Miblewimble协议读书笔记

Mimblewimble协议读书笔记

摘要

本文介绍Mimblewimble协议的基本内容，展示协议如何隐藏信息和提高效率，描述协议在区块链系统中的应用场景和应用方式，简要介绍使用该协议的Grin和Beam区块链应用，回答“Grin和Beam中，如何处理签名私钥；怎样解决双花问题；分析对Mimblewimble协议的攻击”三个问题。本文不会涉及椭圆曲线密码的数学细节，对Grin、Beam的应用价值的评估，以及对加密货币的社会属性和发展前景的讨论。

引言

比特币的交易历史是在区块链网络上公开的，任何人可以通过考察整个交易历史来追溯任何一笔交易数据。但这种基于交易的记账方式会带来比较严重的效率问题：任何人要考察一笔交易是否合法都需要检查交易的输入是否是没有被花掉的，而这可能需要追溯很多笔之前的交易；任何一笔交易都在销毁一笔之前的交易。这造成目前的所有交易中只有极小一部分是未被花掉的。

这个现象会给挖矿者带来严重的效率问题——尤其是针对于没有建立受信任UTXO的新挖矿者来说；不仅如此，任何人想要完整地验证一笔交易都会花费高昂的代价。

同样的原因带来的另一个危险是比特币的每一项交易的输出被用来做了什么都是暴露的，这使得交易历史可以被分析。

2016年8月2日，一个匿名作者使用名字Tom Elvis Jedusor公布了Mimblewimble白皮书[1]（与其说是白皮书，更像是一篇技术随笔），描述了Mimblewimble协议并抛出了一些问题；2016年10月6日，自称Andrew Poelstra的作者公布了另一篇同样名为Mimblewimble的白皮书[2]，指出了原版白皮书中存在的一些问题，回答了原版白皮书抛出的问题，被称为“精确版”Mimblewimble白皮书。本文的内容也主要总结于这两篇文章。

Mimblewimble协议给出了以上两个问题的解决方案，在提供信息隐藏能力的同时解决了部分效率问题。

基础概念

除了比特币已有的技术，Mimblewimble协议的构造用到了椭圆曲线密码学技术和Bulletproofs技术。本章对两种技术作简要介绍，着重于技术的特性和应用，而不会过多介绍背后的数学原理。

椭圆曲线

简要对椭圆曲线密码学进行介绍，阐述Mimblewimble会用到的椭圆曲线特性[3]。

用于密码学目的的椭圆曲线只是一大组我们称之为 C 的点。这些点可以被加、减或乘以整数（也称为标量）。 给定一个整数 k 并使用标量乘法运算，我们可以计算 `k * H`，这也是曲线 C 上的一个点。 给定另一个整数 j，我们也可以计算 `(k + j) * H`，它等于 `k * H + j * H`。 椭圆曲线上的加法和标量乘法运算保持加法和乘法的交换律和结合律：

```
(k+j)*H = k*H + j*H
```

如果我们选择一个非常大的数字 k 作为私钥，则 `k * H` 被作为相应的公钥。 即使人们知道公钥 `k * H` 的值，推导 k 几乎不可能（或者换句话说，椭圆曲线点的乘法计算是微不足道的，然而曲线点的“除法”计算却极其困难。

Bulletproofs

Bulletproofs是一种新的零知识、无交互的交易证明协议。Mimblewimble主要利用了Bulletproofs作为核心的范围证明技术。

简单的说，范围证明可以使得在不暴露数值 `x` 的情况下证明 `x` 落在某个范围内，例如 `x > 0 && x < 100000`。

基本构造

在阐述Mimblewimble的基本构造之前，要先移除已经存在的区块链网络环境，包括比特币脚本，因为它们的内容过于具体和复杂。后面将会解释，基于比特币脚本进行交易混合基本是不可能做到的。但无论如何，Mimblewimble都是一种对比特币的优化方法；基于Mimblewimble协议的其它区块链系统的介绍会在下一章呈现。

