

# 读懂区块链之零知识证明（zk-SNARK）



逐舞传歌 (/u/76eda941ad11) [+ 关注](#)

2018.09.06 11:18 字数 11324 阅读 1782 评论 13 喜欢 11

(/u/76eda941ad11)

## 零知识证明：从小白到明白

如今，知识快餐业发达，区块链这么火的领域自然不会落下。经过一轮轮扫盲，共识、工作量证明、闪电网络等等概念对普罗大众已不再陌生，甚至各种解构、比喻、引申，将术语炒得比本义还玄乎。然而，如果不理解甚至没听说过零知识证明，那你基本还属于区块链小白。

之所以这么说，原因有二。其一，零知识证明是代数数论、抽象代数等数学理论的综合应用，与闪电网络一类的精巧设计不同，属于硬技术。如果不是数学科班出生，它引入的概念、符号会让人眼花缭乱。V神（以太坊ethereum的创始人。本名Vitalik Buterin，对于国人而言，这名字读起来有点困难，同时考虑到这位90后小哥创造以太坊时才19岁，因此币圈尊称其“V神”）在一篇介绍零知识证明的文章中就提醒读者看不懂也不要怀疑自己的智商，因为“它实在是太难了！”因此，能坚持搞懂的，一定是意（固）志（执）力有异常人。呵呵，过奖了o(\*￣▽￣\*)o！

其二，零知识证明解决了区块链应用的一个根本问题。区块链的应用价值在于去中心化共识，无论是货币交易还是权益证明，所有成员都是见证人，众目睽睽之下，一旦交易完成便无法抵赖，权益生成便不能否认。但这也意味着，你收了谁多少钱，名下有几套房对于所有人都是透明的，这显然与隐私保护的刚性需求相违背。虽然一些针对隐私保护的混淆技术被提出，例如达世币（Dash）的PrivateSend，但终究只能在一定程度上缓解而不能根治去中心化引入的这个固有问题。好在世上从来不缺聪明人，没让大家等多久，零知识证明横空出世（严格说来，零知识证明提出应早于区块链技术，但无疑是后者让其为世人所知）。零知识证明要解决的问题是：**以不透露一个论断（statement）的任何信息为前提，向你证明这个论断是对的**。以货币交易为例，就是在不告诉你付款人、收款人是谁，也不告诉你金额多少的前提下，设法证明这笔交易是合法的。o((◎\_◎))o这怎么可能呢？！

这是有可能的。图1是一个经常用来科普零知识证明的例子。图中所示是一个山洞，入口处有两条路A和B，而这两条路在山洞深处被一道门给隔开了，只有说出开门魔咒门才能打开。这里涉及两个角色P（Proofer）和V（Verifier），P试图向V证明，他知道开门魔咒；如果属实，V就饶ta不死。P自然不能直接将魔咒告诉V，因为万一ta知道后把自己干掉怎么办；V则一定不会轻易相信P。他们可以这么做：

1. P从A、B两条路中随机选择一条走进去；这时，V在洞外等着，对P选择了哪条路一无所知；
2. 等待足够长时间后，V进入山洞，然后也从A、B中随机选择一个并且大声喊出来，譬如，“B！”；
3. P听到V的声音后便从对应的那条路走出来。如果P确实知道开门魔咒，那么无论自己和V分别选择的是A还是B，P都能正确地从V报出的路走出来。相反，如果P不知道魔

(https://dsp  
click.youda  
slot=30edc  
dc09-44da  
532ae520i

(http://e.cn



咒，那么ta只有1/2的概率会做到。而从V的角度来说，如果ta看到P从正确的路出来了，ta便有50%的把握肯定P确实知道魔咒；

4. 将第1-3步重复N次，如果P每次都能做对，那么V便有 $1-(0.5)^N$ 的把握相信P。例如，N=5，可靠性就是96.9%，已经足够好了。更重要的是，V对于魔咒仍然一无所知。这便是**零知识证明**。

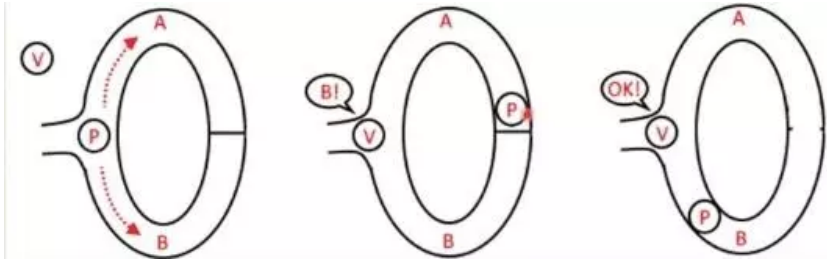


图1 零知识证明示例：我知道开门咒语

(https://dsp  
click.youda  
slot=30edc  
dc09-44da  
532ae520i

有点意思，对吧？可能你会觉得这个例子不够“严肃”，太喜剧化完全不像讲技术。那好，再来一个例子-著名的“三色图问题（graph tree-coloring）”，见图2。所谓三色图问题就是找到一种上色方案使得相邻两个节点的颜色不同（子图(1)中以连线表示两个节点相邻）。三色图是一个NP问题（NP是啥玩意？别急，后文会详细解释），这意味着求解过程十分复杂。因此，当Anna应Carl的委托好容易找出一个三色图问题的答案，没拿到报酬是绝不愿意出示答案的。那作为Prover角色的Anna如何向Carl证明她知道答案呢？步骤如下：

(http://e.cn

1. Anna把问题图中已经正确上色的节点都遮起来，见子图(2)；
2. Carl从中任意选相邻的两个节点，Anna便向其展示这两个节点的颜色，见子图(3)。如果这两个节点的颜色不同，那么Carl便有50%的把握相信Anna确实知道答案；
3. 接着，Anna随机选择一种颜色映射方案将目前图中的颜色变换成另一种颜色，例如“紫色->白色，橙色->黄色，绿色->黑色”，这样便生成了一张新的上色图。虽然颜色不同，但仍是原问题的有效解。接着重复第1-3步N次，如果每次展示的两个节点颜色都不相同，那么Anna知道答案的可靠性便是 $1-(0.5)^N$ 。

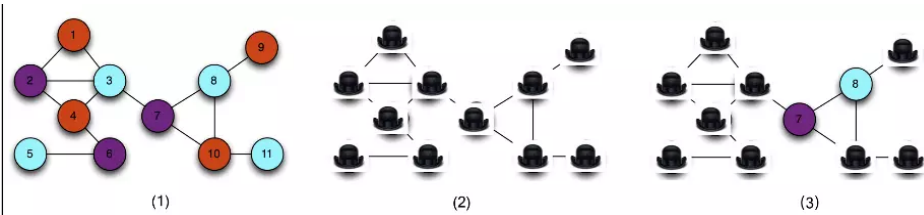


图2 零知识证明示例：三色图问题

类似的例子还可以举出不少，例如数独、finding waldo（在一幅密密麻麻都是人的图片中找到某个特定人物的游戏）。找到点感觉了吧？这就是零知识证明。

那zk-SNARK又是啥，和零知识证明什么关系？好，接着往下看。

zk-SNARK：到底是什么鬼？

zk-SNARK是“zero knowledge Succinct Non-interactive ARgument of Knowledge”的缩写，这一长串名字的主体是“argument of knowledge”，即“知情证明”，也就是掌握某事内幕的证据。修饰主体名词的定语由三部分组成，分别代表了此技术要解决的三个问题，



分别是：

- 1. zero knowledge：零知识，即在证明的过程中不透露任何内情，如上文的例子所示；
- 2. succinct：简洁的，主要是指验证过程不涉及大量数据传输以及验证算法简单；
- 3. non-interactive：无交互。上文中举的两个例子虽然实现了零知识证明，但Prover和 Verifier之间需要经过多次交互才能取得满意的可靠性，而此技术试图彻底避免这些交互。

合起来，zk-SNARK是一种“证明我知道内情的技术，简单、易操作，最关键的是你除了“我是对的”啥也不会知道”。

ZCash（大零币，货币符号是ZEC。本文写作到此处时，ZEC的单价为\$223）是最早广泛应用zk-SNARK的数字货币。ZCash采用zk-SNARK技术，目的是彻底解决交易被追踪从而暴露用户隐私的问题。如果上文的例子让你觉得是toy example，那么结合ZCash便能更确切地理解zk-SNARK的应用场景。

zk-SNARK的实际应用：ZCash

ZCash的核心概念与比特币是一脉相承的。当然，这不奇怪，谁让比特币是“初代吸金鬼”呢？（注：如果读者对比特币基本原理还不清楚，那基本属于走错片场了，请先阅读比特币原理 (https://www.jianshu.com/p/63dfc23c25d5)一文）。简单回顾一下比特币交易：

- 1. 一个比特币交易（Transaction）接受若干输入（Transaction Input, TI），同时产生若干输出（Transaction Output, TO）；
- 2. TI和TO是相对一个特定交易而言的，因为一个交易的TO可能成为另一个交易的TI，这是一个将挣来的钱再花出去的过程；在还没花出去前，这些钱就是“Unspent”的，因此此刻尚未成为下一个交易TI的TO称为“UTXO(Unspent Transaction Output)”。
- UTXO是比特币交易的基本单元；
- 3. 交易的付款方需证明自己有权使用这些UTXO，方法是提供私钥进行验证，因为每个交易TO会指定收款人的公钥，保证只有收款人才能接着花它。

ZCash继承了比特币的交易模型，只不过UTXO被衍生出的新概念“note”所代替，后者是ZCash的基本交易单元。英语中，“note”有“钞票”的意思。不过，翻译成“支票”更贴切，因为每张note上都标注了只有谁才能兑现它（即所有者）。一个交易的输入和输出都是若干note。为描述方便起见，将note记为“note=(PK, v, r)”，其中，PK是所有者的公钥（地址），v是金额，而r是可以唯一区分该note的序列号。

所不同的是，ZCash交易分为两类：透明地址和隐藏地址。透明地址交易的输入、输出直接是可见的note信息，一个例子如图3所示（除了货币单位外，和比特币交易一模一样）。



图3 ZCash交易例子：透明地址

而对于隐藏地址交易，输入和/或输出的地址和金额是隐藏的。例如图4展现了一个真实的ZCash交易。其中，输入和输出两栏为空，这表示地址未知（并非没有输入和输出）；另外，交易的总金额只知道“≥ 0.0001 ZEC”，具体金额未知（0.0001ZEC是交易费，用以付给矿工的）。透明地址和隐藏地址还可以混用。例如图5所示，输入，即钱来自哪是未知的；而对于输出，已知其中约20ZEC转给了地址 t1KvwZC29AWv21tNzqoGPfcX8242j5XxFf8，其它去向未知。

