



# 密码科学技术丛书

## 公钥加密的设计方法

作者：陈宇 & 秦宝东

组织：山东大学 & 西安邮电大学

版本：0.1

微言大义 高屋建瓴

# 前言

本书是在“2017-2019 中国科学院大学研究生暑期课程”和“2022 北京大学应用数学专题讲习班”的讲义基础上,结合多年在公钥加密方面的研究成果编写而成。目的是尽快引导读者到达公钥加密这一极重要的现代密码学领域,力求从高屋建瓴的视角介绍公钥加密的设计方法,对重要的思想剥丝抽茧、对关键的技术条分缕析。写作过程中根据教学和研究经历感悟对内容做了取舍和精简,参阅了国际顶级会议期刊的前沿论文,也融入了作者的独立思考。

计算机网络技术的飞速发展引发了人类社会组织形态的根本性变革,从集中式迁移为分布式,“海内存知己,天涯若比邻”从诗歌意象走进现实世界。面向分布式环境下的隐私保护需求,1976 年 Diffie 和 Hellman 开创了现代密码学的新方向—公钥密码学。迄今为止的半个多世纪以来,公钥密码学一直处于最活跃的前沿,引领驱动了密码学的研究进展,极大丰富了密码学的学科内涵。公钥加密作为公钥密码学最重要的分支,在理论方面孕育了可证明安全方法、将各类数学困难问题纳入工具库、启发了一系列密码原语和重要概念,已有多项历史性成果获得 Turing 奖和 Gödel 奖;在应用方面则是各类网络通信安全协议的核心组件,时刻保护着公开信道上消息传输的机密性。

近期,公钥加密仍处于快速发展阶段,在安全性方面,各类超越传统语义安全的高级安全属性研究已经日趋成熟,基于复杂性弱假设的细粒度安全的研究正在兴起;在功能性方面,函数加密的研究方兴未艾,全态加密的研究如火如荼。我们已经有幸见证了公钥加密之旅的美妙风景,但还有更广袤深邃的领域待探索征服。

公钥加密历经多年发展,各类方案层出不穷,相关概念定义繁多,因此想深入学习的读者往往会感觉陷入书山云海,难识庐山真面目。本书试图快速引导读者登高俯瞰,将公钥加密的设计方法尽收眼底,达到万变不离其宗的认知。为此,本书的内容偏重基于一般假设的通用构造。这样的选择有诸多好处,从理论角度,通用构造剥离了不必要的细节、凸显了核心要素,从而更容易洞察公钥加密的本质和复杂性下界;从实际角度,通用构造可以启发更多的具体构造,可根据安全或应用的需求灵活选择新的困难假设给出实例化方案。

本书的第一章简述了公钥加密的发展历程,第二章介绍了准备知识,为后续章节做好铺垫。第三章回顾经典的公钥加密方案,方便读者先获得具象的认识,预备一些重要的例子。第四章是核心部分,从各类密码组件出发发展了公钥加密的通用构造,并在最后获得更高阶的抽象,与对称加密的构造相互呼应、完美契合。第五章和第六章分别从安全性增强和功能性扩展两个维度介绍公钥加密的重要成果和前沿进展。第七章简介了公钥加密的标准化与工程实践,打通理论与实践的最后一公里。

书中有不少看似不起眼的注记,它们大多来源于作者科研过程中的心得体会,领悟其中蕴含的思辨方式之后或许能看到更美的风景。总的来说,切实掌握本书的内容之后,可使读者进入可证明安全密码学的新层次,为进一步的研究打好基础。

密码科学技术国家实验室对本书的出版给予了极大支持,作者深表感谢。作者也借此机会对在密码学研究给予作者热情支持帮助的上海交通大学的郁昱教授、刘胜利教授和山东大学的王美琴教授深表感谢。最后,感谢家人们的默默支持,没有你们的全情支持,我们不可能完成此书。

