

区块链中的密码学

路远

中国科学院软件研究所

`luyuan@iscas.ac.cn`

密码学基础回顾

从安全通信说起

一个古老(却越来越不正确)的定义:
密码学 = 研究安全通信的学科?

希腊语词源: kryptos = > 隐藏
-graphy = > 书写

现代密码学(80年代后):

严格的定义;
精确的假设;
安全性的证明。

除安全通信以外, 也可应用于更广泛的安全功能(安全多方计算、共识机制、零知识证明、数据安全擦除、持久存储等等), 逐渐成为一种可证明安全的研究方法论。

Cryptography

[Article](#) [Talk](#)

From Wikipedia, the free encyclopedia

"Secret code" redirects here. For the Aya Kamiki album, see [Secret Code](#).

"Cryptology" redirects here. For the David S. Ware album, see [Cryptology \(album\)](#).

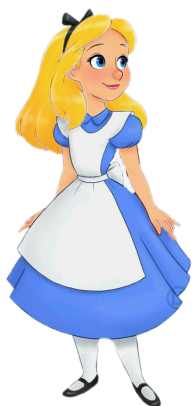


This article **needs additional citations for verification**. Please [add citations to reliable sources](#). Unsourced material may be challenged.
Find sources: "Cryptography" – news · newspapers · books · scholar · JSTOR
[remove this template message](#)

Cryptography, or **cryptology** (from Ancient Greek: κρυπτός, romanized: *kryptós* "hidden, secret"; and γράφειν *graphein*, "to write", or -λογία *-logia*, "study", respectively^[1]), is the practice and study of techniques for secure communication in the presence of [adversarial](#) behavior.^[2] More generally, cryptography is about constructing and analyzing [protocols](#) that prevent third parties or the public from reading private messages.^[3] Modern cryptography exists at the intersection of the disciplines of [mathematics](#), [computer science](#), [information security](#), [electrical engineering](#), [digital signal processing](#), [physics](#), and others.^[4] Core concepts related to [information security](#) ([data confidentiality](#), [data integrity](#), [authentication](#), and [non-repudiation](#)) are also central to cryptography.^[5] Practical applications of cryptography include [electronic commerce](#), [chip-based payment cards](#), [digital currencies](#), [computer passwords](#), and [military communications](#).

从安全通信说起

Alice
参与方A



Hi Bob, Can I take a preview
of your new song?



Eve/
Mallory
信道监听者

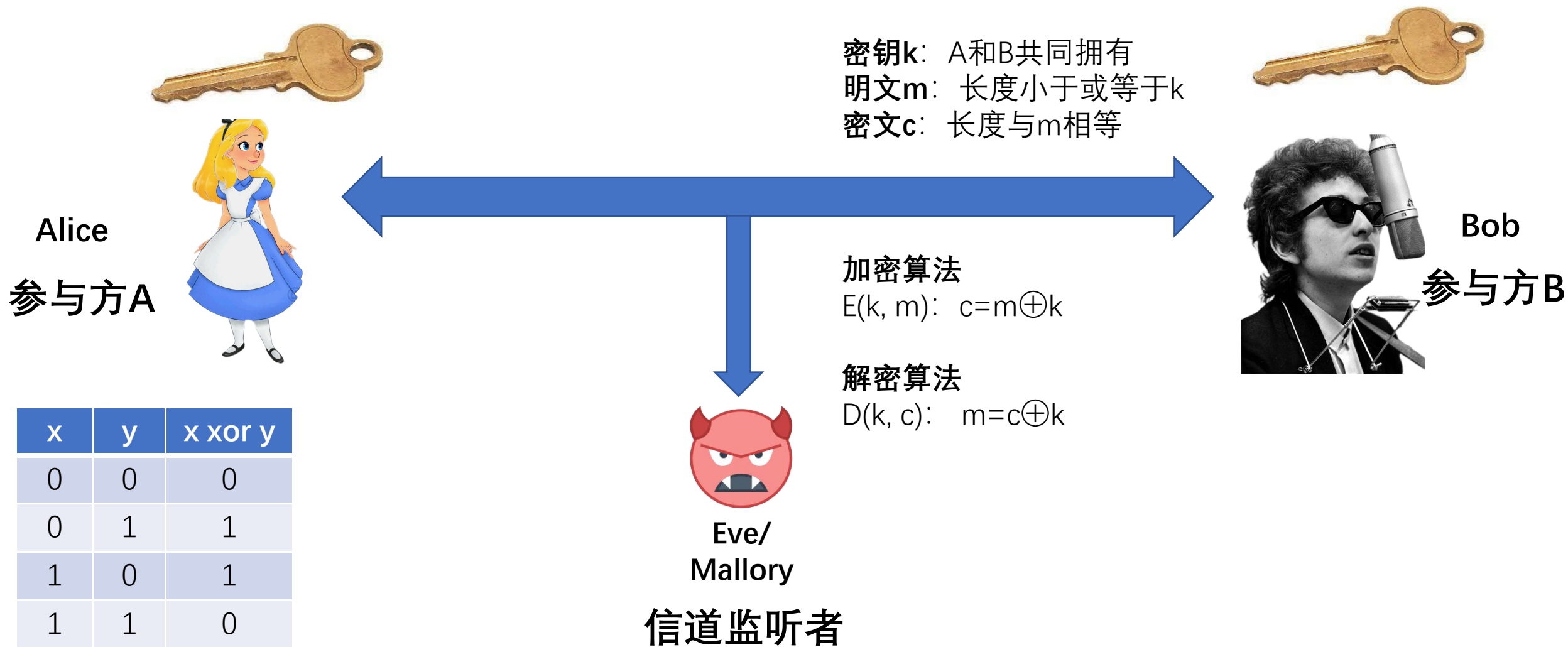
明文直接通信，面临
信道监听时，通信内
容毫无秘密可言！

How many roads
must a man walk
down
Before you call
him a man?
How many seas
must a white
dove sail
Before she sleeps
in the sand?
.....



Bob
参与方B

一次一密 one-time pad

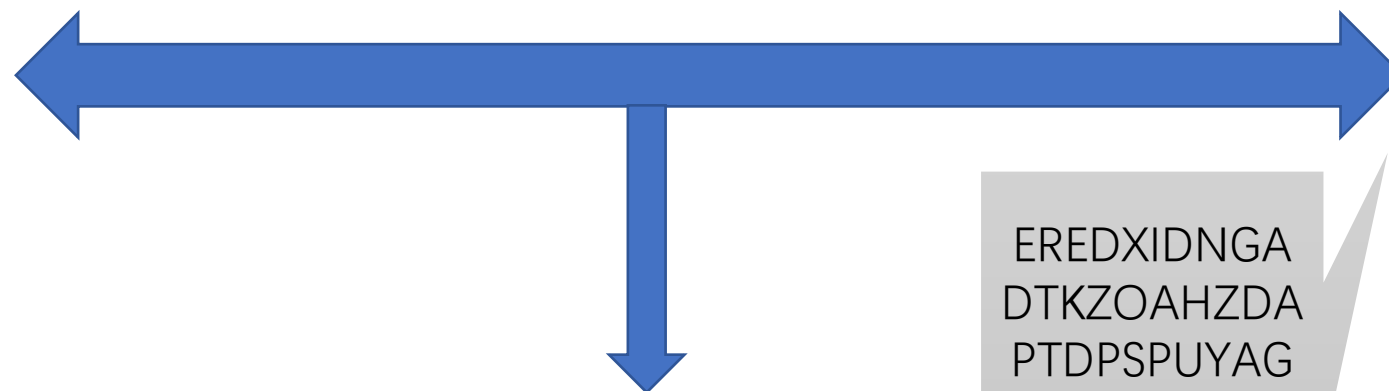


一次一密 one-time pad

How many roads must a man walk down Before you call him a man? How many seas must a white dove sail Before she sleeps in the sand?.....

xor xDlrx6vFwSAaBYFWHhNdnT1TsfpBylzCboVxDWqyqHSiKnfaF9ujFtoOlnNYPdRNnsjverC8FAItnwGHM8p5TVZQSogUzaGijg70.....

EREDXIDNGADTKZOAHZDAPTDPSPUYAGGCMBBKKAQSDPSZFMFHQTPAOVLFCPVDHFLJRLZVJDWOROGPXQXANHWGNY.....



x	y	x xor y
0	0	0
0	1	1
1	0	1
1	1	0

如果加密使用的随机
数序列对敌手保密，
信道内容和随机序列
在统计意义上无法区
分！ => 完美机密性！



Eve/
Mallory
信道监听者

EREDXIDNGA
DTKZOAHZDA
PTDPSPUYAG
GCMBBKKAQS
DPSZFMFHQT
PAOVLFCPVD
HFLJRLZVJDW
OROGPXQXA
NHWGNYRLK
MY.....

一次一密的固有问题

- 完美信息论安全
 - 对于一个密文 c , 总有相同的概率对应到明文空间的每个明文
- 内在的问题 (Shannon 1949) :
 - 密钥的长度 $|k|$ 不少于 消息的长度 $|m|$
 - 密钥无法重复使用
 - 密钥必须完全随机生成

导致没有太多的实际应用价值



Claude Shannon

Communication Theory of Secrecy Systems*

By C. E. SHANNON

1 INTRODUCTION AND SUMMARY

The problems of cryptography and secrecy systems furnish an interesting application of communication theory¹. In this paper a theory of secrecy systems is developed. The approach is on a theoretical level and is intended to complement the treatment found in standard works on cryptography². There, a detailed study is made of the many standard types of codes and ciphers, and of the ways of breaking them. We will be more concerned with the general mathematical structure and properties of secrecy systems.