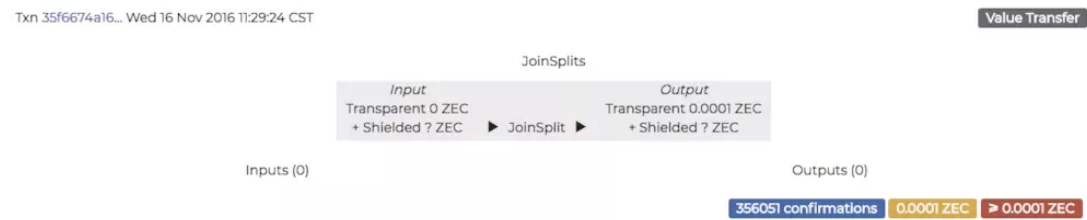


图4 ZCash交易例子：隐藏地址

(<https://dsr.click.youdao.com/slot=30edcdc09-44da532ae5207>)



图5 ZCash交易例子：混用透明和隐藏地址

(<http://e.cn>)

在隐藏地址的交易中，输入、输出不再是明文的note，而分别是note的废止通知和签发通知：

- 签发通知（note commitment）：作为交易的输出，表示一张新note被签发。一个有效的commitment是一张note存在的证明，然而从它包含的信息中并不知道是哪张note，也就无法知道所有者是谁，金额多少。为满足这一点，最简单的方法是对note的描述信息取哈希，因此note对应的commitment可以简单描述为“HASH(note)”；
- 废止通知（note nullifier）：作为交易的输入，表示一张老支票将作废（因为马上要被兑现、花掉了）。同比特币一样，一个交易的输入一定是另一个交易的输出，因此nullifier对应唯一——一个commitment（结合commitment的定义，也就唯一对应一张note），但从它包含的信息并不能推导出是哪个commitment（如果可以的话，ZCash交易便可被追踪，因而丧失隐私性了）。为构造满足要求的nullifier，取哈希依然是个好办法，因此序号为r的note，对应的nullifier可描述为“HASH(r)”。

通过引入nullifier和commitment，交易之间路人皆知的关联变成了付款人和收款人的心照不宣，如图6所示。



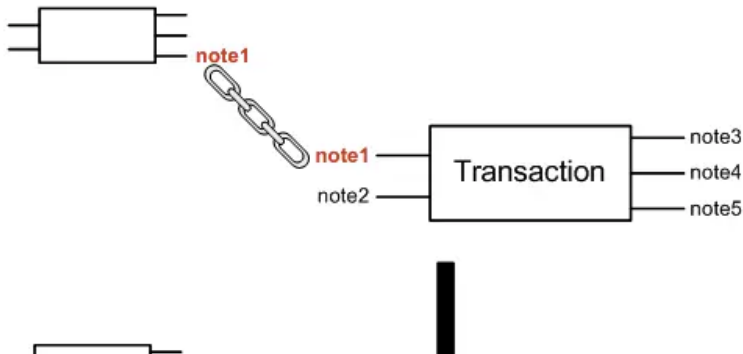


图6 nullifier与commitment的秘密映射

ZCash区块链共识的所有参与者（节点）各自维护一个nullifier和commitment的集合，随着新交易的产生，这两个集合的内容会不断变化。下面介绍一下这个过程。假设当前已存世3张支票：note1=(PK1,v1,r1), note2=(PK2,v2,r2), note3=(PK3,v3,r3), 其中note1属于Anna, note2已经被花掉了。此时各节点维护的nullifier和commitment集合内容如表1所示。

(https://ds  
click.youda  
slot=30edc  
dc09-44da  
532ae520i

(http://e.cn

Commitment Set	Nullifier Set
C1 = HASH(note1)	NF1 = HASH(r2)
C2 = HASH(note2)	
C3 = HASH(note3)	

表1 支付前的commitment和nullifier集合

Anna决定将金额为v1的note1转给Carl, 他的公钥/地址是PK4, 她将这么操作：

1. 随机挑选一个序列号r4, 并以此产生note4 = (PK4, v1, r4);
2. 秘密地将note4发给Carl;
3. 将note1的nullifier, 即nf2 = HASH(r1), 以及新产生的note4的commitment, 即其哈希值HASH(note4)广播给所有节点。

收到Anna广播的节点, 会**检查nf2是否已存在于nullifier集合中**；若没有, 说明对应的支票没有重复兑现, 节点会将HASH(note4)和NF2分别加入到所维护的commitment和nullifier队列中, 如表2所示。nullifier所起的作用就是防止数字货币被“重复支付”的基础问题（我不喜欢将“double spend”翻译成“双花”, 总感觉像在讨论植物学）。

Commitment Set	Nullifier Set
C1 = HASH(note1)	NF1 = HASH(r2)
C2 = HASH(note2)	NF2 = HASH(r1)
C3 = HASH(note3)	
C4 = HASH(note4)	

表2 支付后的commitment和nullifier集合

^

🔗



这就是ZCash的原理...噢，等等，不对吧？！怎么知道Anna给的NF2对应的支票存在真的存在，万一她精心选择的垃圾数据怎么办？就算NF2确实指向一张支票，那怎么知道Anna有权兑现它呢？Anna自然可以通过公布note1的内容来证明，但如此一来，她的小秘密就大白于天下了。啊哈~~，我们的零知识证明这时就能排上用场：Anna会同时提供一份凭证  $\pi$ 。 $\pi$ 足以证明提供人（这里值Anna）知道能满足以下条件的PK1，sk1（PK1对应的私钥）和r1的值：

- 1. 用PK1、r1复原的note数据结构，其哈希值存在于commitment集合中 → 用以支付的note是有效的；
- 2. sk1是PK1的公钥 → Anna有权使用这张note；
- 3. HASH(r1) = NF2 → nullifier与commitment一致。

其他节点在验证 $\pi$ 有效后才承认此次交易合法。同时，它们无法从 $\pi$ 推断出有关PK1、sk1和r1的任何信息。

恭喜你，读到这，你应该基本清楚零知识证明、zk-SNARK是啥，能解决什么问题了。但究竟如何做到的还未交待，zk-SNARK仍是谜一样的存在。子不语怪力乱神。吾辈岂甘浅尝辄止，必要一睹其中机巧为快。因此，继续！

友情提示：前方深水区，请注意安全，量力前往！:)

(https://ds  
click.youde  
slot=30edc  
dc09-44da  
532ae520i

(http://e.cn

zk-SNARK原理大起底

zk-SNARK背后的原理相当复杂，如果没有一张地图，很容易在不断涌现的概念中迷失。因此，为了便于理解，下文将按照如图7所示的线索来展开。

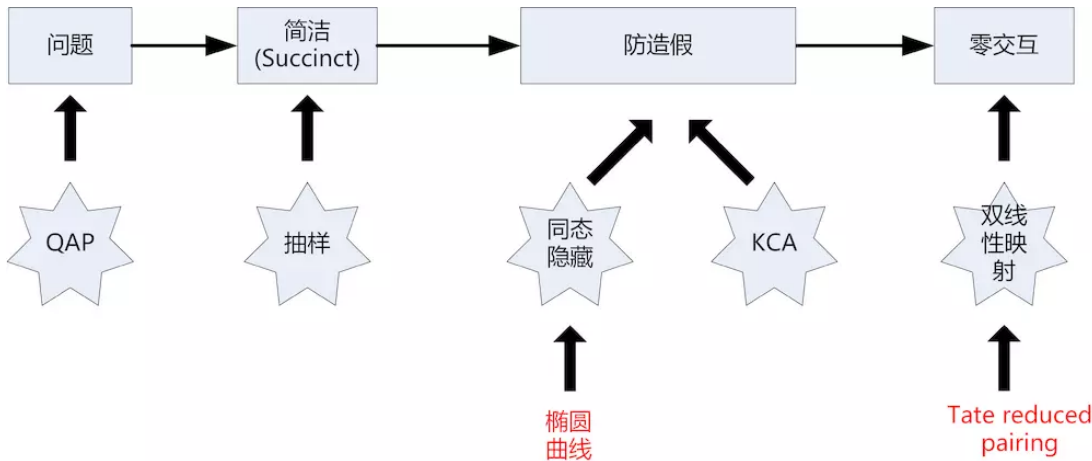


图7 zk-SNARK原理

1. 将计算问题描述成一个QAP

zk-SNARK作为一种数学方法，必须有可量化的输入，证明过程是严谨的数学推导，因此在运用zk-SNARK之前，首先得为目标问题建立一个数学描述模型。例如，对于ZCash，一个交易（严格地讲，是一个交易中包含的JoinSplit描述，相当于一个子交易）可以用以下信息来描述：

^

🔗

对所有人公开:

$(rt : \mathbb{B}^{[\ell_{\text{Merkle}}]},$   
 $nf_{1..N^{\text{old}}}^{\text{old}} : \mathbb{B}^{[\ell_{\text{PRF}}][N^{\text{old}}]},$   
 $cm_{1..N^{\text{new}}}^{\text{new}} : \text{COMM}^{\text{Sprout}}.\text{Output}^{[N^{\text{new}}]},$   
 $v_{\text{pub}}^{\text{old}} : \{0..2^{\ell_{\text{value}}}-1\},$   
 $v_{\text{pub}}^{\text{new}} : \{0..2^{\ell_{\text{value}}}-1\},$   
 $h_{\text{Sig}} : \mathbb{B}^{[\ell_{\text{hSig}}]},$   
 $h_{1..N^{\text{old}}} : \mathbb{B}^{[\ell_{\text{PRF}}][N^{\text{old}}]}).$

仅发起交易者知道:

$(\text{path}_{1..N^{\text{old}}} : \mathbb{B}^{[\ell_{\text{Merkle}}][\text{MerkleDepth}][N^{\text{old}}]},$   
 $\text{pos}_{1..N^{\text{old}}} : \{0..2^{\text{MerkleDepth}}-1\}^{[N^{\text{old}}]},$   
 $n_{1..N^{\text{old}}}^{\text{old}} : \text{Note}^{[N^{\text{old}}]},$   
 $a_{\text{sk},1..N^{\text{old}}}^{\text{old}} : \mathbb{B}^{[\ell_{\text{aSk}}][N^{\text{old}}]},$   
 $n_{1..N^{\text{new}}}^{\text{new}} : \text{Note}^{[N^{\text{new}}]},$   
 $\varphi : \mathbb{B}^{[\ell_{\varphi}]},$   
 $\text{enforceMerklePath}_{1..N^{\text{old}}} : \mathbb{B}^{[N^{\text{old}}]}).$

图8 ZCash交易的描述信息

而一个交易合法需要满足以下条件:

**Merkle path enforcement** for each  $i \in \{1..N^{\text{old}}\}$ , if  $v_i^{\text{old}} \neq 0$  then  $\text{enforceMerklePath}_i = 1$ .