由于水平有限,时间紧迫,定有许多不当之处。诚恳欢迎批评指正。

陈宇 & 秦宝东

2023 年夏

# 目录

前言	i
前言	1
第一章 公钥加密的概述	2
1.1 背景与起源	3
1.2 发展历程简介	4
1.2.1 安全性增强	4
1.2.2 功能性丰富	4
第二章 准备知识	5
2.1 符号与记号	6
2.2 可证明安全方法	7
2.2.1 如何书写安全性证明	8
2.3 困难问题	11
2.3.1 整数分解类假设	11
2.3.2 离散对数类假设	13
2.3.3 格类假设	14
2.4 复杂性理论初步	17
2.5 信息论工具	19
2.5.1 熵的概念	19
2.5.2 随机性提取	20
2.6 公钥加密基本安全模型	21
2.6.1 公钥加密方案	21
2.6.1.1 基本性质	24
2.6.2 密钥封装机制	24
2.6.3 两类混合加密范式的比较	26
第三章 经典公钥加密方案回顾	27
3.1 基于数论问题的经典方案	28
3.1.1 Goldwasser-Micali PKE	28
3.1.2 Rabin PKE	29
3.2 基于离散对数类问题的经典方案	30
3.2.1 ElGamal PKE	30
3.2.2 Twisted ElGamal PKE	31
3.3 基于格问题的经典方案	33
3.3.1 Regev PKE	33
3.3.2 GPV PKE	34
第四章 通用构造方法	37
4.1 单向陷门函数类	38
4.1.1 单向陷门函数	38
4.1.2 有损陷门函数	41

4.1.3 相关积单向陷门函数	47
4.1.4 自适应单向陷门函数	49
4.2 哈希证明系统类	59
4.2.1 起源释疑	60
4.2.2 HPS 的构造	61
4.2.3 基于 HPS 构造 IND-CPA KEM	63
4.2.4 基于 HPS 构造 IND-CCA KEM	64
4.3 可提取哈希证明系统类	68
4.3.1 起源释疑	68
4.3.2 EHPS 的构造	69
4.3.3 基于 EHPS 构造 IND-CPA KEM	70
4.3.4 基于 EHPS 构造 IND-CCA KEM	72
4.3.5 小结	75
4.3.5.1 HPS 与 EHPS 的分析比较	75
4.4 程序混淆类	77
4.4.1 背景知识	77
4.4.2 受限伪随机函数	79
4.4.3 基于不可区分混淆的 KEM 构造	82
4.5 可公开求值伪随机函数类	86
4.5.1 定义与安全性	86
4.5.2 基于 PEPRF 的 KEM 构造	88
4.5.3 PEPRF 的构造	89
4.5.3.1 基于 DDH 假设的 PEPRF	89
4.5.3.2 基于 QR 假设的 PEPRF	89
4.5.3.3 基于 TDF 的 PEPRF	90
4.5.3.4 基于 HPS 的 PEPRF 构造	91
4.5.3.5 基于 EHPS 的 PEPRF 构造	92
4.5.3.6 基于 $i\mathcal{O}$ 的 PEPRF 构造	93
4.5.4 小结	94
<b>第五章 公钥加密的安全性增强</b>	<b>95</b>
5.1 抗泄漏安全	96
5.2 抗篡改安全	97
5.3 消息依赖密钥安全	98
<b>第六章 公钥加密的功能性扩展</b>	<b>99</b>
6.1 确定性公钥加密	100
6.2 可搜索公钥加密	101
<b>第七章 标准化及工程实践</b>	<b>102</b>
7.1 标准化与工程实践	103
7.1.1 公钥加密的标准化	103
7.1.1.1 国内外标准化组织简介	103
7.1.1.2 公钥加密标准方案	104
7.1.2 公钥加密的工程实践	105
7.1.2.1 重要方案的优秀开源实现	105

7.1.2.2 重要的开源密码库 . . . . .	105
7.1.3 密码学的工程实践经验 . . . . .	106
参考文献	107
后记	112

## 前言

# 第一章 公钥加密的概述

## 1.1 背景与起源

密码的历史甚至早于文字的出现. 密码的历史进程大概可以分为以下几个阶段:

- 古典阶段:
- 现代阶段: Shannon 和 DES
- Diffie-Hellman 的密码学新方向.



## 1.2 发展历程简介

目前的设想是从两个维度梳理公钥加密的发展历史: 功能性的丰富和安全性的增强.

自 Diffie 和 Hellman 的划时代论文 [DH76] 后, 公钥密码学的发展一日千里、日新月异, 热潮持续至今, 始终是现代密码学的核心和重要技术的摇篮.