Engineering Impossibility => Cryptographic Opportunities



Moti Yung
ACM/IACR/IEEE fellow

如果一次一密的问题是固有的，我们该如何让互联网中的A和B安全通信呢？

Engineering impossibility brings opportunities to cryptographers. --- M. Yung



目前应用最广泛的密码协议 TLS

TLS-ECDHE-RSA-AES128-GCM-SHA256

应用的密码技术：

ECDHE => 密码交换

RSA => 数字签名

AES => 分组加密


AES-GCM => 认证加密模式

SHA256 => 密码学哈希函数

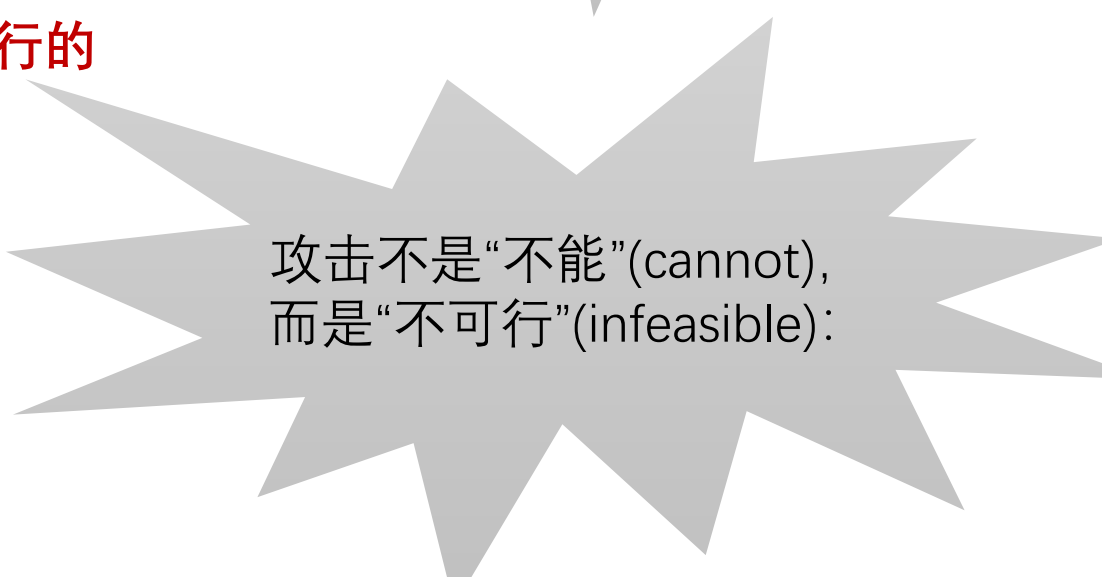
从完美
安全到
有条件
(假设)的
安全性

伪随机生成器PRG => 序列加密

- 函数 $G: \{0,1\}^L \rightarrow \{0,1\}^M$
- G 是伪随机生成器, 如果:
 - 扩展性: $L < M$, 即要求输出长度大于输入长度
 - 可计算性: G 是多项式时间内可计算的
 - 伪随机性: 如果 G 的输入服从均匀分布, 对于任意的**概率多项式算法** D , 区分 G 的输出分布和均匀分布是**不可行的**



攻击者的计算资源
不是无限的



攻击不是“不能”(cannot),
而是“不可行”(infeasible):

伪随机生成器PRG => 序列加密

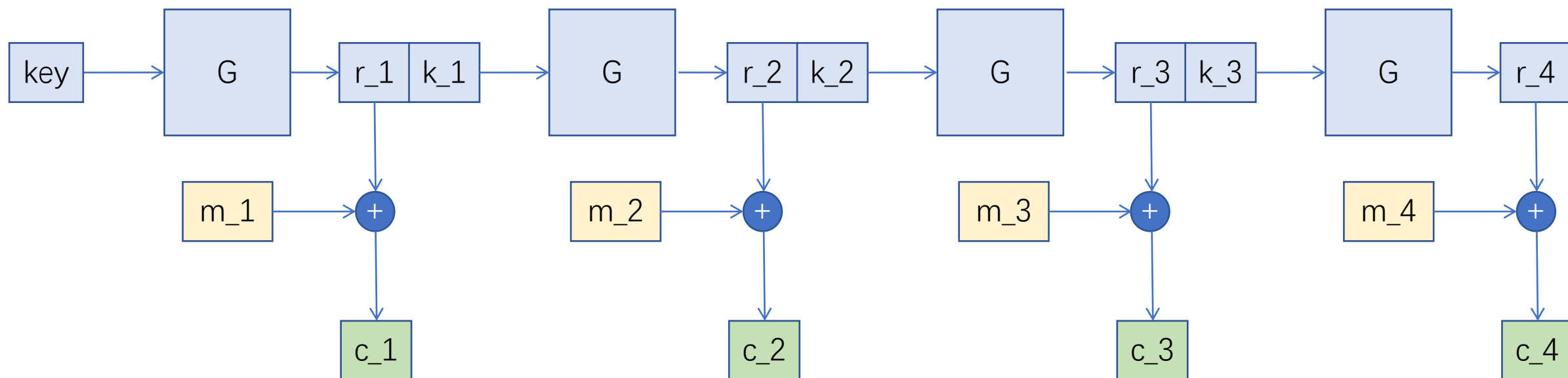
- 函数 $G: \{0,1\}^L \rightarrow \{0,1\}^M$
- G 是伪随机生成器, 如果:
 - 扩展性: $L < M$, 即要求输出长度大于输入长度
 - 可计算性: G 是多项式时间内可计算的
 - 伪随机性: 如果 G 的输入服从均匀分布, 对于任意的**概率多项式算法** D , 区分 G 的输出分布和均匀分布是**不可行的**



思考: PRG 存在么?

伪随机生成器PRG => 序列加密

- 如果存在 伪随机生成器 G ，可以有如下的 加密方案



- Magic! ! 用一个短的固定长度的随机密钥，长生了更多的随机的密钥序列!

攻击不可行 \leq 攻击成功概率是可忽略函数

- 例子：伪随机生成器 $G: \{0,1\}^L \rightarrow \{0,1\}^M$
 - 如果 $L = 1$ ，攻击者能将 $G(U_L)$ 的分布式和均匀分布 U_M 区分成功么？怎么做到？
 - 如果 $L = 128$ ，攻击者又该怎么做？之前的方法还有效么？
 -

攻击不可行 \leq 攻击成功概率是可忽略函数

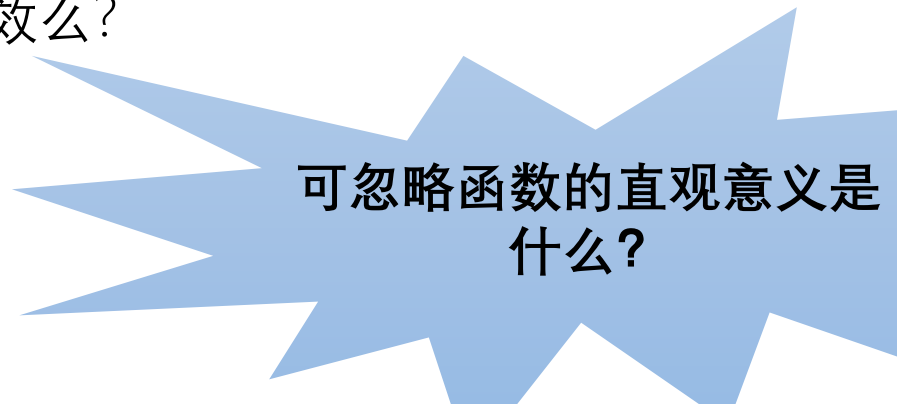
- 例子：伪随机生成器 $G: \{0,1\}^L \rightarrow \{0,1\}^M$
 - 如果 $L = 1$ ，攻击者能将 $G(U_L)$ 的分布式和均匀分布 U_M 区分成功么？怎么做到？
 - 如果 $L = 128$ ，攻击者又该怎么做？之前的方法还一定有效么？
 -
- 我们这样定义“不可行”：攻击成功的概率 $< \text{negl}(L)$
- 这里 $\text{negl}(L)$ 代表 L 的**可忽略函数 (negligible function)**
- L 的可忽略函数：对于 L 的任意多项式函数 $\text{poly}(L)$ ，一定存在某个 L_n ，如果 $L > L_n$ ，则有
$$|\text{negl}(L)| < 1 / \text{poly}(L)$$
- 另一个等价的定义：对于所有的正整数 c ，一定存在某个 L_n ，如果 $L > L_n$ ，则有
$$|\text{negl}(L)| < 1 / L^c$$

攻击不可行 \leq 攻击成功概率是可忽略函数

- 例子：伪随机生成器 $G: \{0,1\}^L \rightarrow \{0,1\}^M$

- 如果 $L = 1$ ，攻击者能将 $G(U_L)$ 的分布式和均匀分布 U_M 区分成功么？怎么做到？
- 如果 $L = 128$ ，攻击者又该怎么做？之前的方法还一定有效么？
-

- 我们这样定义“不可行”：攻击成功的概率 $< \text{negl}(L)$
- 这里 $\text{negl}(L)$ 代表 L 的**可忽略函数 (negligible function)**



可忽略函数的直观意义是什么？

- L 的可忽略函数：对于 L 的任意多项式函数 $\text{poly}(L)$ ，一定存在某个 L_n ，如果 $L > L_n$ ，则有
$$|\text{negl}(L)| < 1 / \text{poly}(L)$$
- 另一个等价的定义：对于所有的正整数 c ，一定存在某个 L_n ，如果 $L > L_n$ ，则有
$$|\text{negl}(L)| < 1 / L^c$$

计算安全性 vs 完美安全性

	一次一密	基于PRG的序列加密
敌手能力的限制	没有限制	只有多项式时间的计算能力
安全新的强度	攻击者不能区分密文的分布和随机分布	攻击者计算资源受限时，区分成功的概率很小，是密钥长度（安全参数）的可忽略函数