**Balance**  $v_{\text{pub}}^{\text{old}} + \sum_{i=1}^{N^{\text{old}}} v_i^{\text{old}} = v_{\text{pub}}^{\text{new}} + \sum_{i=1}^{N^{\text{new}}} v_i^{\text{new}} \in \{0..2^{\ell_{\text{value}}}-1\}.$

**Nullifier integrity** for each  $i \in \{1..N^{\text{old}}\}$ :  $nf_i^{\text{old}} = \text{PRF}_{\text{aSk},i}^{\text{nf}}(\rho_i^{\text{old}}).$

**Spend authority** for each  $i \in \{1..N^{\text{old}}\}$ :  $a_{\text{pk},i}^{\text{old}} = \text{PRF}_{\text{aSk},i}^{\text{addr}}(0).$

**Non-malleability** for each  $i \in \{1..N^{\text{old}}\}$ :  $h_i = \text{PRF}_{\text{aSk},i}^{\text{pk}}(i, h_{\text{Sig}}).$

**Uniqueness of  $\rho_i^{\text{new}}$**  for each  $i \in \{1..N^{\text{new}}\}$ :  $\rho_i^{\text{new}} = \text{PRF}_{\varphi}^{\text{p}}(i, h_{\text{Sig}}).$

**Note commitment integrity** for each  $i \in \{1..N^{\text{new}}\}$ :  $cm_i^{\text{new}} = \text{NoteCommitment}(n_i^{\text{new}}).$

图9 ZCash交易合法的条件

这两大坨看不懂没关系，就没准备让各位看懂，有个基本概念就好：一组变量的特定输入值代入目标问题对应的若干计算方程后，能使它们成立，这些输入称为问题的“解”。zk-SNARK要应对的就是如何证明“我知道解”。

zk-SNARK只适合特定形式的计算问题，即所谓QAP (Quadratic Arithmetic Programs)。不要纠结于这个术语，笼统地说，就是计算方程中需要出现多项式。

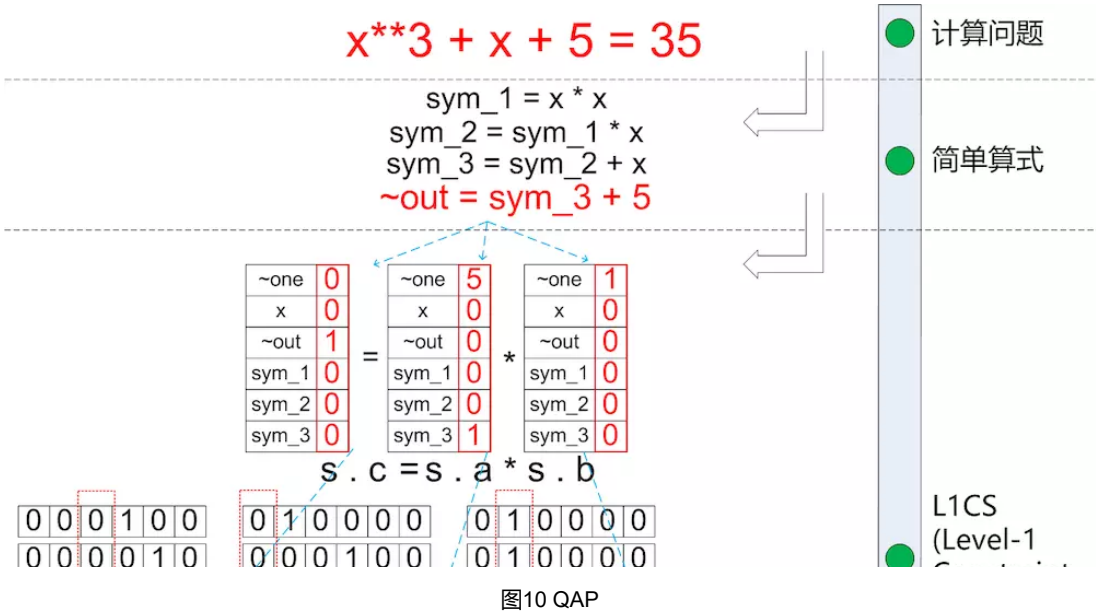
ZCash的例子太复杂，不适合讲解，因此以一个简单的多项式方程  $x^3 + x + 5 = 35$  (这个方程的解是  $x=3$ ，因为  $3^3 + 3 + 5 = 35$ ) 来举例说明将目标计算问题转换为一个QAP的过程。

请重点观察：伴随着问题的转化，解的形式会发生什么变化。

(https://ds  
click.you  
slot=30edc  
dc09-44da  
532ae520

(http://e.cn





(https://dsp  
click.youde  
slot=30edc  
dc09-44da  
532ae520;

(http://e.cn

如图10所示，转换分三步完成：

1. 通过引入中间变量，将计算式 $x^3 + x + 5$ 转换为若干简单算式。这些简单算式要么是“ $x = y$ ”或者“ $x = y (op) z$ ”的形式。操作符“op”代表加 (+)、减 (-)、乘 (\*) 和除 (/)。这些简单算式可视为数字电路中的逻辑门，因此也被称为“代数电路 (Algebraic Circuit)”。图10例中引入的中间变量是 $sym\_1$ 、 $sym\_2$ 和 $sym\_3$ ；

解的新形式：

$x=3, sym\_1=9, sym\_2=27, sym\_3=30, \sim out=35$ （将这些变量的值代入各个简单算式，所有等式成立）

2. 定义向量 $s=[\sim one, x, \sim out, sym\_1, sym\_2, sym\_3]$ （ $\sim one$ 是伪变量，表示常数1），将每个简单算式转换为“ $s \cdot c = s \cdot a * s \cdot b$ ”的形式。其中，“ $\cdot$ ”代表向量内积（将两个向量对应位置的成员相乘，结果再累加）， $a$ 、 $b$ 和 $c$ 是其系数向量。依次完成所有简单算式的转化，将系数向量分别顺序排列，便得到A、B和C三个矩阵（例如，矩阵 $C$ 的最后一行，就是简单算式“ $\sim out = sym\_3 + 5$ ”的系数向量 $c$ ）。这个描述形式有个专门的术语，称作“一阶约束系统 (L1CS)”。满足所有约束条件的向量 $s$ 就是问题的解；

解的新形式：

$s = [\sim one, x, \sim out, sym\_1, sym\_2, sym\_3] = [1, 3, 35, 9, 27, 30]$

3. 最后一步，将每个矩阵压缩为一个多项式组成的向量，例如矩阵 $C \rightarrow C(n)=[C_1(n), C_2(n), C_3(n), C_4(n), C_5(n), C_6(n)]$ 。方法是：对矩阵的每一列分别运用拉格朗日插值法。譬如，矩阵 $C$ 的第3列为 $[0,0,0,1]$ ，现在要求取一个多项式 $C_3(n)$ ，使得 $n$ 分别取1、2、3、4时， $C_3(n)$ 的值为：

n	$C_3(n)$
1	0





n	C <sub>3</sub> (n)
2	0
3	0
4	1

表3 C<sub>3</sub>(n)在不同n的取值

按照拉格朗日插值法，C<sub>3</sub>(n)可以分解为4个部分之和：

- C<sub>3\_1</sub>(n) = A(n-2)(n-3)(n-4)
- C<sub>3\_2</sub>(n) = B(n-1)(n-3)(n-4)
- C<sub>3\_3</sub>(n) = C(n-1)(n-2)(n-4)
- C<sub>3\_4</sub>(n) = D(n-1)(n-2)(n-3)
- C<sub>3</sub>(n) = C<sub>3\_1</sub>(n) + C<sub>3\_2</sub>(n) + C<sub>3\_3</sub>(n) + C<sub>3\_4</sub>(n)

(https://dsp  
click.youde  
slot=30edc  
dc09-44da  
532ae520i

按照表3，已知n = 4时，C<sub>3</sub>(n) = 1。因为，n = 4时，C<sub>3\_1</sub>(n)、C<sub>3\_2</sub>(n)和C<sub>3\_3</sub>(n)分别为0，因此，C<sub>3\_4</sub>(4) = D(4-1)(4-2)(4-3) = 1，从而得到D = 1/6。同理，A = B = C = 0。于是，可知C<sub>3</sub>(n) = 1/6\*(n-1)\*(n-2)\*(n-3) = 0.166n\*\*3-n\*\*2+1.833n-1。

(http://e.cn

求得多项式向量A(n)、B(n)和C(n)后，计算问题便转换为求取解向量s，使得等式s . C(n) - s . A(n) \* s . B(n) = 0 在n=1,2,3,4,5,6时成立，等价于：  
s . C(n) - s . A(n) \* s . B(n) = H(n) \* Z(n)，其中，Z(n) = (n-1)(n-2)(n-3)(n-4)(n-5)(n-6)。  
**各位请注意：方程式中如愿以偿地出现了多项式！**

问题算式发生变化，但解的形式未变，仍为：  
s = [~one, x, ~out, sym\_1, sym\_2, sym\_3] = [1, 3, 35, 9, 27, 30]

