO的MEV交易，并且可以通过点击交易签名旁的Solana标识来一键跳转到Solscan。

关于这笔交易的信息被分成了四个板块，分别是：Basic Info，Fund Flow，Balance Changes和Invocation Flow，你可以通过点击右上角的图标来切换到相应的部分。

BasicInfo



Basic Info一栏的信息相对简单，它提供了一些关于当前交易的关键信息。和交易签名类似，你也可以点击区块号和签名者地址来跳转到Solscan上。

Fund Flow



Fund Flow一栏为分析者提供了交易执行时产生的资金流转移和时序信息。我们可以看到：

交易的发起者兼签名者59vLEsmV5VCCGTxjHCoRiXkNgHDVcq7dGx98v9HCn2F首先向被标记为Raydium Authority V4的地址转移了一定数量的某种Token；

接着Raydium Authority V4向签名者59vLEsmV5VCCGTxjHCoRiXkNgHDVcq7dGx98v9HCn2F转账了约6.747的Wrapped SOL Token；

最后，签名者还向Jito提供了0.000003的SOL Token作为Jito验证者执行交易的小费。

下图为Solscan中的资金流向图。相较于Solscan，每一个地址在Phalcon Explorer中的资金流向图唯一对应到一个节点，故而能更容易发现地址之间的资金流向关系，提高分析时的效率。



如果你想要对交易的资金流向进行更深入的分析还可以点击右上角的棕色标识进入MetaSleush。

BalanceChanges



Balance Changes一栏为我们清晰展示了不同账户在当前交易结束后所有Token的变化情况。

比如，交易的签名者59vLEsmV5VCCGTxjHCoRiXkNgHDVcq7dGx98v9HCn2F就有三种Token发生了变化，分别是原生的SOL Token、名为61Hh8Udg7zruvG3BhyNiHF4UmULnC8reB9RBFtwi8uKp的Token，以及Wrapped SOL Token。

在Balance Changes中出现的每一个Account Address都拥有一个或多个Token Account，通过点击相应的地址能够对其进行拷贝或跳转到solscan上。

我们可以看到，Phalcon Explorer并没有对SOL Token和其它Token作区分，故而能直接反映出某一账户所有Token的变化情况，并且能更加直观地展示Token Account和其Owner之间的关系。

InvocationFlow



Invocation Flow记录了交易的指令执行流程，其中的每一行都对应了Solana交易执行时的一条指令。

在这里，我们重点关注2和4这两条涉及到Token转移的指令。

第二条指令调用了Raydium的AMM的swapBaseIn来卖掉Token。展开该指令可以看到它由两条CPI（Cross Program Invocation）指令组成，这两条指令的作用是在Raydium Authority V4和交易签名者之间进行Token转移；第四条指令则是签名者向Jito支付小费的过程。

如下图所示，通过点击指令后的Accounts标签，我们可以查看指令涉及到的所有账户。相较于Solscan，这种展示方式会更加简洁，让分析者将注意力放在交易中更关键的信息上。



# 结论

在本文中我们首先介绍了Solana中Token的实现原理，随后使用Solscan查看了一笔交易中Token Account的变化情况。最后，我们使用Phalcon Explorer对该交易作了进一步分析，并介绍了Phalcon Explorer功能上的创新和优化细节。