- 计算安全性定义的实际意义：

- 如果每隔一段时间，我们线性地增长安全参数：
 - 密码算法使用者的计算代价只会多项式的增长，这对密码使用者而言，是可接受的
 - 攻击者的成功概率指数减少（或亚指数减少），或者说需要指数增长（或亚指数增长）的计算资源来维持攻击成功概率，这是攻击者无法接受的代价

哈希函数、数字签名

密码学哈希函数

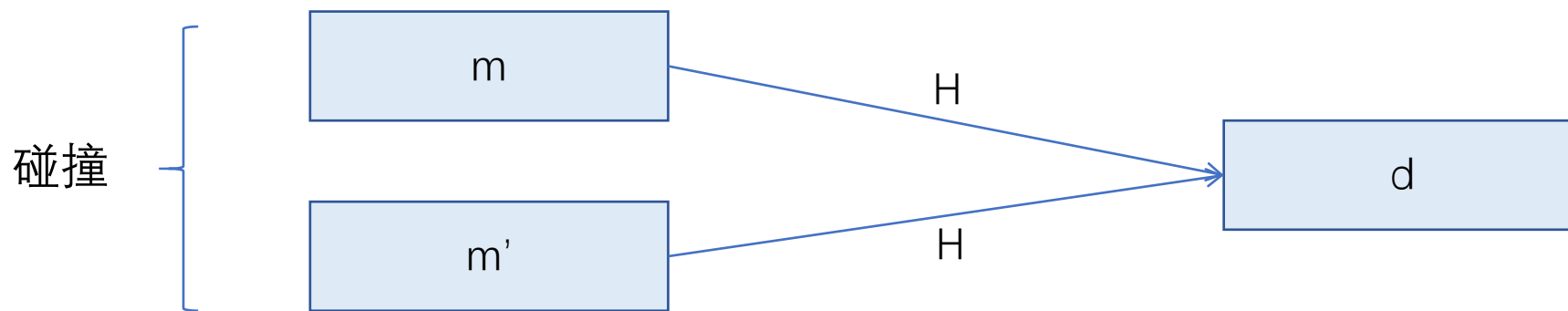
- 函数 $H: \{0,1\}^* \rightarrow \{0,1\}^k$



- H 是密码学哈希函数（或者叫做杂凑函数），如果具有：

- 抗碰撞性 collision-resistance

- 攻击者在多项式时间内，找到两个不相同的 m 和 m' 使得 $H(m)=H(m')$ 是**不可行的**



常用的哈希函数

- SHA256

- 构造区块链所用的主要杂凑函数
- 用于设计工作量证明的共识机制、生成比特币地址

- RIPEMD160

- 主要用于生成比特币地址：
RIPEMD160(SHA256(K))

- Keccak256

- 主要用于生成以太坊地址
Keccak256(K)[-160:]

- SM3 (杂凑函数算法国家标准)

- 安全性及效率与SHA-256相当
- 多用于国产区块链系统

区块链常用哈希算法对比

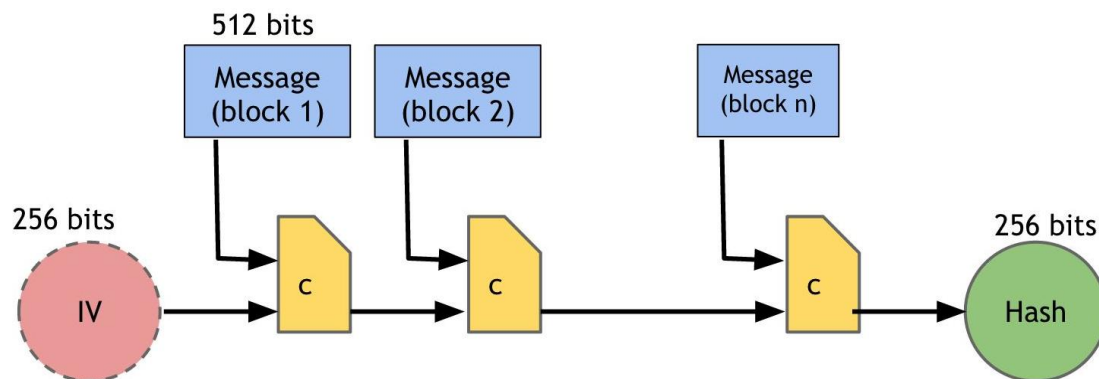
算法名称	输入长度 (比特)	分组长度 (比特)	基本字长 (比特)	输出长度 (比特)
SHA-256	$< 2^{64}$	512	32	256
RIPEMD-160	$< 2^{64}$	512	32	160
Keccak-256	不限	1088	64	256
SM3	$< 2^{64}$	512	32	256

哈希函数在区块链的应用



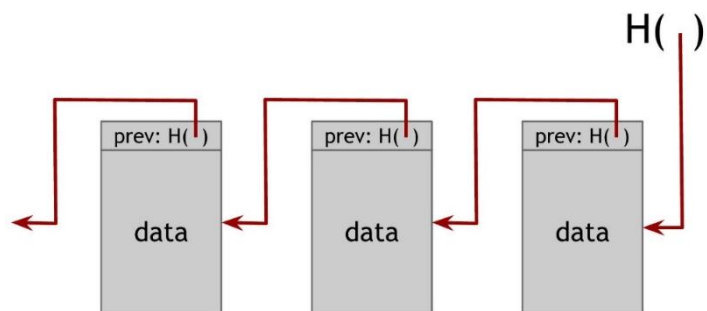
1. “产生账户地址”

- 将公钥信息转换为更短的账户地址



2. “数据存证”

- 将任意长度的消息分拆成 n 个组，与杂凑初始化值 IV 一起作为函数的输入，得到文件存证的摘要值



3. “哈希指针”与“数据结构”

- 指示信息存储位置
- 提供信息检索功能
- 防止任意修改

哈希函数在区块链的应用： PoW

- **工作量证明 (Proof of Work, PoW)**

对于给定的 T 和 $prev$, 找到 x , 使得 $H(x \parallel prev) < T$, x 即工作量证明

我们期望的安全性： 找到 满足条件的 x 的概率 和 计算 H 的次数 是等比例变化的

- 抗原像攻击和抗碰撞攻击不足以满足 PoW 的理想安全性！ **为什么？**

- *两性质没有保证输出是无关于输入的随机值*
- **能否给出不满足安全性的反例？**

- 我们可以在哈希函数的 **随机预言机 (random oracle)** 理想模型下 证明 PoW 的安全性

- 对任何一个新的输入， 都回传一个均匀且随机的输出

数字签名

可以类比于现实生活中签名/盖章的理想版：

- 可以被公开验证
- 无法被本人以外的人恶意伪造

• 一个数字签名机制，通常包含 (Gen, Sign, Verify) 三个算法：

- $(pk, sk) \leftarrow \text{Gen}(1^\lambda)$ ： **密钥生成算法**。输入安全参数的一元表示 1^λ ，输出一对密钥，其中 pk 称为公钥或验证密钥， sk 称为私钥或签名密钥。
- $\sigma \leftarrow \text{Sign}(sk, m)$ ： **签名算法**。输入私钥 sk 和消息空间 \mathcal{M} 中的消息 m ，输出签名 σ ，表示为 $\sigma = \text{Sign}_{sk}(m)$ 。
- $b \leftarrow \text{Vrfy}(pk, m, \sigma)$ ： **签名验证算法**。输入公钥 pk ，消息 m 和签名 σ ，输出比特 b ，取0表示签名无效，取1表示签名有效，表示为 $b = \text{Vrfy}_{pk}(m, \sigma)$ 。



数字签名

- 什么才是安全的数字签名机制？
- **数字签名方案的安全性要求即使概率多项式时间的敌手可以获得若干有效的消息签名对，也无法可行地伪造出一个新的有效的消息签名对 (m, σ) 。**可细分为：
 - 根据攻击目标：
 - 如果消息签名对的 m 没有被询问过签名，则称数字签名方案**存在性不可伪造**。
 - 如果消息签名对 (m, σ) 未在询问阶段出现过，则称数字签名方案**强不可伪造**。
 - 根据敌手对消息的控制程度：
 - **随机消息攻击**：敌手对获得的签名消息没有任何控制，消息是随机选择的。
 - **已知消息攻击**：敌手对获得的签名消息具有有限的控制，可以选择消息并获得相应的有效签名，但他比较提前指定这些消息，独立于签名者的公钥以及任何他可以观察到的后续签名。
 - **自适应选择消息攻击**：敌手对获得的签名消息具有完全的控制，可以在得到签名者的公钥之后选择消息，并可以在得到一些对所选消息的签名之后继续选择消息，得到相应的有效签名。

数字签名

安全性定义

攻击游戏（自适应选择消息攻击下的存在性不可伪造） 对于一个数字签名方案 $\mathcal{DS} = (\text{Gen}, \text{Sign}, \text{Vrfy})$, 给定一个敌手 \mathcal{A} , 定义以下实验。

1. 挑战者计算 $(pk, sk) \leftarrow \text{Gen}(1^\lambda)$, 把 pk 发送给敌手。
2. 设有一个签名预言机 $\text{Sign}_{sk}(\cdot)$, 敌手可以与预言机交互, 请求对他自适应选择的消息进行签名, 令 M 表示敌手向预言机请求签名的消息的集合, 敌手最终输出 (m, σ) 。
3. 若 $\text{Vrfy}_{pk}(m, \sigma) = 1$ 且 $m \notin M$, 则称**敌手成功**。

定义（自适应选择消息攻击下的存在性不可伪造） 记为 **EUF-CMA**。如果对于所有概率多项式时间的敌手 \mathcal{A} , 在攻击游戏中 \mathcal{A} 成功的概率总是可忽略的, 则称数字签名方案 \mathcal{DS} 在自适应选择消息攻击下是存在性不可伪造的。