为啥一定要处心积虑地搞出多项式来呢？为了以抽样来实现验证过程的“简洁”。这是下一节的内容，在动身之前先解释一下什么叫“NP问题”。

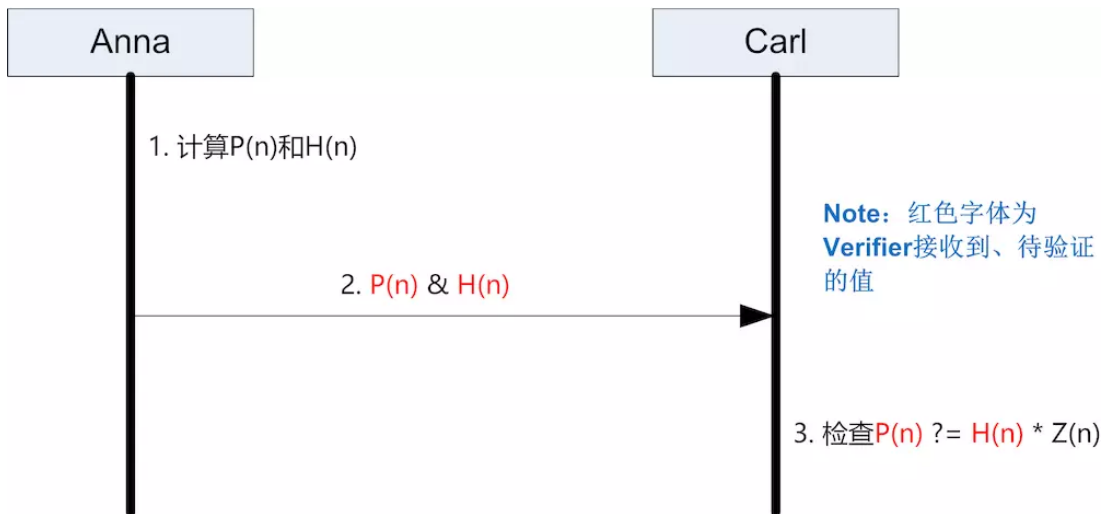
NP是根据计算复杂性对计算问题划分的一种类别：对于NP问题，验证一个解是否正确的步骤可表示输入规模的一个多项式。譬如，验证一个n次方程的解是否正确需要完成的乘法次数为O(n<sub>2</sub>)，因此n次方程就是一个NP问题。**请注意，这里说的是“验证”，而不是求解。**如果一个问题的解可在多项式步骤内求得，这个问题称为P问题。显然，P问题是NP问题的一个子集。“在多项式时间内完成”相当于“可行”，因此，对于NP问题，验证它的解是否正确是“可行的”；而对于P问题，更进一步，求出它的解也是可行的。验证和求解的不对称性是密码学应用的基础。譬如，对于哈希函数，已知Hash(x) = N（常数），当x的取值范围很大时，求解x十分困难；而给定一个x=X，验证Hash(X) ?= N却十分简单。因此，用户密码经常是以哈希值来保存，防止原始密码泄露。NP和P的划分并不是绝对的，如果数学方法上取得突破，NP也可能变成P，即NP=P。应该庆幸目前还没有人做到这一点，否则当今互联网的安全基础将彻底崩塌。

2. 抽样实现简洁验证

上一节对问题进行一系列转化，目的是向“零知识”证明靠近。假设Anna知道原始问题“x\*\*3+x+5 = 35”的解，如果不对问题进行转化，Anna为了自证，好像除了公布解（x=3），似乎也没有别的办法。然而，问题转化为求解“s . C(n) - s . A(n) \* s . B(n) =



$H(n) * Z(n)$ ”后 ( 其中, 系数向量 $A^*(n)$ 、 $B^*(n)$ 和 $C^*(n)$ 是公开的 ), Anna可以她知道解 $s$ , 计算多项式 $P(n)$ 和 $H(n) = P(n)/Z(n)$ , 然后将两个多项式 $P(n)$ 、 $H(n)$ 发给验证者Carl; 后者通过检查 $P(n) \stackrel{?}{=} H(n) * Z(n)$ 是否成立来判断Anna是否真的知道解。从图11可以看出, Anna并没有直接将解 $s$ 告知Carl, 也可以向Carl证明自己知道解, 是不是有点“零知识”的感觉了?



(<https://dsr.click.youda.slot=30edcdc09-44da532ae5207>)

(<http://e.cn>)

图11 zk-SNARK的初次尝试: 计算多项式

当然, 这个方法实际应用是不可行的, 因为至少存在一个效率的问题。例如, 对于ZCash, 多项式的度 ( degree ) 最高可达2,000,000, 这就意味着每次验证都需要传输大量数据 ( 多项式数以百万计的系数值 ), 显然不符合zk-SNARK简洁性的要求。既然不能把整个多项式传送过去进行比较, 那何不在单个点上求值后再比较? 如图12所示:

- 1) Carl任意选择一个点 $n = t$ 发给Anna, 这个点称为抽样点;
- 2) Anna计算 $P(t)$ 和 $H(t)$ ;
- 3) Anna把 $P(t)$ 和 $H(t)$ 发还给Carl;
- 4) Carl检查 $P(t) \stackrel{?}{=} H(t) * Z(t)$ 。如果等式成立, 说明**有很大可能**Anna掌握的 $P(n)$ 和 $H(n)$ 满足“ $P(n) = H(n) * Z(n)$ ”。之所以说“很大可能”, 是因为假设Anna并不掌握正确的 $s$ , 而是任意的 $s'$ , 那么 $P'(n) = s' \cdot C(n) - s' \cdot A(n) * s' \cdot B(n)$ 。曲线 $f(n) = P'(n)$ 与 $g(n) = H(n)Z(n)$ 将相交于有限的点 $\{n_i, i \leq l\}$ , 在这些点,  $P'(n) = H(n)Z(n)$ 也成立。也就是说, 如果Carl选择的 $n = t \in \{n_i, i \leq l\}$ , 那么“ $P'(t) = H(t) * Z(t)$ ”同样成立, Anna便可欺骗Carl相信自己知道正确的解, 而实际并非如此。好在 $l$ 有限, 而 $n$ 的取值范围是无限的, 只要Carl随机选取 $t$ , 撞到的概率很小 ( 与图1和图2所示的例子不同, Carl不需要重复查验多次, 凭一次随机抽样的结果就可达到足够的可信度。好比你买第一张彩票就中了头奖, 那基本可以肯定你买了张假彩票。 )。

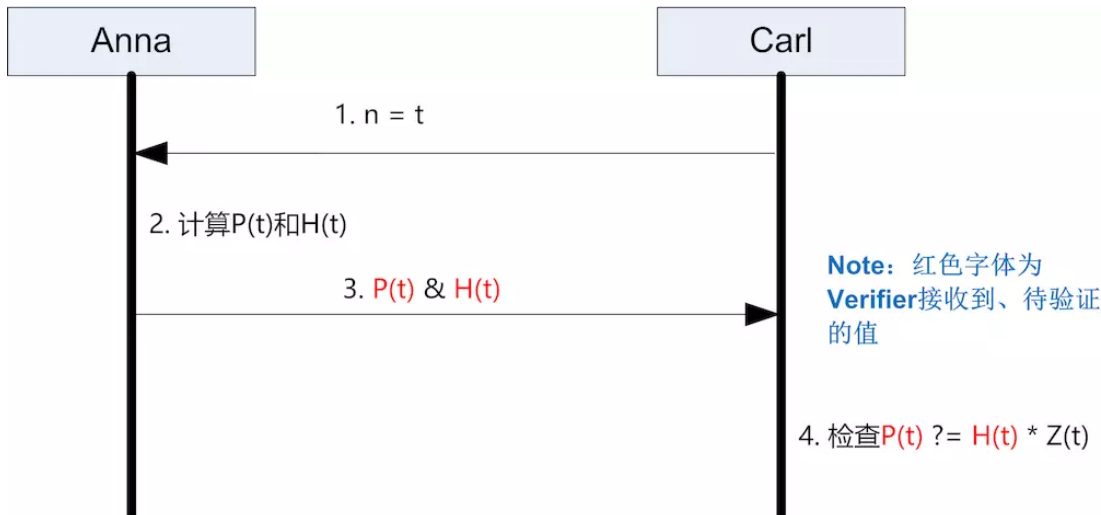


图12 zk-SNARK的再次尝试：抽样验证

可以开香槟庆祝了？还不行，到目前为止，我们的方法还有两个致命漏洞。首先，因为Anna知道抽样点 $n = t$ ，即使她不知道正确的解 $s$ ，她仍可通过精心构造一个 $H'(n)$ ，使得至少在抽样点 $t$ 上 $H'(t) = P'(t)/Z(t)$ ，显然， $H'(t)$ 和 $P'(t)$ 组成的证据会被Carl所接受。

那这样看来，抽样点 $t$ 不能让Prover (Anna) 知道，同时还得让Prover能给出抽样点处的值。这做得到么？答案是可以做到，通过“同态隐藏 (Homomorphic Hiding)”。

### 3. 利用同态映射隐藏抽样点

“同态隐藏”是输入 $x$ 到输出 $X$ 的某种映射 (mapping)  $E$ 的特性：

- 对于绝大多数的 $x$ ，已知 $X=E(x)$ ，无法推导出 $x$ ；
- 如果 $x_1 \neq x_2$ ，则 $E(x_1) \neq E(x_2)$ ；
- $E(ax_1 + bx_2) = a * E(x_1) + b * E(x_2)$ ，即加法同态：经过映射后，加法的计算形式仍然得以保留

(<https://dsr-click.youda.com/slot=30edcdc09-44da532ae5207>)

(<http://e.cn>)

假设我们找到了一个具有同态隐藏特性的映射 $E$ ，便可利用它来对我们的零知识证明方法进行改进。如图13所示，Carl (Verifier) 不再直接将抽样点告知Anna，而是提供了 $t$ 的一系列指数 $t^0, t^1, t^2, t^3 \dots t^N$ 的映射值 $E(1), E(t^1), E(t^2), E(t^3) \dots E(t^N)$  ( $N$ 是一个比任何涉及多项式的阶数都大的整数)。由于不知道 $t$ ，Anna无法直接计算 $P(t)$ 和 $H(t)$ ，只能根据上述映射值来计算 $E(P(t))$ 和 $E(H(t))$ 。多项式 $P(t)$ 和 $H(t)$ 分别是 $\{t^n, n=(1,2,3 \dots, N)\}$ 的线性组合，按照同态映射性质， $E(P(t))$ 和 $E(H(t))$ 也应分别是 $\{E(t^n), n=(1,2,3 \dots, N)\}$ 的线性组合。Carl收到Anna的响应后，通过检查 $E(P(t)) \stackrel{?}{=} E(H(t) * Z(t))$  (Carl知道 $t$ 的值，因此可以算出 $Z(t) = a$ ，进而求解 $E(aH(t))$ ) 便可判定 $P(t) \stackrel{?}{=} H(t) * Z(t)$ ，进而决定是否接受Anna的证据。Anna不知道 $t$ ，也就不能找到合适的 $H(t)$ 正好使“ $E(P(t)) = E(H(t) * Z(t))$ ”成立，这样第一个漏洞便堵上了。