公钥密码的发展大体可以分为两条主线: 一条是安全性的增强, 从最初的直觉安全演进到严格健壮的语义安全, 再到不可区分选择密文安全和各类超越传统安全模型的高级安全性, 如抗泄漏安全、抗篡改安全和消息依赖密钥安全; 另一条是功能性的丰富, 从最初的一对一解密到基于身份加密, 再到属性加密乃至极致泛化的函数加密.

### 1.2.1 安全性增强

最初的 RSA PKE 只满足朴素单向安全. Goldwasser 和 Micali 思考了公钥加密应该满足何种最低安全性这一问题, 提出了严格的安全模型, 设计了首个可证明安全的公钥加密方案 Goldwasser-Micali PKE.

### 1.2.2 功能性丰富

## 第二章 准备知识

## 2.1 符号与记号

对于正整数  $n$ , 用  $[n]$  表示集合  $\{1, \dots, n\}$ . 对于集合  $X$ ,  $|X|$  表示其大小,  $x \xleftarrow{R} X$  表示从  $X$  中均匀采样  $x$ ,  $U_X$  表示  $X$  上的均匀分布. 如果一个概率算法的运行时间是关于输入规模  $n$  的多项式函数  $\text{poly}(n)$ , 则称其是概率多项式时间的算法, 简记为 (probabilistic polynomial time, PPT). 令  $\mathcal{A}$  是一个随机算法,  $z \leftarrow \mathcal{A}(x; r)$  表示  $\mathcal{A}$  在输入为  $x$  和随机带为  $r$  时输出  $z$ , 当上下文文明确时, 常隐去随机带  $r$  简记为  $z \leftarrow \mathcal{A}(x)$ . 令  $f(\cdot)$  是关于  $n$  的函数, 如果对于任意的多项式  $p(\cdot)$  均存在常数  $c$  使得  $n > c$  时总有  $f(n) < 1/p(n)$  成立, 则称  $f$  是关于  $n$  的可忽略函数, 记为  $\text{negl}(n)$ . 在本书中,  $\kappa \in \mathbb{N}$  表示计算安全参数,  $\lambda \in \mathbb{N}$  表示统计安全参数. 令  $X = \{X_k\}_{k \in \mathbb{N}}$  和  $Y = \{Y_k\}_{k \in \mathbb{N}}$  是两个由  $k$  索引的分布簇, 如果  $X$  和  $Y$  之间的统计距离是关于  $\lambda$  的可忽略函数, 则称  $X$  和  $Y$  统计不可区分, 记为  $X \approx_s Y$ ; 如果任意 PPT 敌手区分  $X$  和  $Y$  的优势函数为  $\text{negl}(\kappa)$ , 则称  $X$  和  $Y$  计算不可区分, 记为  $X \approx_c Y$ .

对于实数  $x \in \mathbb{R}$ , 令  $\lfloor x \rfloor$  表示  $x$  的下取整,  $\lfloor x \rfloor := \lfloor x + 1/2 \rfloor$  表示与  $x$  最接近的整数.

令  $F$  是带密钥的函数,  $F_k(x)$  表示函数  $F$  在密钥  $k$  控制下对  $x$  的求值, 也常记作  $F(k, x)$ .

表 2.1: 缩略词及其含义对照表

缩略词	英文表达	中文含义
CPA	chosen-plaintext attack	选择明文攻击
CCA	chosen-ciphertext attack	选择密文攻击
PKE	public-key encryption	公钥加密方案
–	hardcore function	硬核函数
–	hardcore predicate	硬核谓词
–	oracle	谕言机
–	one-way trapdoor function	单向陷门函数

## 2.2 可证明安全方法

Talk is cheap. Show me the code.

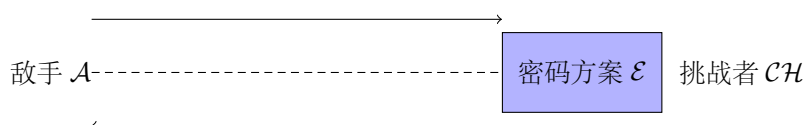
— Linus Torvalds

长久以来, 密码方案的安全性分析没有系统标准的方式, 传统方式由密码分析者通过尝试各类攻击来检验密码方案的安全性. 容易看出, 传统的分析方式存在以下局限: (1) 分析结果严重依赖分析者的个人能力 (如细致的观察和敏锐的直觉); (2) 分析者难以穷尽所有可能的攻击. 因此, 绝大多数古典密码陷入“设计—攻破—修补—攻破”的循环往复怪圈, 难以称之为真正的科学.