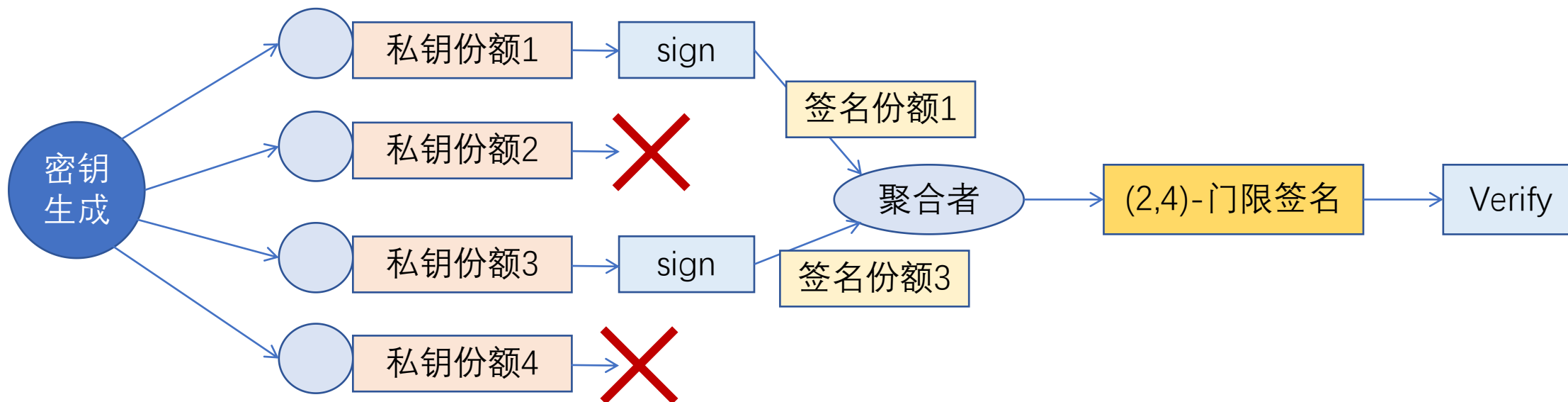
常用的数字签名

- **RSA签名**：基于大质数分解问题的困难性
 - 目前要求至少2048比特，尺寸太大、计算较慢，区块链中应用较少
- **Schnorr签名**：离散对数假设 + 随机预言机模型
 - 比特币在近期升级后已经引入，具有计算简洁高效、方便聚合、批量验证加速等优势
- **BLS签名**：co-CDH假设 + 随机预言机模型
 - 在Dfinity等明星项目使用，能够方便聚合、门限密码体系兼容、签名唯一性 等优势
- **ECDSA签名**：没有标准假设下的安全性证明（需要GGM或AGM假设）—— 产业标准
 - 在比特币、以太坊等项目中广泛使用，在转账时需要提供有效签名

门限签名

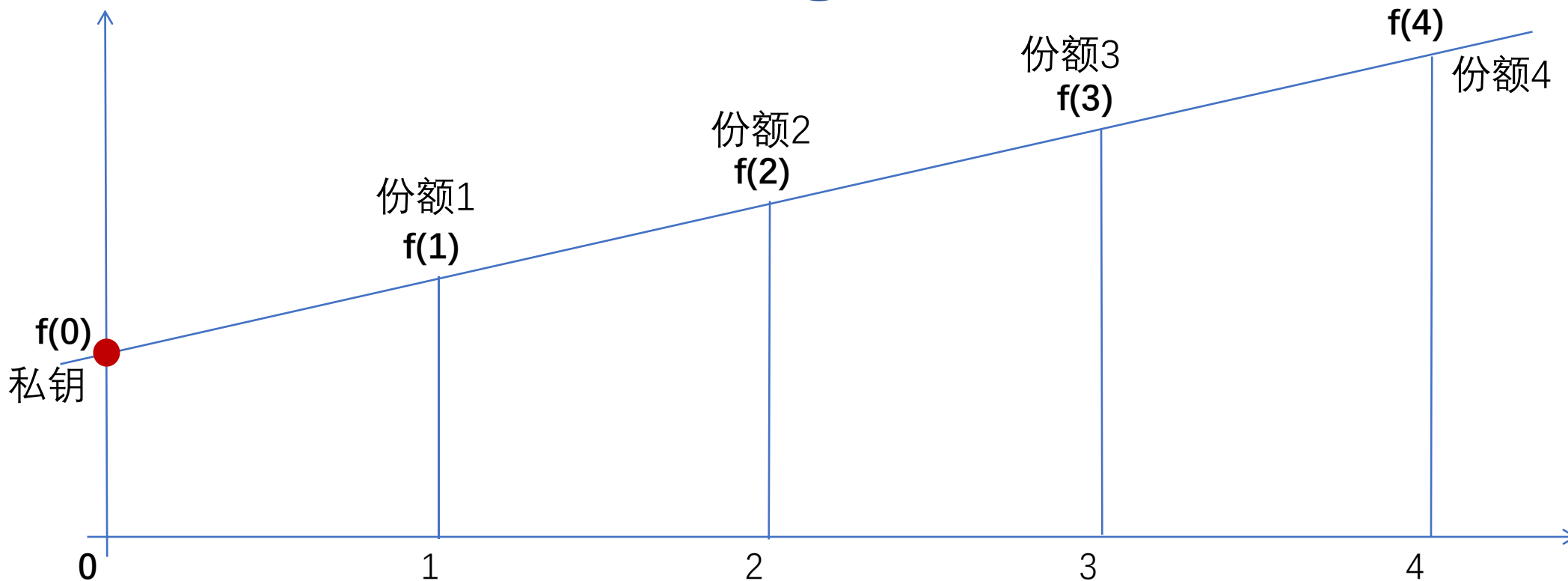
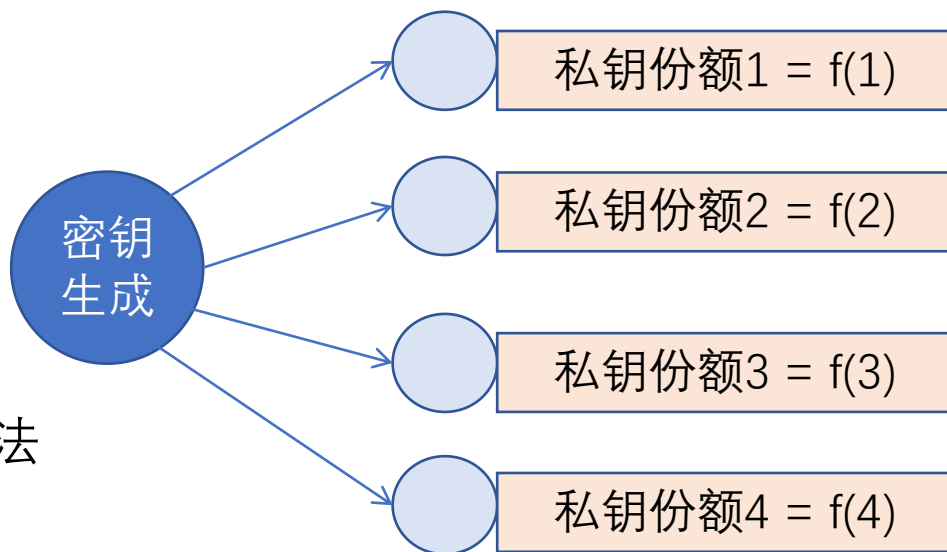
门限签名的初步概念（把签名的权利拆分）

- 允许多个实体共享在单个公钥下发布签名的能力。
- 在 (t, n) -门限签名方案中，共有 n 个实体分别持有不同的私钥份额。
- 只有大于等于 t 的实体集合才能生成一个有效的签名。



门限签名

Shamir 秘密分享：
作为拆分私钥的方法



门限签名

拉格朗日差值（以点1和点3为例）：

观察：

$$f_1(x) = (x-x_3)/(x_1-x_3) \Rightarrow \text{在 } x_1 \text{ 为 } 1、\text{在 } x_3 \text{ 为 } 0$$

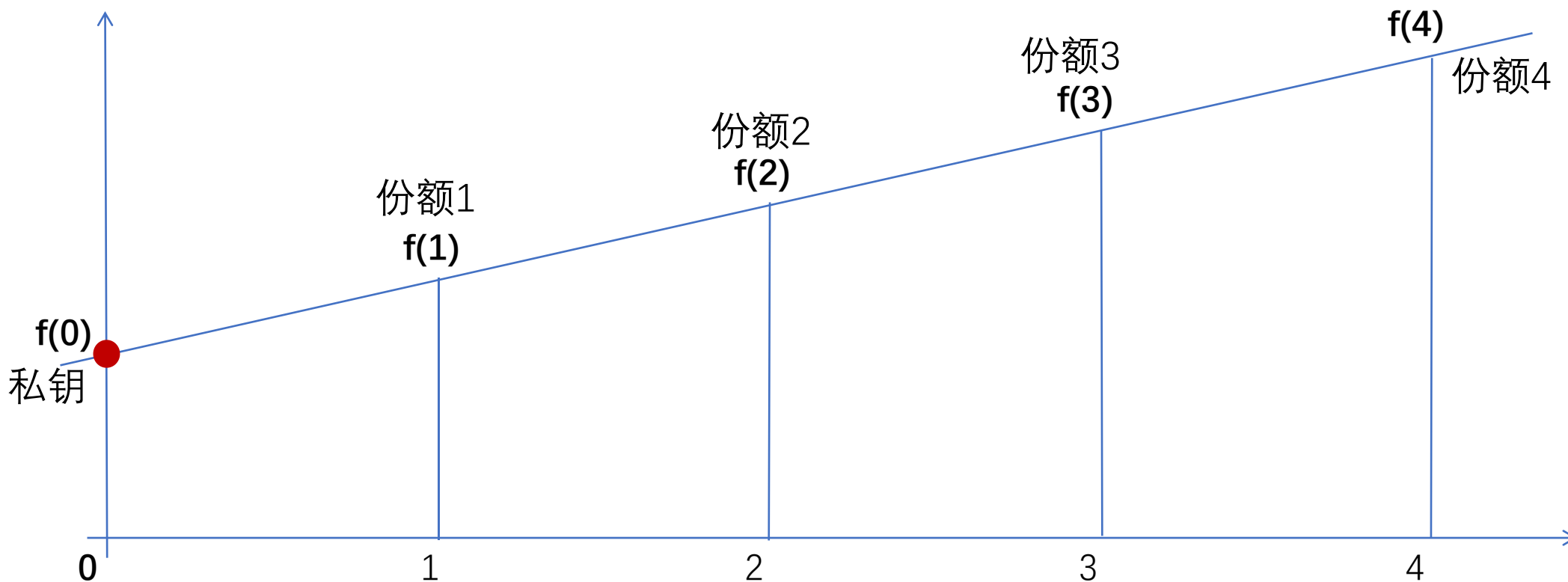
$$f_3(x) = (x-x_1)/(x_3-x_1) \Rightarrow \text{在 } x_3 \text{ 为 } 1、\text{在 } x_1 \text{ 为 } 0$$