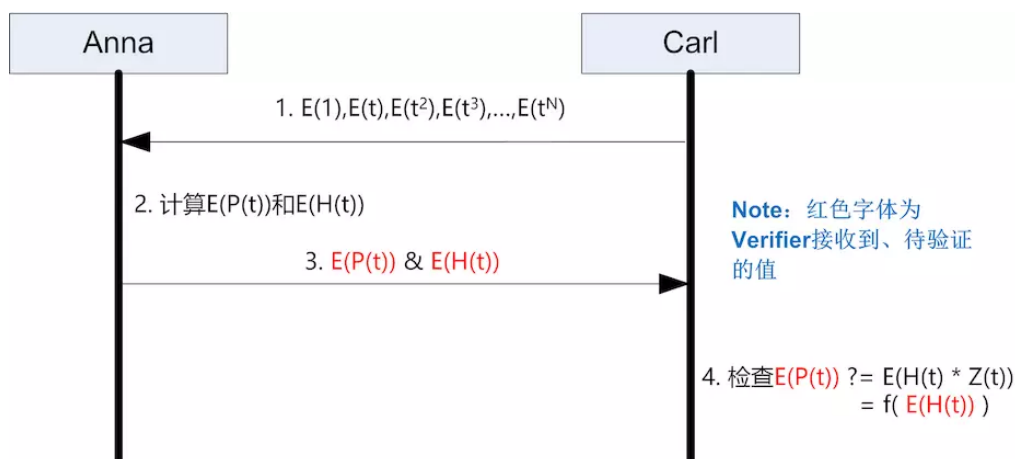


图13 zk-SNARK的又次尝试：同态隐藏抽样点

那另一个漏洞又是啥呢？系数向量 $A(n)$ 、 $B(n)$ 和 $C(n)$ 代表要求解的问题本身。假设Verifier不知道使“ $s \cdot C(n) - s \cdot A(n) * s \cdot B(n) = H(n) * Z(n)$ ”成立的解 $s$ ，但知道另一个问题的解 $s'$ ： $s' \cdot C'(n) - s' \cdot A'(n) * s' = H'(n) * Z(n)$ 。Verifier便可用不同于原始问题的系数向量 $A'(n)$ 、 $B'(n)$ 和 $C'(n)$ 来生成 $P'(n) = s \cdot C'(n) - s \cdot A'(n) * s \cdot B'(n)$ ，然后将 $P'(n)$ 和 $H'(n)$ 作为响应发给Verifier，那么Verifier一定会得出“嗯，ta确实知道解”的结论，只不过“此解非彼



解”也。相当于：老师发给你一份高考数学试题，你偷偷把它换做小学一年级数学试题并且答好交上来；改卷老师并不知道你应该答的是什么题，一检查全对，于是你高考数学便得到满分了！

别灰（得）心（意），兵来将挡，水来土掩，zk-SNARK的下一招过来了。

#### 4. KCA：除了规矩做人，你别无选择

Verifier如何才能知道Prover计算 $P(n)$ 使用的是不是规定的系数向量 $A(n)$ 、 $B(n)$ 和 $C(n)$ 呢？这一过程便称为KCA（Knowledge of Coefficient Test and Assumption）。原理如下。

假设有两个东东 $a$ 、 $b$ ，满足 $b = \alpha * a$ 的约束（ $\alpha$ 为整数，“ $\alpha * a$ ”相当于 $\alpha$ 个 $a$ 相加），那么这一对东东 $(a, b)$ 称为一个“ $\alpha$ 对”。如果我把 $(a, b)$ 告知你，但 $\alpha$ 对你保密，现在要求你提供另一个 $\alpha$ 对，你该怎么办？你可能会说，这还不容易，先通过“ $b/a$ ”求出 $\alpha$ ，然后再任意挑一个 $a'$ ，并算出对应的 $b'$ 就好。如果 $a$ 和 $b$ 是普通数字、加法是普通加法，“ $b$ 除以 $a$ ”是存在的，的确如此。可如果“ $b$ 除以 $a$ ”的运算做不了（这是可能的，后文解释），又该如何？这时，你只有一个选择，那就是分别将 $a$ 和 $b$ 乘以一个整数 $\gamma$ ，则 $(a', b') = (\gamma * a, \gamma * b)$ 也是一个 $\alpha$ 对，即 $b' = \gamma * b = \alpha * \gamma * a = \alpha * a'$ 。

接下来，将此问题扩展一下：如果预先提供给你的不是一个而是 $N$ 个 $\alpha$ 对 $(a_1, b_1)$ ， $(a_2, b_2)$ ，...， $(a_N, b_N)$ ，还是让你返回一个新的 $\alpha$ 对，那怎么整呢？方法是类似的，那就是返回一个由 $a$ 系列和 $b$ 系列值的相同线性组合组成的值对，即 $(c_1 * a_1 + c_2 * a_2 + \dots + c_N * a_N, c_1 * b_1 + c_2 * b_2 + \dots + c_N * b_N)$ ，其中 $c_n$ 是任意整数。反过来，从出题者的角度而言，ta通过检查你提供的 $(a', b')$ 是否一个 $\alpha$ 对，便可基本确信：你返回的两个值 $a'$ 和 $b'$ 是ta所提供的 $a$ 系列和 $b$ 系列值的相同线性组合（为啥说“基本”呢？因为还无法从数学上证明，只能算“good enough”）。

回到我们的zk-SNARK方案存在的漏洞二：如何才能保证Prover回答的是应该回答的问题呢？如上文所述，此漏洞的根源在于Anna可以选择任意 $P'(n)$ 来响应Carl的质询，而 $P'(n)$ 可能与目标问题的系数多项式向量 $A(n)$ 、 $B(n)$ 和 $C(n)$ 没有一毛钱关系。既然如此，我们可以在图13所示方法的基础上进一步改进：Carl可以在质询中只提供 $A(t)$ 、 $B(t)$ 和 $C(t)$ 的值，Anna因而被迫只能基于它们来构建应答，此漏洞由此得以堵上。强迫的手段便是KCA。

具体描述如下。已知目标问题的QAP形式为 $s \cdot C(n) - s \cdot A(n) * s \cdot B(n) = H(n) * Z(n)$ ，其中，多项式系数向量 $A(n)$ 、 $B(n)$ 、 $C(n)$ 和解向量 $s(n)$ 分别为（ $M$ 为QAP形式解向量 $s$ 的阶数）：

- $A(n) = [A_1(n), A_2(n), A_3(n), \dots, A_M(n)]$
- $B(n) = [B_1(n), B_2(n), B_3(n), \dots, B_M(n)]$
- $C(n) = [C_1(n), C_2(n), C_3(n), \dots, C_M(n)]$
- $s(n) = [s_1(n), s_2(n), s_3(n), \dots, s_M(n)]$

令：

- $A(n) = s(n) \cdot A(n) = \sum s_i * A_i(n)$
- $B(n) = s(n) \cdot B(n) = \sum s_i * B_i(n)$
- $C(n) = s(n) \cdot C(n) = \sum s_i * C_i(n)$

(https://dsr  
click.youda  
slot=30edc  
dc09-44da  
532ae520i

(http://e.cn



那么, QAP方程式可描述为“ $C(n) - A(n) * B(n) = H(n) * Z(n)$ ”。按照图13所示方法, Carl要求Anna提供 $A(n)$ 、 $B(n)$ 、 $C(n)$ 和 $H(n)$ 在 $n=t$ 的采样值的同态隐藏 $E(A(t))$ 、 $E(B(t))$ 、 $E(C(t))$ 和 $E(H(t))$ , Carl再设法检查“ $E(C(t)) - A(t) * B(t) \stackrel{?}{=} E(H(t)) * Z(t)$ ”, 从而验证Anna知道的解是否正确。与图13方法不同的是, Carl在第1步提供给Anna的不是基本粒子 $E(t^n)$ , 而是三组、每组 $M$ 个 $\alpha$ 对 ( $\alpha$ 是Carl产生的随机值) :

- 第一组数据

$E(A_1(t)), E(\alpha_A A_1(t))$  (注: 根据同态映射的性质,  $E(\alpha_A A_1(t)) = \alpha_A E(A_1(t))$ , 下同。)

$E(A_2(t)), E(\alpha_A A_2(t))$

...

$E(A_M(t)), E(\alpha_A A_M(t))$

- 第二组数据

$E(B_1(t)), E(\alpha_B B_1(t))$

$E(B_2(t)), E(\alpha_B B_2(t))$

...

$E(B_M(t)), E(\alpha_B B_M(t))$

- 第三组数据

$E(C_1(t)), E(\alpha_C C_1(t))$

$E(C_2(t)), E(\alpha_C C_2(t))$

...

$E(C_M(t)), E(\alpha_C C_M(t))$

(<https://dsr-click.youdao.com/slot=30edcdc09-44da532ae5207>)

(<http://e.cn>)

同时, Carl要求Anna在响应中返回三个 $\alpha$ 对 $\langle E(A(t)), E(\alpha_A A(t)) \rangle$ 、 $\langle E(B(t)), E(\alpha_B B(t)) \rangle$ 和 $\langle E(C(t)), E(\alpha_C C(t)) \rangle$ 。Anna不知道这几个 $\alpha$ 的值, 因此根据KCA推断: 为了生成第一个 $\alpha$ 对, 她只能以Carl提供的第一组 $\alpha$ 对的某种线性组合来合成 $E(A(t))$ 和 $E(\alpha_A A(t))$ :

- $E(A(t)) = E(a_1 * A_1(t) + a_2 * A_2(t) + \dots + a_M * A_M(t)) = a_1 * E(A_1(t)) + a_2 * E(A_2(t)) + \dots + a_M * E(A_M(t))$

- $E(\alpha_A A(t)) = E(a_1 * \alpha_A A_1(t) + a_2 * \alpha_A A_2(t) + \dots + a_M * \alpha_A A_M(t)) = a_1 * E(\alpha_A A_1(t)) + a_2 * E(\alpha_A A_2(t)) + \dots + a_M * E(\alpha_A A_M(t))$

同理, Anna可以构建:

- $E(B(t)) = b_1 * E(B_1(t)) + b_2 * E(B_2(t)) + \dots + b_M * E(B_M(t))$

- $E(\alpha_B B(t)) = b_1 * E(\alpha_B B_1(t)) + b_2 * E(\alpha_B B_2(t)) + \dots + b_M * E(\alpha_B B_M(t))$

- $E(C(t)) = c_1 * E(C_1(t)) + c_2 * E(C_2(t)) + \dots + c_M * E(C_M(t))$

- $E(\alpha_C C(t)) = c_1 * E(\alpha_C C_1(t)) + c_2 * E(\alpha_C C_2(t)) + \dots + c_M * E(\alpha_C C_M(t))$

其中, 系数 $a_i$ 、 $b_i$ 和 $c_i$ 如果仅为满足 $\alpha$ 对的约束, 可以是任何整数, 三个序列也不用相同。但如果要进一步强迫Anna使用相同的系数序列, 怎么办呢? 答案是引入多项式序列 $\{L_i(n)\}$ :

- $L_i(n) = A_i(n) + B_i(n) + C_i(n)$

令 $L(n) = \sum L_i(n)$ , 那么当“ $a_i=b_i=c_i=L_i(t)$ ”时, 有:

$$L(n) = \sum L_i(n) = A(n) + B(n) + C(n)$$

等价于:

$$\text{随机选择一个 } \beta, \text{ 对任意 } n=t, E(\beta L(t)) = E(\beta * (A(t) + B(t) + C(t))) = \beta * (E(A(t)) + E(B(t)) + E(C(t)))$$





另一计算 $E(\beta L(t))$ 的方法是： $E(\beta L(t)) = \sum_i E(\beta L_i(t))$ 。如果 $\beta$ 对Anna而言是未知的，那么有理由相信：只有当Anna选择相同的系数 $a_i$ 、 $b_i$ 、 $c_i$ 和 $l_i$ ，才能保证两种方法计算的 $E(\beta L(t))$ 对于任何采样点都是相等的。

因此，Carl发给Anna的质询中还应包括第四组数据：

- 第四组数据

$E(\beta L_1(t))$

$E(\beta L_2(t))$

...