Pederson Commitment

Mimblewimble协议中，交易内容的编码是交易金额v的Pederson Commitment，编码方式如下：

```
C = r*G + v*H
```

其中，v是交易金额，r是致盲因子，G和H是被选择的椭圆曲线点（准确地说，是椭圆曲线加密函数生成的固定的值）。

Pederson Commitment构造保证了两点：

- 交易可以隐藏具体交易数值，并且在不知道具体交易数值的情况下完成验证
- 拥有私钥即拥有输出的所有权

接下来对这两点依次进行说明：

隐藏交易

考虑一笔交易的输入是`v1`和`v2`， 输出是`v3`，那么验证交易的零和是容易的，只需验证以下等式即可：

```
v1 + v2 = v3
```

以明文编码提供不了任何隐秘性。要隐藏具体数值，最容易想到的做法是引入一个椭圆曲线点`H`作为因子。现在，交易数值不变，但编码分别称为了`v1*H`、`v2*H`、`v3*H`。我们知道“除法”的计算相当困难，所以可以认为通过`v1*H`难以反推出`v1`，交易数值被隐藏了；零和验证依然是容易的：

```
v1*H + v2*H = (v1 + v2)*H = v3*H
```

在这种情况下，`H`在每一个交易中都需要使用，是公开的。所以攻击者可以通过尝试每个可能的`v1`值来计算出真正的`v1`。此时问题不在于`v1`的取值范围有多大，而在于一旦`v1`被破解，所有等于`v1`的交易由于拥有相同的编码，也一并被破解了。

Pederson Commitment解决该问题的奥秘之处在于引入了另一个椭圆曲线点`G`， `r`是一个私钥，由发起交易的人自己选择，完全可以和交易值`v`无关。交易的编码分别为：

输入：`v1*H + r1*G`，`v2*H + r2*G`

输出：`v3*H + r3*G`

Pederson Commitment的加法同态性使得零和验证依然是容易的：

```
(v1*H + r1*G) + (v2*H + r2*G) = (v1 + v2)*H + (r1 + r2)*G = v3*H + r3*G
```

这里要注意的是，要保证上式成立，需要有`r1 + r2 = r3`。毕竟这三个致盲因子来源于不同的交易，如何保证`r1 + r2 = r3`？我们在下一节所有权部分回答这个问题。

Pederson Commitment具有良好的性质：在知道`v*H + r*G`的情况下，不能推出`v`和`r`；而因为两个数值毫无关系，能满足等式的组合也可能不止一种，暴力破解也不可能；因为每一笔交易的`r`都不同，也不用担心一个交易被暴露后相似的交易也会暴露的问题。

所有权

上一节引入了致盲因子`r`来隐藏交易，我们知道`r`同时也是私钥的一部分。每一笔交易产生后一个必要的作用就是让交易输出的接受者能够使用这笔钱。Mimblewimble中，这个致盲因子`r`就可以被用来证明钱的所有权。

假设Alice给Bob转了一笔钱`v1`，这笔交易的记录为：

```
C1 = v1*H + r1*G
```

其中`C1`是公开的；`r1`只有Alice知道；`v1`是转账金额，只有Alice和Bob两个人知道。

接下来，Bob要使用这笔钱，假设他将数值同样为`v1`的钱转给了某个人，钱的来源是`C1`这笔交易。`C2`交易的记录为：

```
C2 = v1*H + r2*G
```

其中`r2`是一个私钥，只有Bob自己知道。现在要验证这笔交易的零和性，应当有`C1 = C2`，可是我们却得到了`r1*G = r2*G`。这是不成立的，因为`r2`没有理由和`r1`相等，Alice也不可能将`r1`告诉Bob（或者说Bob不敢使用Alice给他的致盲因子，因为这样会导致自己的钱被Alice花掉）。既然`r1`和`r2`不想等，那么我们可以计算`C1`和`C2`的差值：

```
E = C2 - C1 = (r2 - r1)*G
```

其中`r2 - r1`被称为excess value。此时，Bob在交易中公开`E`值（具体公开方式应该选择对`E`进行签名，这样就可以验证`E`确实是Bob公开的），零和验证过程就变成了：

```
C1 + E = v1*H + r1*G + (r2-r1)*G = v2*H + r2*G = C2
```

在此过程中，椭圆曲线密码学保证了只有`v2 - v1 = 0`时等式才会成立，即`E`一定是`G`上的有效公钥。

交易费

有了上述Pederson Commitment构造，在交易中加入一笔交易费就很容易了。和比特币有交易费一样，只需要明确一个具体的交易费数值`vt`，在验证时将它乘上`H`再减掉就可以了。假设输入是`C1`，输出是`C2`，验证过程：

```
C1 + E = v1*H + r1*G + (r2-r1)*G = (v2+vt)*H + r2*G = v2*H + r2*G + vt*H =
C2 + vt*H
```