Shamir 秘密分享：

作为拆分私钥的方法

$f(x)$ 一定可以写做 f_1 和 f_3 的线性组合地形式：

$$f(x) = f(1)*f_1(x) + f(3)*f_3(x)$$



门限签名

拉格朗日差值（以点1和点3为例）：

观察：

$$f_1(x) = (x-x_3)/(x_1-x_3) \Rightarrow \text{在 } x_1 \text{ 为 } 1, \text{ 在 } x_3 \text{ 为 } 0$$

$$f_3(x) = (x-x_1)/(x_3-x_1) \Rightarrow \text{在 } x_3 \text{ 为 } 1, \text{ 在 } x_1 \text{ 为 } 0$$

(2, 4) - BLS 门限签名：

节点1的签名 = $H(m)^{f(1)}$

节点3的签名 = $H(m)^{f(3)}$

$f(x)$ 一定可以写做 f_1 和 f_3 的线性组合地形式：

$$f(x) = f(1)*f_1(x) + f(3)*f_1(x)$$

$$\text{BLS 门限签名} = [H(m)^{f(1)}]^{f_1(0)} * [H(m)^{f(3)}]^{f_3(0)} = H(m)^{f(1)*f_1(0) + f(3)*f_1(0)} = H(m)^{f(0)}$$

BLS 门限签名验证： $e(H(m)^{f(0)}, g) = e(H(m), g^{f(0)})$

如何扩展到 (t, n) - BLS 门限签名？

区块链钱包

基本原理： 数字签名

- 回顾数字签名：

- 有一对公私钥 pk 和 sk
- sk 私钥的拥有者 \Rightarrow 能够计算对 pk 有效的数字签名
- 攻击者 \Rightarrow 难以可行地伪造针对 pk 的数字签名

抗抵赖性

(有效的签名必定来自拥有 sk 的实体)

- 天然地，采用数字签名作为身份认证的技术

- 每个账户 或者 每笔余额 都绑定一个 pk (或 pk 的哈希值，称为地址)
- 想从账户转出 需要提供能够通过 pk 验证的 有效的数字签名

- **难点：** 可靠和安全的存储和使用私钥 $sk \Rightarrow$ “钱包”



私钥、公钥与椭圆曲线

- 区块链使用的签名方案主要是基于椭圆曲线群，比如比特币和以太坊都采用secp256k1曲线。
- **私钥** (256比特): x
- **公钥** (secp256k1椭圆曲线上的有理点): $G^x = G * G * \dots * G * G$
- secp256k1椭圆曲线定义在素域 F_p , 其中

$p = \text{FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFC2F}$

- 曲线方程:

$E: y^2 = x^3 + ax + b \pmod p$, 其中

$a = \text{00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000}$

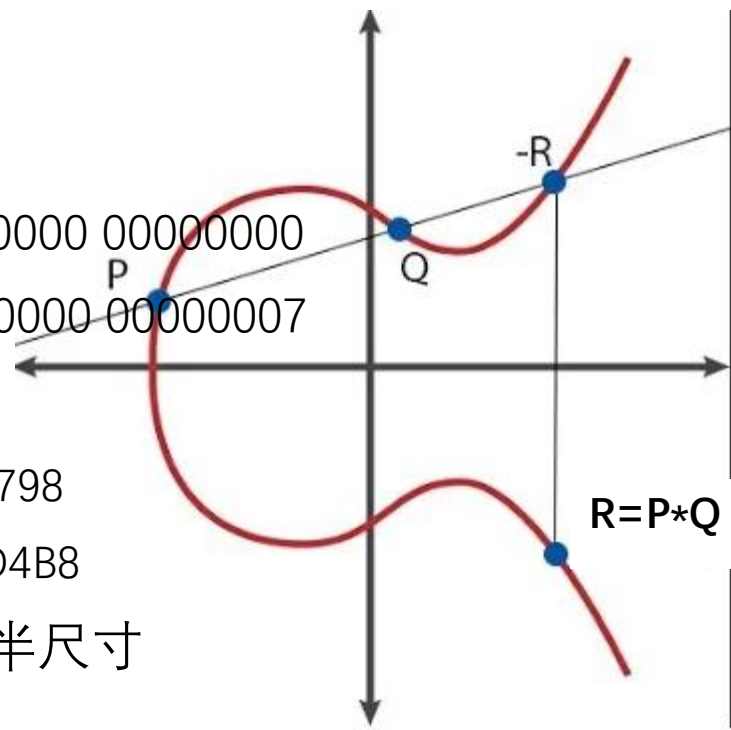
$b = \text{00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000007}$

- 基点G (生成元)

$G_x = \text{0x79BE667EF9DCBBAC55A06295CE870B07029BFCDB2DCE28D959F2815B16F81798}$

$G_y = \text{0x483ADA7726A3C4655DA4FBFC0E1108A8FD17B448A68554199C47D08FFB10D4B8}$

- **公钥的压缩模式**: 只需要公钥的x坐标、以及y坐标的奇偶信息, 减少一半尺寸



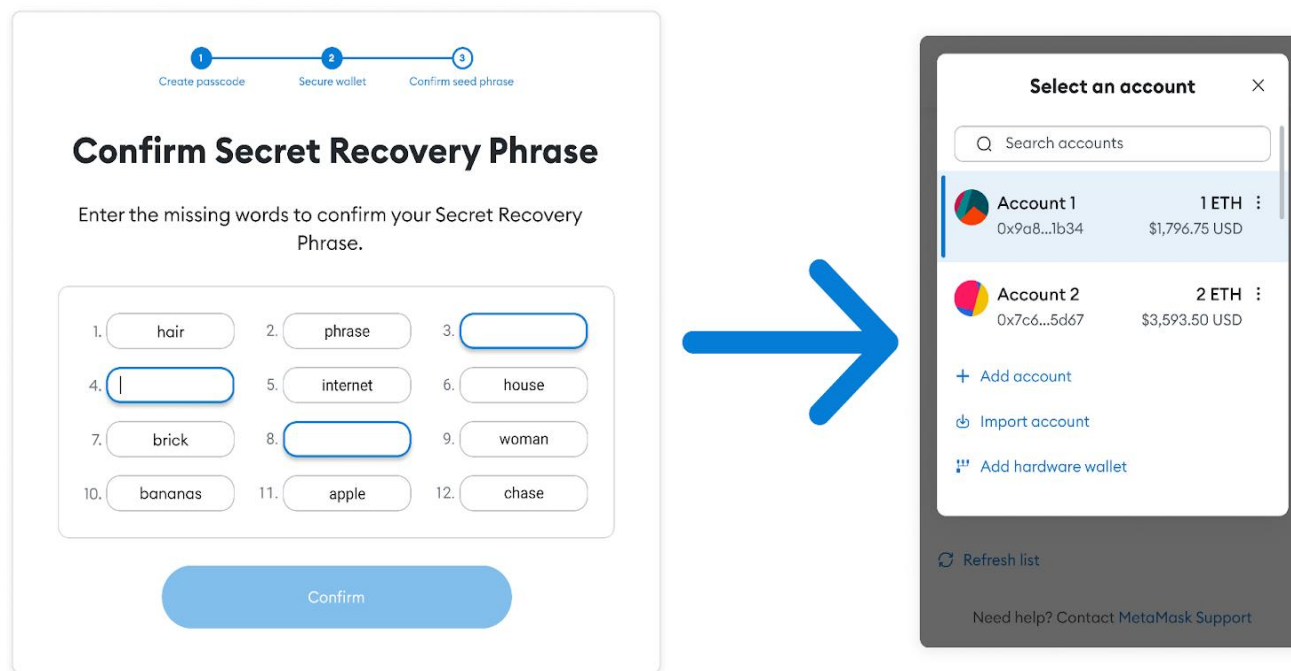
热钱包

- 助记词 (密钥恢复语 Secret Recovery Phrase), 比如metamask使用12个短语

pass
original
rally
buzz
input
sustain
.....