$E(\beta L_M(t))$

(https://dsr  
click.youda  
slot=30edc  
dc09-44da  
532ae5207

而Anna的响应中相应增加 $E(\beta L(t))$ ：

- $E(\beta L(t)) = E(l_1 \beta L_1(t) + l_2 \beta L_2(t) + \dots + l_M \beta L_M(t))$

Carl通过检查Anna响应数据的一致性，即“ $E(\beta L(t)) \stackrel{?}{=} \beta * (E(A(t)) + E(B(t)) + E(C(t)))$ ”来检验系数 $a_i$ 、 $b_i$ 和 $c_i$ 是否相同。

(http://e.cn

最后，为了让Anna能计算 $E(H(t))$ ，质询中还应增加第五组数据：

- 第五组数据

$E(1)$ ,  $E(t)$ ,  $E(t^2)$ , ...,  $E(t^N)$

综上所述，引入KCA机制后，我们的zk-SNARK方案如图14所示。

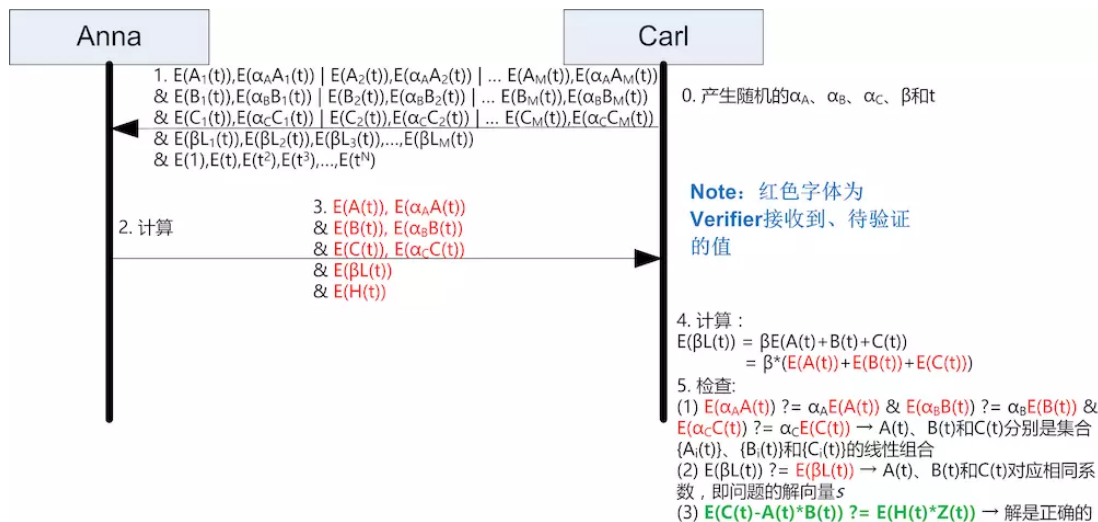


图14 zk-SNARK的另次尝试：KCA

读到这里，眼光犀利的朋友可能会质疑：

(1) 不是要“简洁”么，Verifier发给Proofer的质询中包含的序列 $\{E(t^i)\}$ 包含了N个元素（对ZCash而言，高达数百万），每次验证都要传输，怎能算得上“简洁”？

(2)  $E(C(t) - A(t) * B(t))$  包含了乘法，如何用 $E(A(t))$ 、 $E(B(t))$ 和 $E(C(t))$ 来求值呢？

问题(1)的解决办法很简单：把Verifier发给Anna的一大坨数据（见图14）变成所谓的“共同参考数据集”（CRS, Common Reference String），通过某种可信的方式产生，作为一种全体节点的共识，在所有交易的验证过程中使用，因而“质询-响应”的交互式验证方式变成了只需要Proofer提交证据即可（如图15所示）：



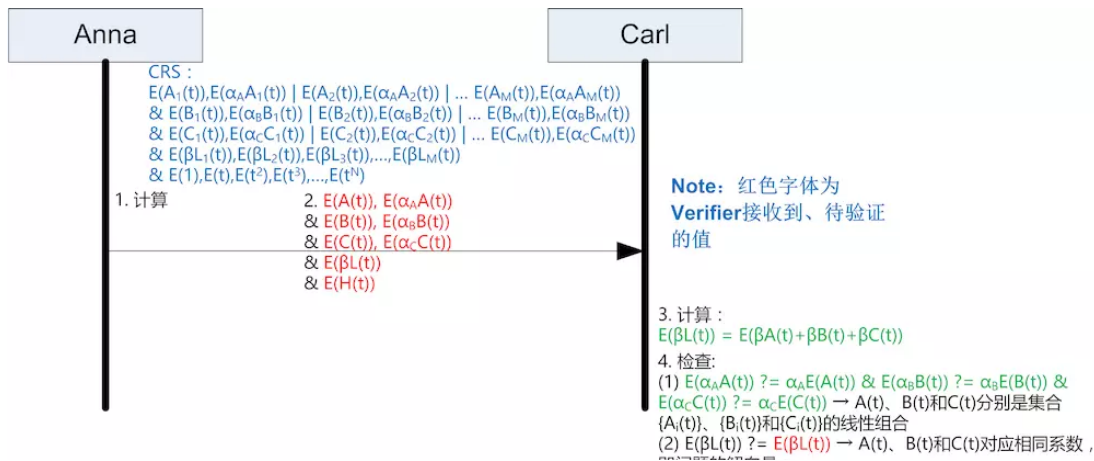


图15 零交互验证

可是这样一来，几个 $\alpha$ 和 $\beta$ 对于Verifier也是未知的了。即使在CRS中增加它们的同态隐藏值，可图15中的第3步和第4-(1)步计算式中出现了乘法，目前也变得无法计算了。为了解决乘法的同态隐藏问题，急需新英雄加入，这便是“双线性映射 (bilinear map)”。

(<https://dsr.click.youda.slot=30edcdc09-44da532ae5207>)

(<http://e.cn>)

## 5. 双线性映射(bilinear map)：乘法的同态隐藏

前文介绍的同态隐藏是一对一的，即将一个输入映射到一个输出。而双线性映射是将分别来自两个域的两个元素映射到第三个域中的一个元素： $e(X, Y) \rightarrow Z$ ，同时在两个输入上都具备线性：

- $e(P+R, Q) = e(P, Q) + e(R, Q)$
- $e(P, Q+S) = e(P, Q) + e(P, S)$

假设对于 $x$ 的任意两种因数分解 $(a, b)$ 和 $(c, d)$  (即 $x=ab=cd$ )，存在两个加法同态映射 $E_1$ 和 $E_2$ ，以及一个双线性映射 $e$ ，使得以下等式总是成立：

- $e(E_1(a), E_2(b)) = e(E_1(c), E_2(d)) = X$

那么， $x \rightarrow X$ 的映射也是加法同态映射，记作 $E$ 。 $E$ 的线性属性证明如下：

$$\begin{aligned} & E(ax_1 + bx_2) \\ &= e(E_1(ax_1 + bx_2), E_2(1)) \\ &= e(a \cdot E_1(x_1) + b \cdot E_1(x_2), E_2(1)) \\ &= a \cdot e(E_1(x_1), E_2(1)) + b \cdot e(E_1(x_2), E_2(1)) \\ &= a \cdot E(x_1) + b \cdot E(x_2) \end{aligned}$$

如果我们找到这种映射 $E$ ，乘法的同态隐藏问题便能迎刃而解： $E(xy) = e(E_1(x), E_2(y))$ 。于是，我们的zk-SNARK方案有了最终版本，如图16所示。



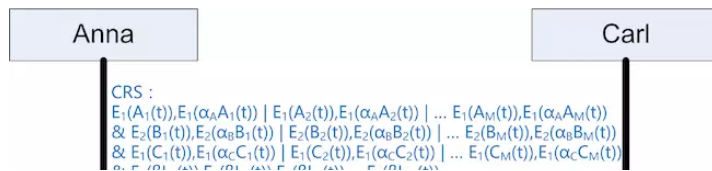


图16 最终的zk-SNARK方案

请注意与图15所示方案的主要区别：

- CRS中包含了两种同态映射值： $E_1$ 和 $E_2$ 。同时，新增了几个 $\alpha$ 、 $\beta$ 的映射值，它们会在后续的验证过程中用到；
- 为了验证Anna返回的 $\pi'_A \stackrel{?}{=} \alpha_A \pi_A$ ，将其转化为验证 $e(\pi_A, E_2(\alpha_A)) \stackrel{?}{=} e(\pi'_A, E_2(1))$ ，因为按照两者是等价的。同理，图15中4-(2)、4-(3)步的验证等式也通过双线性映射 $e$ 进行了转化，验证工作变得可行。

(<https://dsplay.click.youdao.com/slot=30edcdc09-44da532ae5207>)

恭喜坚持读到这里的同学，zk-SNARK的基本原理应该已经清楚了。接下来，你可能会问：上文推导zk-SNARK做出了一系列假设，例如，无法通过 $\alpha a$ 和 $a$ 的值推导出 $\alpha$ ，又例如上述双线性映射存在，那么究竟这些看似不靠谱的假设是否成立呢？感觉不靠谱是因为我们总是站在最熟悉的普通数世界思考问题，譬如，在这个世界，有乘就有除，因此知道乘积和一个乘数，做一次除法不就知道另一个乘数了，哪来的单向性呢？为了得到想要的，我们必须跳出固有思维，进入一个新世界，这便是椭圆曲线的世界！

(<http://e.cn>)

## 7. 椭圆曲线：一个新世界

数学有一个称为“抽象代数”的分支，主要研究对象是代数结构，例如群、环、域。所谓“代数结构”就是一个集合以及定义在此集合上的运算。例如，“群 (group)”就是由一个集合以及一个二元运算符“ $\cdot$ ”组成，它具备以下性质：