这样，这笔交易还可以当作矿工以后交易的输入值。

范围证明

在以上的讨论中，我们都假设交易值都是正数。现在考虑这样一笔交易（这个例子来自Grin项目的文档[4]）：输入为2，输出为两笔钱，分别是5和-3。这笔交易在上述的构造中完全是合法的，如果我们忽视-3的账单，将多出来的3个单位的钱汇入自己的账户，就凭空多出来了3个单位的钱。这是非常危险的。

Mimblewimble解决此问题的方案是引入范围证明。范围证明保证了交易的数值`v`落在一个合法的范围内。一般来说，我们希望交易数值是一个正数并且不会溢出，那么我们给出一个`v`在范围`[0, 2^64]`的范围证明`range_proof`，验证方只需要知道`v*H + r*G`和`range_proof`就能在不知道`v`的情况下验证`v`的确属于范围`[0, 2^64]`。

总结

上述所有构造可以总结为记录到区块内的一个交易的具体内容：

- 一组输入：`vi1*H + ri1*G, vi2*H + ri2*G, ...`
- 一组输出：`vo1*H + ro1*G, vo2*H + ro2*G, ...`
- 和输出相关的范围证明：`range_proof_vo1, range_proof_vo2, ...`
- 交易费：`vt`
- excess value的签名：`E`

和验证过程：

- 验证每个`vo`的范围都是合法范围
- 验证`E`有交易发起者的签名
- 验证输入输出是零和的

高级构造

将上一章以Pederson Commitment为基础的基本构造巧妙地隐藏了交易数值。本章介绍Mimblewimble的实现Grin如何通过cut-through技术来提高效率和提高匿名性。

在叙述本章内容之前，方便后面内容的讨论，对基础构造内容作一个简单的抽象：将一笔交易所有输入抽象为`inputs`，所有输出抽象为`outputs`，所有的范围证明、交易费、excess value等用于验证的信息抽象为`kernel_excess`。

验证过程就简化为：

```
sum(inputs) - sum(outputs) = kernel_excess
```

交易聚合

对于多笔交易，比如一个区块里的所有合法交易，可以看成一个交易序列。将这些交易的所有输入进一步抽象为`inputs`，所有输出抽象为`outputs`，所有`kernel_excess`抽象为`kernel_excess`，如下等式成立：

```
sum(inputs) - sum(outputs) = sum(kernel_excess)
```

也就是说，交易聚合后仍然是合法的。

我们希望交易聚合后能达到“混合”的效果，区块的内容只能展示输入有哪些以及输出有哪些，而不能通过公开的信息推出哪些输入对应哪些输出。上面关于聚合的构造还不能做到这一点，因为攻击者完全可以通过枚举输入和输出来尝试通过验证，这样就很容易将某些输入和输出匹配。Mimblewimble利用kernel_offsets[5]技术解决了这一问题，这里不详细说明。

Cut-through

考虑在同一区块内有这样两笔交易：Alice给Bob转了`v1`；Bob又给Ted转了`v1`， 花的正是Alice转来的这笔钱。如果这两笔交易都通过了验证，那么在区块里可以只记录这样的一笔交易：Alice给Ted转了`v1`。这样做就完成了关于Bob的两笔交易的隐藏，同时减小了空间消耗。

更进一步地，可以对一个区块内的所有交易都做这样的处理，使得最终区块看上去只有一系列的输入和一系列的输出，交易的结构完全被消除了，而且输入和输出的顺序也变得无关紧要。同时，零和验证依然可以正常进行。交易结构的消除使得匹配输入和输出称为不可能。

区块链级别的优化

在引言部分提到，比特币系统的一大问题在于交易历史过于臃肿，只有极少一部分的输出是没有被花掉的。尽管区块链是不更改的，UTXO对于全节点而言仍然是一个一致的集合。区块链的状态总是可以被代币总量+UTXO+每笔交易的kernel_excess刻画，以目前的构造每一项都是易于计算和验证的，而且所占空间非常小。这就意味着新的全节点不需要再像比特币的全节点那样输入巨大的交易历史并计算UTXO。