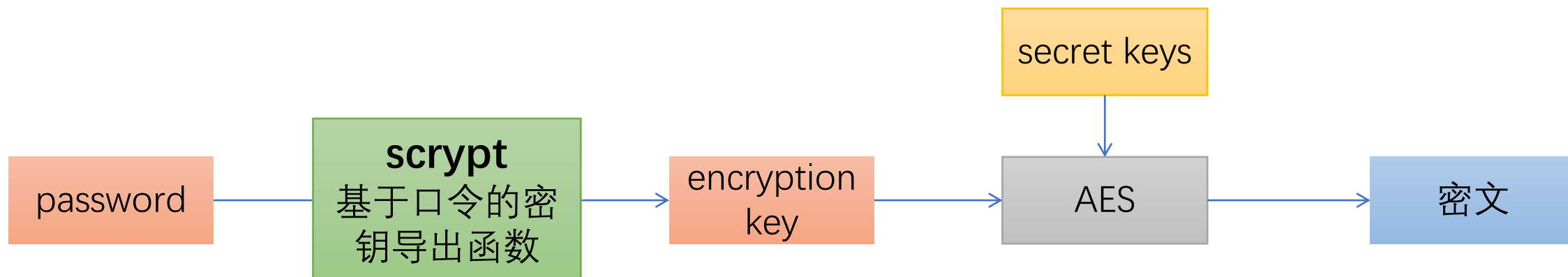
- 在metamask钱包中, 用户的私钥从2048个短语中随机选取的12个助记词导出

- 导出密钥的方式成为基于口令的密钥导出函数
Password-Based Key Derivation Function 2

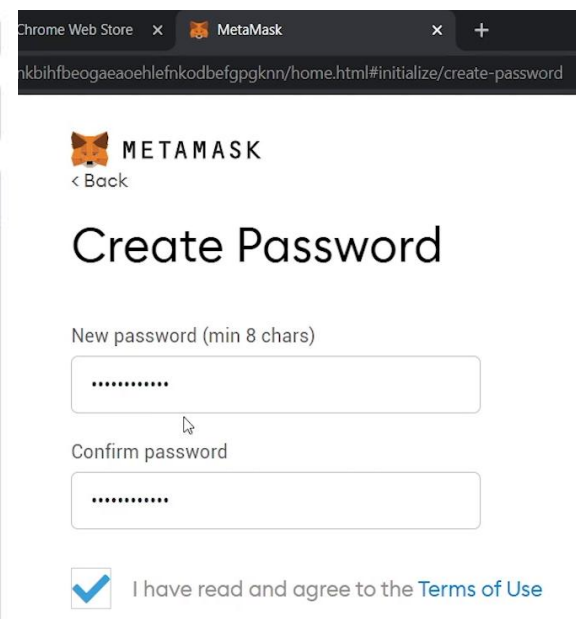
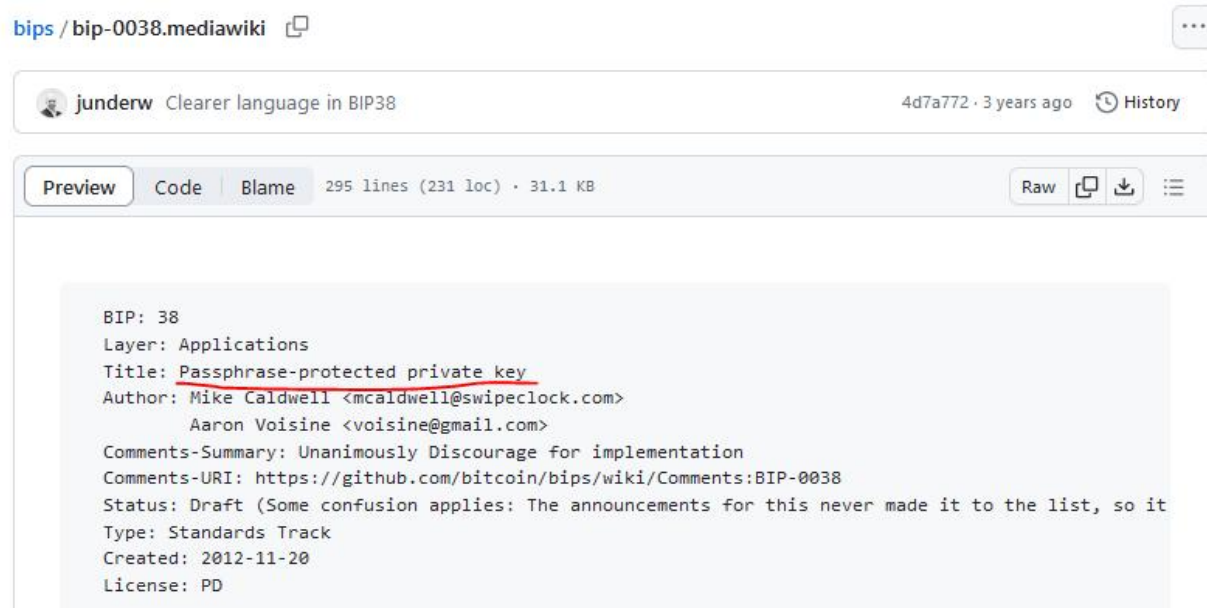


热钱包

- 口令加密 (passphrase-based encryption) ，为私钥在软件环境下提供机密存储



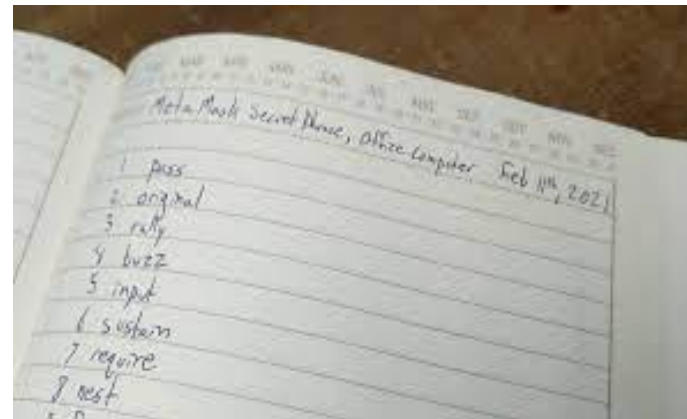
为抵抗暴力破解，我们期望
PBKDF资源开销较大并且难
以硬件加速，激发了近期流
行的一系列研究：
内存困难哈希函数(memory-
hard hash function)



冷钱包

- “纸钱包”:

- ✓ 用笔记本记录下助记词
- ✓ 打印私钥的二维码
- ✓ 私钥离开了在线的终端设备 => 更难泄密
- ✓ 但纸上记录的格式也变得更难使用



SECRET



KxwQRxC1X1s3jF7bdL2ToGSYhhFMhVNMCMzFkgCq7WHfR8bGLmt9

- 硬件钱包:

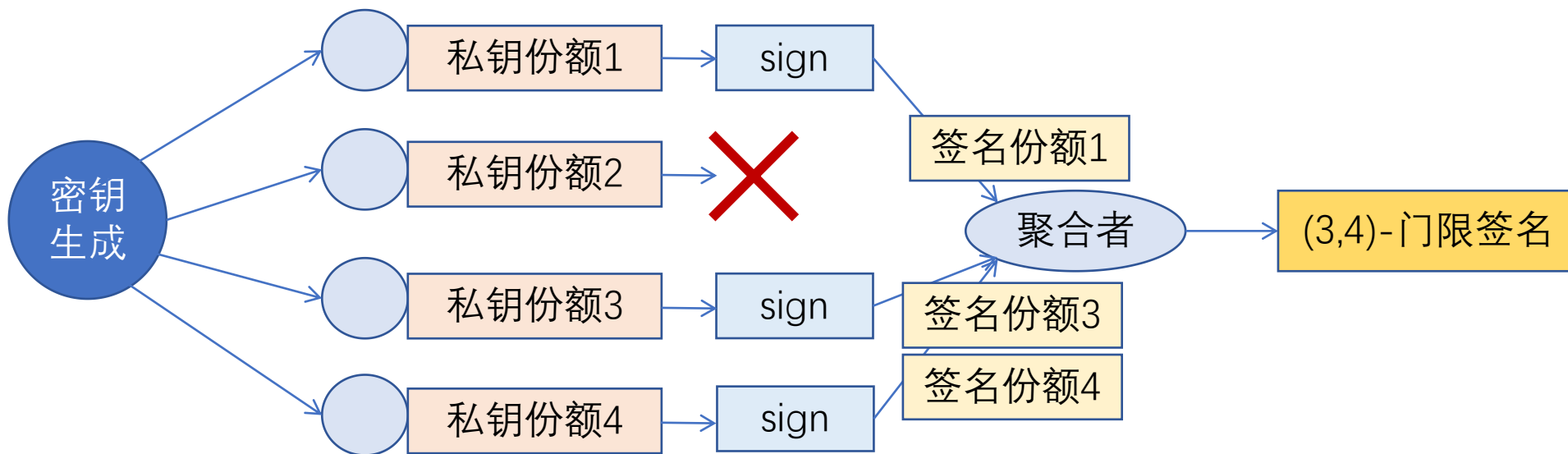
- ✓ 私钥存储在特定的硬件设备
- ✓ 设备经过特殊的“硬化”（比如TPM等可信硬件）
- ✓ 硬件设备可以给特定的信息签名
- ✓ 类比银行UKey



多方钱包

- 利用门限签名技术：

将私钥秘密分享给4个服务器节点，3个节点才能给出正确的门限签名
只要系统中还有2个服务器没有比敌手攻击成功，私钥就是安全的



练习题

- 以下哪些是 x 的可忽略函数？

$$f(x) = 1 / x^2$$

$$f(x) = 1 / 2^x$$

$$f(x) = 1 / 2^{20 \cdot \log x}$$

$$f(x) = x^{-\log \log \log x}$$

- 给定密码学哈希函数 H ，以下哪些函数 F 也是密码学哈希函数？解释 F 为什么满足或不满足哈希函数的安全性质。

$$F(x) = H(x) || 0000$$

$$F(x) = H(H(x))$$

$$F(x) = H(x) || H(x)$$

$$F(x) = H(0000 || x)$$

- 在metamask钱包中，用户的主私钥从2048个公开的短语中随机选取的12个助记词导出，攻击者通过暴力搜索助记词破解主私钥的搜索空间有多大。（注意：不同顺序的相同助记词导出的主私钥也不同）