- 封闭性：对于所有 $G$ 中 $a, b$ ，运算 $a \cdot b$ 的结果也在 $G$ 中。
- 结合律：对于所有 $G$ 中的 $a, b$ 和 $c$ ，等式 $(a \cdot b) \cdot c = a \cdot (b \cdot c)$ 成立。
- 单位元：存在 $G$ 中的一个元素 $e$ ，使得对于所有 $G$ 中的元素 $a$ ，总有等式 $e \cdot a = a \cdot e = a$ 成立。
- 逆元：对于每个 $G$ 中的 $a$ ，存在 $G$ 中的一个元素 $b$ 使得总有 $a \cdot b = b \cdot a = e$ ，此处 $e$ 为单位元。

如果进一步满足交换律，那么这个群称为“阿贝群”：

- 交换律：对于所有 $G$ 中的 $a$ 和 $b$ 和 $c$ ，等式 $a \cdot b = a \cdot b$ 成立。

请注意，集合和操作是构成群的两个缺一不可的组成部分。以我们熟知的整数集合 $\{\dots, -3, -2, -1, 0, 1, 2, 3, \dots\}$ 为例，整数集分别与加法（+）或乘法（\*）结合可以构成两个不同的群：整数加法群和整数乘法群。这里请大家注意了：**运算符“ $\cdot$ ”的含义对于不同的群是不同的，有时候为了便于理解，操作符会写作“+”或“\*”来分别类比普通数的加法或乘法。对于只有一个操作符的群而言，操作符写成什么样子都OK，例如双线性映**



射“ $e(P+R, Q)=e(P,Q)+e(R,Q)$ ”，也可表示为“ $e(P+R, Q)=e(P,Q)*e(R,Q)$ ”，因为等式中“+”和“\*”分别是两个不同群的操作符。虽然用+更符合表达“线性”的习惯，奈何小爷我愿意呢！

令p为一个素数，在集合 $\{0,1,2,3,...,p-1\}$ 上，也可以定义加法和乘法操作，不过与普通加法、乘法不同的是：结果需要对p取模，分别称为模加和模乘。例如，假设 $p=7$ ，则：

- 模加：  $4 + 5 = 2 \pmod{7}$
- 模乘：  $3 * 6 = 4 \pmod{7}$

同整数集一样，集合 $\{0,1,2,3,...,p-1\}$ 分别与模加或模乘结合构成了两个群。同时，它与这两个操作符一起还构成了一个域，记做 $F_p$ 。所谓“域”，就是一个集合及定义在其上的两个操作：加法和乘法，这两个操作满足以下全部条件：

- 结合加法，构成一个加法阿贝群。加法群的单位元记为“0”；
- 非“0”元素组成的子集，结合乘法，构成一个乘法阿贝群；
- 乘法对加法符合交换律：对于任何 $a、b、c \in F_p$ ， $a * (b + c) = a * b + a * c$ 。

(https://ds  
click.youda  
slot=30edc  
dc09-44da  
532ae520i

(http://e.cn

那位读者说，你叨叨了这半天，只字未提“椭圆曲线”，是不是串线了啊？呵呵，别急，这是必要的铺（前）垫（戏）。通过上述例子，想传递一个概念：如果普通数与普通加、乘构建是我们熟悉的世界，那么通过重新定义规则，便可构建起无数的平行世界。在旧世界看似不可能的事情，放到新世界却有可能实现。基于椭圆曲线便可构建一个这样的新世界。

先解释什么是椭圆曲线。假设p为大约3的素数，取 $u,v \in F_p$ （敲黑板，就是上面由两个模运算和集合 $\{0,1,2,3,...,p-1\}$ 构建的域）且满足： $4u^3+27v^2 \neq 0$ ，可定义等式 $Y^2 = X^3 + uX + v$ ，那么，由座标(x,y)满足上述等式的点组成的集合就称为**椭圆曲线**。

在椭圆曲线这个由点组成的集合上也可以定义“加法”。加法遵守的基本规则是：

- 规则1：单位元记为**O**；
- 规则2：任意一条直线与椭圆曲线相交的所有点（x的指数为3，最多3个）相加，结果为**O**；
- 规则3：为计算一个点Q的两倍 $Q+Q$ （简写为 $2Q$ ），可画一条经过Q、并与椭圆曲线相切的直线，令这条直线与椭圆曲线的另一个交点为S，那么 $S = Q + Q$ 。

基于规则2，可以有以下两个推论：

- 推论1：经过任意点P画一条垂直x轴的线，与椭圆曲线相较于Q，则 $P+Q=O$ ，因此， $Q=-P$ ，Q为P的逆元。见图17；
- 推论2：若P、Q两点的x座标不相同（P、Q不互为逆元），连接P和Q画一条直线，与椭圆曲线相较于R，则 $P+Q+R=O$ ，即 $P+Q=-R$ 。见图18。



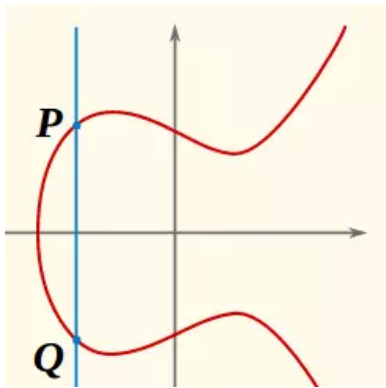


图17 点运算：逆元

(https://ds  
click.youda  
slot=30edc  
dc09-44da  
532ae520i

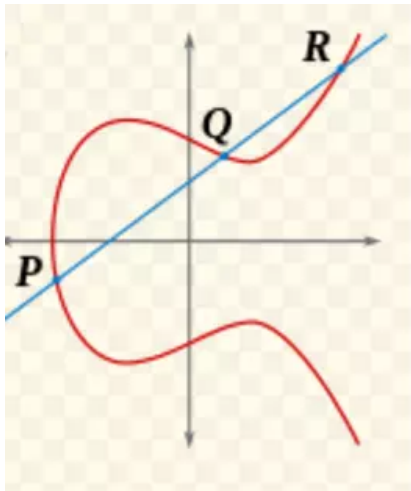


图18 点运算：加法

(http://e.cn

基于上述点运算的规则和推论不难看出：椭圆曲线的点与我们定义出来的加法也构建出了另一个阿贝群，记作 $C(F_p)$ ，一个新世界。这个新世界又给我们带来什么可能呢？读者是否还记得，我们在引入KCA时提到：已知 $\alpha * a$ 和 $a$ ，无法求出 $\alpha$ ？如果 $a$ 和 $b$ 是普通数，难以想象；但如果 $a$ 和 $b$ 是椭圆曲线上的点，这便是千真万确的，因为在椭圆曲线运算中，两个点的除法不存在（或者更精确的说，还没有找到有效的算法）。

由此，我们可以定义映射 $E: x \rightarrow P$ ，其中 $x$ 属于 $F_p$ ，而 $P$ 是椭圆曲线上的点，并且满足： $P = x * G$ 。等式中的 $G$ 也是椭圆曲线的一个点，不过它比较特殊的是：椭圆曲线上的所有点都可以由它经过若干次自加（ $G+G+G+\dots$ ）得到，因此它被称为创世点（generator）。映射 $E$ 就是一种前面几节透支的加法同态映射：

- 对于 $x, y \in F_p$ ， $E(ax+by) = (ax+by) * G = ax * G + by * G = a * E(x) + b * E(y)$

当点的座标 $x,y \in F_p$ 时，这些点组成的群是 $C(F_p)$ ，它的创世点记作 $G_1$ 。而当座标 $x,y$ 属于某个复数集合时，同样也能使椭圆曲线方程成立，这些点组成的集合是 $C(F_{p^k})$ ，它的创世点记作 $G_2$ 。基于 $G_1$ 和 $G_2$ 便可分别定义同态映射 $E_1(x) = x * G_1$ 和 $E_2(x) = x * G_2$ ，而经过证明存在一种被称作“Tate reduced pairing”的映射具有双线性特性，即对于任意 $P, R \in C(F_p)$ ， $Q, S \in C(F_{p^k})$ ，存在：

- $e(P+R, Q) = e(P, Q) + e(R, Q)$
- $e(P, Q+S) = e(P, Q) + e(P, S)$





鉴于篇（理）幅（解）有限，Tate reduced pairing的证明过程便不再展开。由此，实现zk-SNARK的需要的原料找齐了。

## 未完待续

zk-SNARK已在ZCash中应用，因而不是纸上谈兵。然而其实用性还有待检验，毕竟产生零知识证明的代价是相当高的（在ZCash，高达30~40秒），是否真能带来足够的安全尚不十分清楚。我接下来会对ZCash的实现进行进一步研究，将在下一篇博文中向同样好学的你介绍其技术实现细节，希望到时候咱们可以动手实践yi'fan。

## 参考文献

- [1] ZCash protocol, <https://github.com/zcash/zips/blob/master/protocol/protocol.pdf> (https://github.com/zcash/zips/blob/master/protocol/protocol.pdf) 注：ZCash的详细技术文档，符号漫天作雪飞，作为参考吧！
- [2] “How Transactions Between Shielded Addresses Work”, <https://blog.z.cash/zcash-private-transactions/> (https://blog.z.cash/zcash-private-transactions/)。注：zcash基本原理的简单介绍，读起来没有压力，推荐！
- [3] “Introduction to zk-SNARKs with examples”, <https://media.consensys.net/introduction-to-zksnarks-with-examples-3283b554fc3b> (https://media.consensys.net/introduction-to-zksnarks-with-examples-3283b554fc3b)。注：通过智能合约实现zk-SNARK验证的例子，找到一些落地的感觉。同时，清楚解释了PK（Proof Key）和VK（Verification Key）
- [4] “Explaining SNARKs series”, <https://z.cash/technology/zksnarks.html> (https://z.cash/technology/zksnarks.html) 注：ZCash官网，详细解释zk-SNARK的原理。循序渐进，逐步展开，是本文的主要参考
- [5] “zk-SNARKs: Under the Hood”, <https://medium.com/@VitalikButerin/zk-snarks-under-the-hood-b33151a013f6> (https://medium.com/@VitalikButerin/zk-snarks-under-the-hood-b33151a013f6) 注：V神对zk-SNARKS的解读，与[4]相互印证着看，更容易理解。特别是对QAP和椭圆曲线原理的解释是[4]所没有的
- [6] “Zero Knowledge Proofs: An illustrated primer”, <https://blog.cryptographyengineering.com/2014/11/27/zero-knowledge-proofs-illustrated-primer/> (https://blog.cryptographyengineering.com/2014/11/27/zero-knowledge-proofs-illustrated-primer/) 注：对zk-SNARK原理的零数学解读，适合建立起基本概念。特别是对于证明“零知识”可能性的思想游戏证明很有启发
- [7] “How toxic is the waste in a zkSNARK trusted setup?”, <https://medium.com/qed-it/how-toxic-is-the-waste-in-a-zksnark-trusted-setup-9b250d59bdb4> (https://medium.com/qed-it/how-toxic-is-the-waste-in-a-zksnark-trusted-setup-9b250d59bdb4) 注：解释了如何利用“核废料”造假
- [8] “群、环、域”，<https://blog.csdn.net/u013281331/article/details/28233961> (https://blog.csdn.net/u013281331/article/details/28233961) 注：一些必要数学概念的解释
- [9] “Succinct Non-Interactive Zero Knowledge for a von Neumann Architecture”, <https://eprint.iacr.org/2013/879.pdf> (https://eprint.iacr.org/2013/879.pdf) 注：25页有一张图很好地概述了zk-SNARK协议