总结

Grin作为Mimblewimble一个较为成熟的实现，使用cut-throught技术缩小了必要账本的体积，同时提高了匿名性。可以看出Grin非常看重匿名性。

基本应用

基于Mimblewimble的大项目有两个：Grin和Beam。Beam在2019年1月3日主网上线，Grin是在1月15日。对这两个项目的介绍不是本文的重点内容，这里只展示一段我对两个项目的对比总结。

Grin坚持完全交给开源社区管理，Beam则是成立公司；Grin拥有更广为人知的出色的技术，Beam在技术上低调很多，也承认他们一直在向Grin学习；Grin团队明确选择放弃扩展除核心功能之外的其他功能，Beam提出可审计钱包，允许企业或其他用户生成独有的公/私钥对，密钥对可以让外部第三方验证资金或其他任何存储在交易中的元数据；Grin更理想主义，Beam更现实主义。

Beam的货币和很多古老的加密货币一样，总量是有限的。Grin的货币则更大胆：每分钟挖出60个，永远如此！这是典型的通胀货币政策，它会让Grin更像是现实货币（拥有应付通胀能力，更加稳定），而不是黄金。这意味着Grin几乎没法割韭菜。事实上，Grin社区的态度也非常鲜明，他

们拥有如此高的技术，却没有创始人奖励、没有ICO、不接受公司或基金会的控制，比比特币更加理想化。可以这么说，Grin的胜利将会是理想主义的胜利，同时也是比特币、去中心化、无政府主义的胜利。但就目前看来，Grin离他们想要的胜利还非常遥远。

Grin最大的问题不在于能保护多少隐私，而在于有多少人用！目前的情况是，Grin的区块链中有大量的区块只包含一条交易，这使得第三章“高级构造”中提到的cut-through技术没有用武之地。越来越少的关注会使得Grin币值缓慢下降，而不是像其它的总量有限的冷门货币那样至少可以保值。

最后我想说，正因为Grin的这些理想化又似乎在现实中受冷落的特点，我们才不能小看它。加密货币的价值来源是人们的信仰。“韭菜收割者”正是看中了这一点，一番炒作下，近几年的币圈才会出现巨大的泡沫同时又出现很多批判的声音。Grin的各种政策表明它不是一个能被“炒火”的币，它的价值是失去了“韭菜收割者”的支持后剩下的，或许不会突然飞黄腾达，但凝聚下来的都是顽强的、憧憬无政府、无规则社会的理想主义。

问题

本章解答作业中提出的三个问题。

签名私钥的处理

答案在第二章”基本构造“中，私钥体现在每笔交易中由交易发起者选择的致盲因子`r`。致盲因子的作用包括：隐藏交易、防止暴力破解、证明所有权。致盲因子`r`最终作为签名体现在交易的excess_kernel中。

双花问题

双花问题的解决方案在于第二章“基本构造”的excess value。和比特币系统类似，每一笔交易都包含交易发起者的签名（excess_kernel），通过UTXO的验证即可发现交易的输出是否已经被花掉；更进一步的，在第三章“高级构造”的Grin系统中，区块的交易经过聚合，输入序列和输出序列的总值想等。如果出现双花，必然会很容易发现不等的情况。

对Mimblewimble协议的攻击

在第二章到第三章提出了很多种攻击模型，分别是：

- 针对交易值`v`的暴力破解
- 利用已暴露的交易数值破解其它交易
- 通过共享致盲因子窃取别人的钱
- 通过构造负数或者溢出数值凭空造钱
- 通过拼凑输入序列和输出序列的组合来匹配具体交易
- 通过篡改UTXO来进行双花

在攻击模型提出的相应部分都给出了具体应对方案。

参考文献

（本文定价0元）

---

1. "MIMBLEWIMBLE" [online]. Available: https://download.wpsoftware.net/bitcoin/wizardry/mimblewimble.txt ↩
2. "MIMBLEWIMBLE" [online]. Available: https://download.wpsoftware.net/bitcoin/wizardry/mimblewimble.pdf ↩
3. https://zh.wikipedia.org/wiki/椭圆曲线密码学 ↩
4. "Introduction to Mimblewimble and Grin" [online]. Available: https://github.com/mimblewimble/grin/blob/master/doc/intro.md ↩
5. "Mimblewimble Non-Interactive Transaction Scheme" [online]. Available: https://eprint.iacr.org/2020/1064.pdf ↩