练习题

- 比特币的私钥为
e931897499a4835c125c86078a3eaaf1c7077f3a187adb56e553ed9a9bbeef79, 试计算其对应的公钥和地址, 计算对字符串“hello world”的签名。(结果均用十六进制表示、不用加校验位与编码, 可以编程。)
- 有3次多项式 $f(x) = a*x^3 + b*x^2 + c*x + d$; 并且
 $f(1)=369$
 $f(2)=1404$
 $f(3)=3739$
 $f(5)=14709$
用拉格朗日插值法求 $f(0)$ 。

其他密码学工具(进阶)

密码学承诺

数字保险箱：发送者将某物品放进保险箱后发送给接收者，此时发送者无法再替换保险箱中的内容，只能将“钥匙”再发送给接受者，如实打开其中的物品；

承诺方案包括三个算法：

Setup: 产生公共参数

Commit(x, r) \rightarrow c: 输入被承诺的值 x 、随机数 r ，输出承诺 c

Open(c, x, r) \rightarrow {0,1}: 输入 c 、 x 和 r ，输出0或1，分别代表拒绝或接受 x 为被承诺的值

承诺方案的安全性质：

隐藏性 hiding: 承诺 c 不泄露关于 x 的信息

绑定性 binding: 同一个 c ，不存在不相同的 x 和 x' ，使得 Open 都输出接受

典型应用：

信息存证： (1) 不会泄露被存证数据的任何内容； (2) 无法在未来变更打开存证后的数据

链上拍卖： (1) 承诺不能更改的报价； (2) 同时不泄露报价的信息。

向量承诺

最基本的**向量承诺方案**，包括：

Setup: 产生公共参数

Commit($\langle x_1, x_2, \dots, x_n \rangle$)->**c**: 输入被承诺的向量 $\langle x_1, x_2, \dots, x_n \rangle$ 、输出承诺 **c**

Open(**c**, **i**, x_i)-> π_i : 输入**c**、**i**和 x_i 以及可能的辅助数据，输出 π_i ，作为第**i**个位置承诺值的证明

Verify(**c**, **i**, x_i)-> $\{0,1\}$: 输入**c**、**i**和 x_i ，输出0或1，分别代表拒绝或接受 x_i 为被承诺向量的第**i**个值

向量承诺方案的安全性质：

位置绑定性：同一个承诺 **c**，对每个位置 **i**，不存在不同的 x 和 x' ，使得 **Verify** 都接受其为第 **i** 个位置的被承诺值

向量承诺方案在区块链的应用：

构造零知识证明的底层工具

共识机制中的可验证信息扩散；

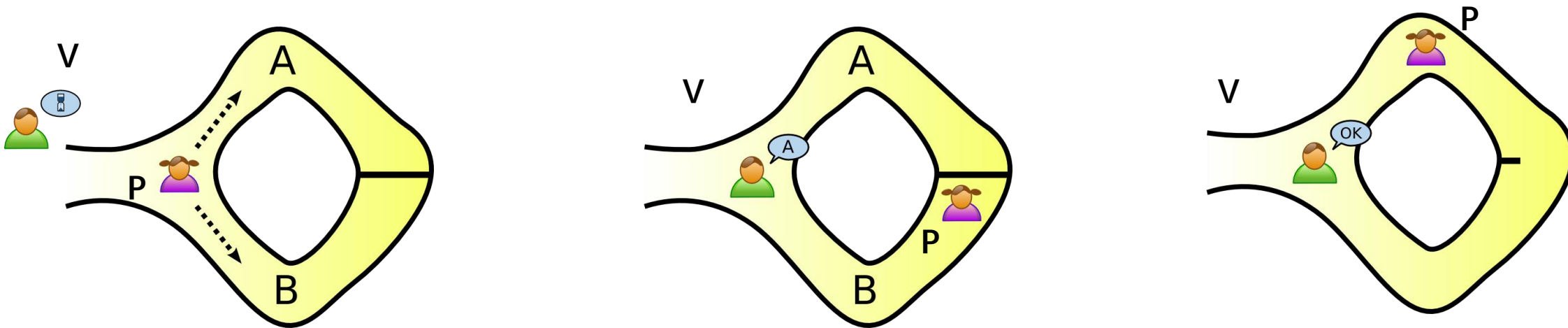
可靠存储证明；

无状态挖矿。

零知识证明

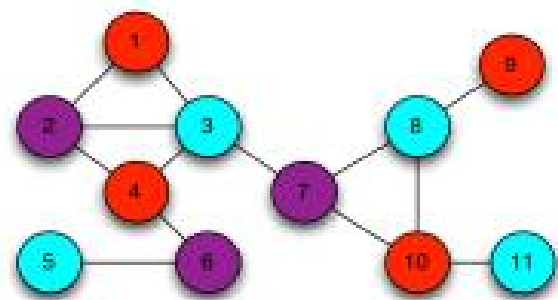
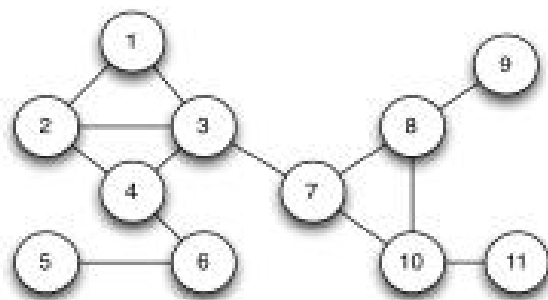
- **零知识证明** (zero-knowledge proof) : 一个**证明者P**向一个**验证者V**证明某个命题 x 的确为真, 同时不泄露该命题为真以外的其他信息。
 - **完备性 completeness**: 如果命题 x 为真, 诚实的 V 能够接收诚实 P 的证明。
 - **可靠性 soundness**: 如果命题 x 为伪, P 说服 V 接受 x 为真是计算意义上不可行的。
 - **零知识性 zero-knowledge**: 证明 (在计算意义上) 不泄露命题 x 为真以外的其他信息。

Quisquater-Guillon 洞穴故事



零知识证明

- 图的3填色 (3-color) 问题的零知识证明。
- P 尝试证明图 $G=(E,V)$ 是能够3填色的。



1.承诺



P

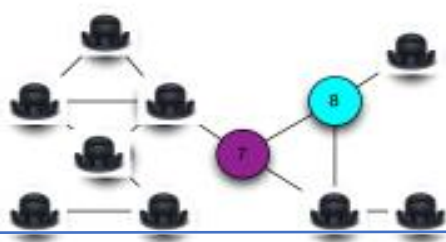
将填色的结果承诺后发给V

V

2.挑战

打开 边 7-8 的两个节点

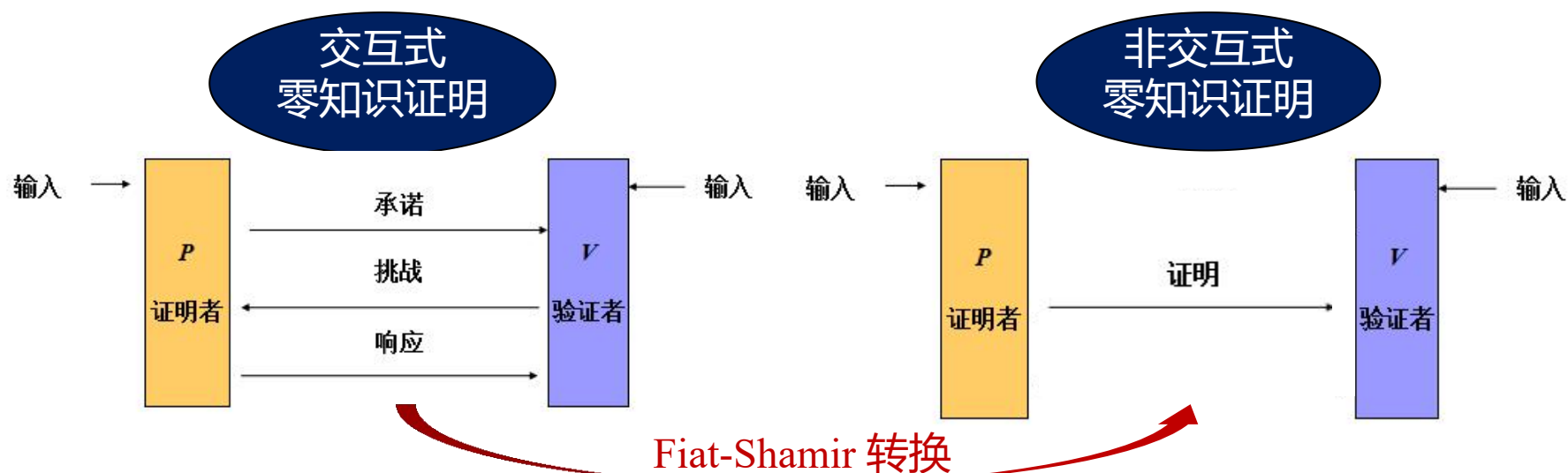
3.响应



将 节点7 和 节点8 的承诺打开

重复若干
次放大
soundness

零知识证明



- 区块链中应用更广泛的是**非交互**证明。
- 承诺-挑战-响应 式的交互式协议，可以通过 **Fiat-Shamir 变换** 变为非交互式
 - 依赖随机预言机模型产生挑战

零知识证明

非交互零知识证明中，将证明尺寸和验证者开销降低到极致 就得到了 **zk-SNARK**：

- **zero-knowledge Succinct Non-Interactive ARgument of Knowledge**

- **zero-knowledge**：零知识。
- **succinct**：简洁。生成的证据消息非常小，验证过程很高效。
- **non-interactive**：非交互。证明者和验证者无需交互，仅提供一次输入即可。