(https://dsp  
click.youda  
slot=30edc  
dc09-44da  
532ae520i

(http://e.cn

小礼物走一走，来简书关注我

赞赏支持





逐舞传歌 (/u/76eda941ad11)

写了 42694 字, 被 32 人关注, 获得了 23 个喜欢 (/u/76eda941ad11)

+ 关注

喜欢 | 11



更多分享

(https://dsp  
click.youda  
slot=30edc  
dc09-44da  
532ae520i

开发10年  
全记在这本Java进阶宝典了

Spring源码分析

分布式架构

微服务架构

JVM性能优化

高效DevOps

多线程并发编程

点击领取



(http://e.cn


(/p/428251ede1aa)



登录后发表评论 (/sign后发表评论source=desktop&utm\_medium=not-signed-in-comr

13条评论 只看作者

按时间倒序 按时间正序



回头是墙 (/u/13241d25ae6e)  
8楼 · 2018.11.27 17:40  
(/u/13241d25ae6e)  
如果用于现实生活中, 是不是首先得将问题转换为多项式, 然后进行后面的A (x) ,  
B (x) , C (x) , Z (x) .....等计算?


赞 回复


逐舞传歌 (/u/76eda941ad11): @回头是墙 (/u/13241d25ae6e) 文中描述的计算过程只是原理, 证明“可行”, 但肯定不是“最优”。类比一下, 好比求取123的456次方, 当然是“可行的”:123自乘456次就好, 但实际计算时通过求对数, 再求指数会更容易些。至于在实际问题中的应用, 等我研究好了再分享给大家。  
2018.11.27 18:58 回复

回头是墙 (/u/13241d25ae6e): @逐舞传歌 (/u/76eda941ad11) 好的, 谢谢  
2018.11.28 09:36 回复

添加新评论

^





猜6 (/u/ee3bd1b1fffd)  
7楼 · 2018.11.20 13:57  
(/u/ee3bd1b1fffd)

https://www.jianshu.com/p/7b772e5cdaef?utm\_source=oschina-app

20/24

我知道国内一个叫SERO超零协议的项目，在零知识证明技术上有非常重大的突破，所应用Super-ZK技术，比Zcash所用的zk-Snarks加密速度快20倍，并且支持智能合约，DApp开发者可以在SERO CHAIN上发行自己的具有隐私保护的通证和票据，进行流通，并且支持选择性透明。这SERO项目的视频介绍，如果零知识证明感兴趣，非常值得研究一下：<https://mp.weixin.qq.com/s/60s4pg90lVePR-H68h1ZdA>  
(<https://mp.weixin.qq.com/s/60s4pg90lVePR-H68h1ZdA>)

赞 回复

逐舞传歌 (/u/76eda941ad11): @猜6 (/u/ee3bd1b1fffd) 谢谢推荐  
2018.11.20 19:44 回复

猜6 (/u/ee3bd1b1fffd): @逐舞传歌 (/u/76eda941ad11) 不客气，我最近一直在研究sero，写了点东西，今晚会发布出来，希望能分享给你。  
2018.11.21 19:11 回复

逐舞传歌 (/u/76eda941ad11): @猜6 (/u/ee3bd1b1fffd) 期待  
2018.11.27 18:50 回复

添加新评论

(<https://dsr-click.youda.com/slot=30edcdc09-44da532ae5207>)

(<http://e.cn>)



Hansirrrrr (/u/c1f08697be40)  
5楼 · 2018.10.27 11:40

(/u/c1f08697be40)  
有个问题，一开始的时候，验证 $P(n)=H(n)*Z(n)$ ，既然 $Z(n)$ 已知，那我任意选取一个多项式 $H(n)$ ，再通过 $Z(n)$ 、 $H(n)$ 计算 $P(n)$ ，最后将 $P(n)$ 、 $H(n)$ 发送给验证者不是一定会验证通过吗？

赞 回复

逐舞传歌 (/u/76eda941ad11): @Hansirrrrr (/u/c1f08697be40) 请继续阅读第4节KCA，对你的问题作了详细解释。  
2018.10.31 08:10 回复

添加新评论



凌若何 (/u/f43f30213923)  
4楼 · 2018.10.08 17:27

(/u/f43f30213923)  
大写的fu

赞 回复



吴佩在天涯 (/u/958bffce17ba)  
3楼 · 2018.09.12 15:25

(/u/958bffce17ba)  
厉害了

赞 回复



中本小猫 (/u/32a14525948f)  
2楼 · 2018.09.06 12:45

(/u/32a14525948f)  
厉害









赞 回复

逐舞传歌 (/u/76eda941ad11): 😊  
2018.09.06 13:58 回复

添加新评论

被以下专题收入，发现更多相似内容


-  区块链研习社 (/c/b17f09dc2831?utm\_source=desktop&utm\_medium=notes-included-collection)
-  区块链大学 (/c/b410c42c3933?utm\_source=desktop&utm\_medium=notes-included-collection)
-  互联网科技 (/c/93d58e9169cb?utm\_source=desktop&utm\_medium=notes-included-collection)
-  区块链研究 (/c/9d18f721ec4c?utm\_source=desktop&utm\_medium=notes-included-collection)
-  区块链技术 (/c/1ce1650dde5f?utm\_source=desktop&utm\_medium=notes-included-collection)
-  区块链 (/c/e38b6e8ebc1b?utm\_source=desktop&utm\_medium=notes-included-collection)

(https://dsp  
click.youde  
slot=30edc  
dc09-44da  
532ae520i

(http://e.cn


2018-06-06 (/p/cb1ef2554746?utm\_campaign=maleskine&utm\_content...

https://etherscan.io/tokens?q=a&p=1 https://etherscan.io/tokens?q=a&p=2 https://etherscan.io/tokens?  
q=a&p=3 https://etherscan.io/tokens?q...

 阿东\_75c8 (/u/ef49f9e177c9?  
utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendatio


高考3500 (/p/0bda5d804ee3?utm\_campaign=maleskine&utm\_content=...

A a (an) [ə, eɪ(ə)n] art. 一 (个、件.....) abandon [əˈbændən] v.抛弃, 舍弃, 放弃 ability [əˈbɪlɪti] n. 能  
力; 才能 able [ˈeɪb(ə)] a. 能够; 有能力的 abnormal [æbˈnɔːm...

 涂桃子 (/u/02eb49244585?  
utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendatio

snapshot-1.0.0.js (/p/b7dc909dcaae?utm\_campaign=maleskine&utm\_...


"use strict";function \_classCallCheck(e,t){if(!(e instanceof t))throw new TypeError("Cannot call a class as a  
function");}var \_createClass...

 久些 (/u/8ef889f37263?  
utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendatio




ROGOT\_GRE (/p/01f650445f3c?utm\_campaign=maleskine&utm\_conte...

[TOC] Class I. Words Expressing Abstract Relations Section I. Existence 1. Being, in The Abstract  
existence 1 absolute a.绝对的，完全的; 无(条件)...

 蕾娜漢默 (/u/45ffb137fb0f?  
utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendatio

爱上你我喜欢上了寂寞 (/p/1a9231f7ea4f?utm\_campaign=maleskine&ut...

《爱上你我喜欢上了寂寞》文/白传英 总想和你在一起 就像我们过去相恋的日子 你却变得有气无力 诺言撒了一地 我不知如何把你想起 拾起标有你的日期 我在日历上寻找 发现对你的思念已过期 我不知如何走进你 看...

 白清风 (/u/0c80b78062e0?  
utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendatio  
(https://dsp  
click.youde  
slot=30edc  
dc09-44da  
532ae520i

(/p/825e79e7545c?




(http://e.cn

utm\_campaign=maleskine&utm\_content=note&utm\_medium=seo\_notes&utm\_source=recommendatio

孤独的收获 (/p/825e79e7545c?utm\_campaign=maleskine&utm\_content...

单身，空手，带着心一起走。带本书，单身一人出门游，心非常的自由。因为不认识任何人，在车上，不会有人和我说话。这样因为无人打扰，我的心绪，我的思想就自由的飞翔，不会有人来打断自己的思路。想...

 谭皓匀 (/u/40ca87ff425a?  
utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendatio


(/p/f78d8eb47797?



utm\_campaign=maleskine&utm\_content=note&utm\_medium=seo\_notes&utm\_source=recommendatio


还我一片蓝色的淡然 (/p/f78d8eb47797?utm\_campaign=maleskine&utm...

我踏浪而来乌云狰狞而笑将我的白裙染黑暴雨淫威而来浴我成黑色的精灵 我的眼睛迸出黎明之火拔剑向天闪亮了整个海天 那颗深处的灵魂屹立不倒我用尽所有的力量拉成一道弓用灵魂做成一支带火的箭射向那风雨...

 蘅芜潇潇 (/u/f7f45e6f8cfd?  
utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendatio

反复 (/p/f2df877b462f?utm\_campaign=maleskine&utm\_content=note&...

反反复复 无常。不嫌累。

 简筱微 (/u/ae9645035042?  
utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendatio

(/p/c009e19b6ce4?




utm\_campaign=maleskine&utm\_content=note&utm\_medium=seo\_notes&utm\_source=recommendatio



玻璃心？你以为我愿意 (/p/c009e19b6ce4?utm\_campaign=maleskine&ut...

1 前两天晚上躺在床上看到朋友圈有这么一条动态，内容是这样的：“你一定要做一个有两把刷子的人，不要光说不练。”看到这个动态我的心理活动是：我靠，在说我么？谁让今天恰巧跟这位朋友一起在学校食堂吃...

 路晖 (/u/764bb8897c82?

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendatio

(https://dsp  
click.youda  
slot=30edc  
dc09-44da  
532ae520i

(http://e.cn