以上三点非常契合区块链下的环境（**提供隐私保护的同时、将宝贵的链上成本压缩到极致**）！

- **arguments**：证据而非证明。即soundness只针对计算资源受限的敌手才成立。
- **proof of knowledge**：知识证明 => 更强的 soundness。对于证明者来说，在不知晓秘密即witness的前提下，构建一个有效的零知识证据在计算上是不可能的。
- **NP complete languages**：通用性强，支持NP语言类中的任意关系的衍生命题。

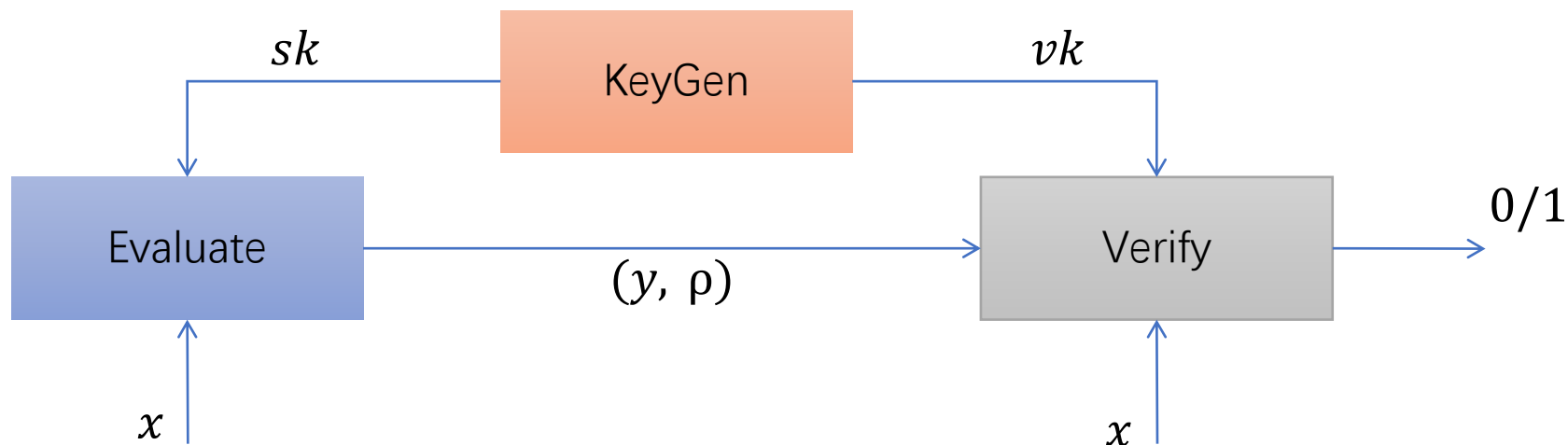
zk-SNARK在区块链的应用极其广泛：

- **Layer-2扩容 zk-rollup；隐私保护技术，如zcash**

可验证随机函数 (VRF)

VRF由三个算法组成: *Keygen*、*Evaluate* 和 *Verify*。

- $Keygen(r) \rightarrow (vk, sk)$: 对任意随机输入 r , 生成验证密钥 vk 和私钥 sk 。
- $Evaluate(sk, x) \rightarrow (y, \rho)$: 输入私钥 sk 和消息 x , 输出伪随机字符串 y 和证明 ρ 。
- $Verify(vk, x, y, \rho) \rightarrow 0/1$: 输入验证密钥 vk , 消息 x 及伪随机字符串 y 和证明 ρ , 输出结果 $0/1$, 分别代表接收 y 和拒绝 y 。



可验证随机函数 (VRF)

VRF 具有的安全性：

- **公开可验证性**：任何公钥的持有者都可以验证正确计算的VRF的有效性；
- **不可预测性**：对于不知道私钥的攻击者，VRF输出的分布和均匀分布计算不可区分；
- **唯一性**：找到 $x, y_1, \rho_1, y_2, \rho_2$ 使得 $Verify(vk, x, y_1, \rho_1) = 1$ 和 $Verify(vk, x, y_2, \rho_2) = 1$ 是计算不可行的。

构造容易、计算高效：所有的确定性签名，比如 EdDSA、BLS、RSA，在 random oracle 模型下都能平凡的构造 VRF：先计算确定性签名、再计算签名的哈希，签名为证明、哈希为随机数。

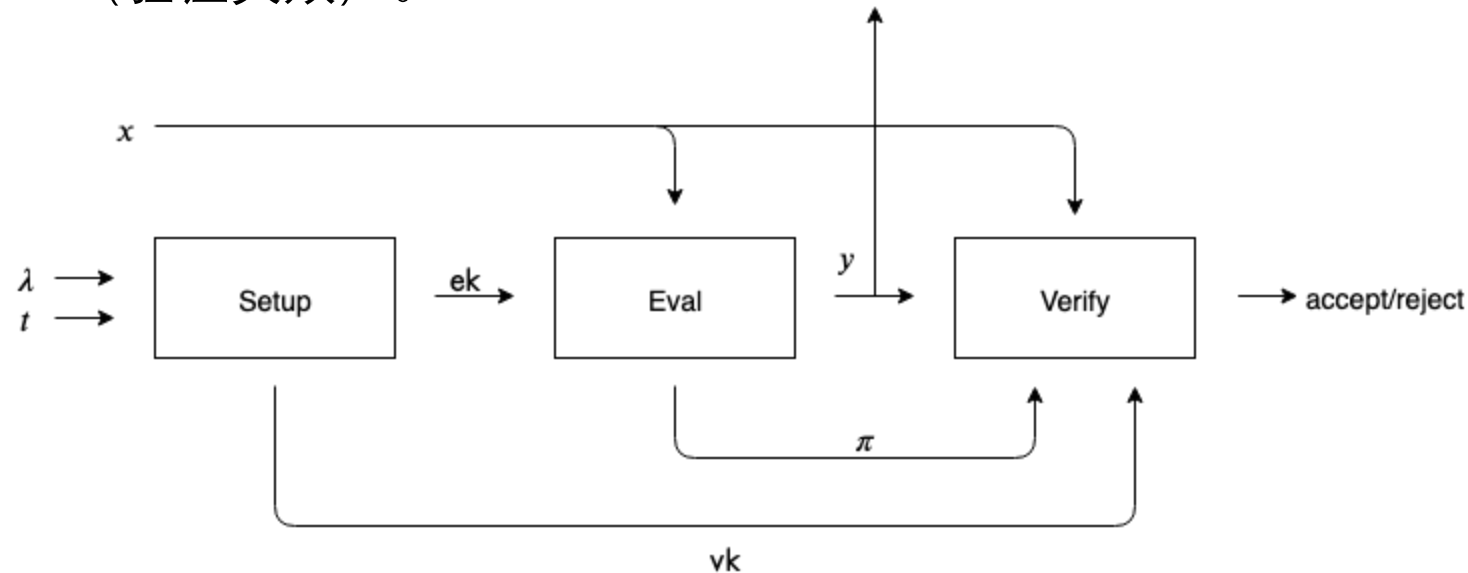
VRF 在区块链的应用：

智能合约需要的公平随机数的生成；
Algorand共识机制中的委员会选取；
分布式的公共随机数生成协议。

可验证延迟函数 (VDF)

VDF由三个算法组成: *Keygen*、*Eval* 和 *Verify*。

- $Setup(\lambda, t) \rightarrow pp = (ek, vk)$: 输入安全参数 λ 以及**时间参数** t , 生成一个公共参数 pp 。
公共参数 pp 包含了一个用于计算的参数 ek 和一个用于验证的参数 vk 。
- $Eval(ek, x) \rightarrow (y, \pi)$: 输入计算参数 ek 和 $x \in X$, 计算函数值 $y \in Y$ 和证明 π 。
- $Verify(vk, x, y, \pi) \rightarrow \{accept, reject\}$: 输入 vk , x , y 以及 π , 输出 *accept* (验证通过) 或者 *reject* (验证失败)。



可验证延迟函数 (VDF)

VDF 具有的安全性:

- **串行性 Sequentiality**: 诚实的实体能够以 t 个串行步骤计算 $(y, \pi) \leftarrow Eval(ek, x)$; 但不存在拥有线性数量并行处理器的敌手, 能够以少于 t 的步骤区分 VDF 结果和随机数。
- **高效可验证性**: 诚实实体执行验证算法 $Verify$ 应该是高效的, 其计算复杂度应为 $O(polylog(t))$ 。
- **唯一性**: VDF 的输入 x 和输出 y 满足一对一关系, 即对于任意一个 VDF 的输入 x , 难以找到一个满足验证算法 $Verify$ 的元素 y , 使得 $y \neq Eval(ek, x)$ 。

。

VDF示例:

$$y \equiv x^{2^t} \bmod \lambda$$
$$x \rightarrow x^2 \rightarrow x^{2^2} \rightarrow \dots \rightarrow x^{2^t} \bmod \lambda$$

在阶数未知的群 (如RSA群) 上进项以上计算

可验证延迟函数 (VDF)

VDF 和 PoW 计算时都需消耗较多计算资源，但是两者存在关键区别：

- 抵抗并行计算加速

- PoW无法抵抗并行计算加速：符合比特币的“一CPU一票”（one-CPU-one-vote）的设计思想。
- VDF**能够抵抗并行计算加速**：多 CPU 用户和单 CPU 用户相比，在计算VDF 时几乎没有优势。

- 解的唯一性

- PoW拥有众多**有效解**：对于固定的难度设定 d ，PoW 拥有很多合法解，从而保证 PoW 共识网络拥有稳定的吞吐量以及激励矿工进行竞争。
- VDF拥有**唯一有效解**：对于任一给定的输入 x ，VDF 拥有唯一对应的合法输出。

VDF 也具有十分广泛的应用：

- 用于增强区块链上公共可验证随机数的安全性
- 实现可验证的去中心化稳定时钟
- 副本证明（空间证明+存储证明）

谢谢!