直到上世纪八十年代, Goldwasser 和 Micali [GM82] 借鉴计算复杂性理论的归约技术, 开创了可证明安全方法. 从此, 密码方案的安全性分析手段由反向攻击转为正向证明, 安全性由“声称安全”变为“可证安全”, 密码学也从此由艺术蝶变为真正的科学.

简言之, 可证明安全方法的核心是以下三要素组成:

- **精确的安全模型:** 通常由攻击者和挑战者之间的交互式游戏进行刻画, 如图 2.1 所示, 包括:
  - 敌手的计算能力: 常见的有概率多项式时间和指数时间
  - 敌手能够获取的信息, 包括:
    1. 固定信息: 如方案的公开参数等公开信息
    2. 非固定信息: 在攻击过程中获得的信息, 形式化为访问相应预言机获得的输出
  - 敌手的攻击效果: 以加密方案为例, 敌手恢复密钥和恢复明文是不同的攻击效果



$$\text{Adv}_A(\kappa) = |\Pr[S] - \Pr[T]|$$

图 2.1: 安全模型

令事件  $S$  表示敌手攻击成功这一事件,  $t$  表示某固定的基准优势 (如区分游戏定义为  $1/2$ , 搜索性游戏定义为  $0$ ), 定义  $A$  的优势函数为  $\text{Adv}_A(\kappa) = |\Pr[S] - t|$ , 其中  $S$  所在的概率空间由  $A$  和  $CH$  的随机带确定. 如果对于所有 PPT 敌手  $A$ ,  $\text{Adv}_A(\kappa)$  均是关于安全参数  $\kappa$  的可忽略函数, 则称密码方案  $E$  在既定的安全模型下是安全的.

- **清晰的困难性假设**
- **严格的归约式证明:** 通过反证式论证将方案的安全性归结到困难性假设.

图 2.2 是归约式证明的交换图表. 归约式证明的步骤如下:

1. 假设存在 PPT 的敌手  $A$  在既定安全模型下针对密码方案  $E$  具有不可忽略的优势  $\epsilon_1(\kappa)$ ;
2. 利用  $A$  的能力, 构建 PPT 的算法  $B$  以不可忽略的优势  $\epsilon_2(\kappa)$  打破困难问题. 这里  $R$  通常以扮演敌手  $A$  的挑战者的方式调用  $A$ , 因此也常称  $R$  是模拟算法或归约算法.  $R$  调用敌手  $A$  的方式又可细分为两类:
  - 黑盒方式:  $R$  以黑盒的方式调用  $A$ , 从算法的角度理解就是  $R$  以  $A$  作为子程序调用. 黑盒方式实质上证明了比所需更强的结果, 即  $\exists R \forall A$ . 此类证明最为常见, 被称为黑盒归约 (black-box reduction) 或者一致归约 (universal reduction).
  - 非黑盒方式:  $R$  以非黑盒的方式调用  $A$ , 充分利用了  $A$  的个体信息, 如算法结构、运行时间等. 这类证明是完全契合可证明安全思想的, 即  $\forall A \exists R$ . 此类证明相对少见, 被称为个体归约 (individual reduction) [Den17], 常可以突破黑盒归约下的安全性下界.

上述两步归约式论证的逻辑是: 构造出的算法  $R$  与困难假设相矛盾, 因此不存在算法  $R$ , 进而得出  $A$  不存在的结论.

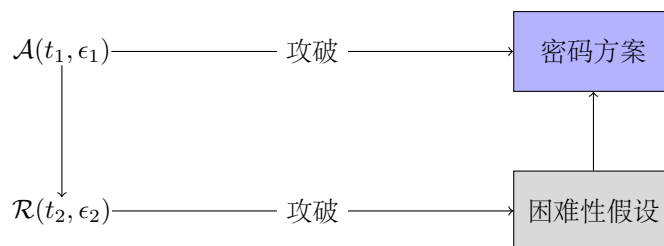


图 2.2: 归约式证明的交换图表

**安全性强弱.** 在明确可证明安全方法后, 密码方案的强弱可以根据三要素的强弱定性分析:

- 安全模型的强弱: 敌手的计算能力越强大、获得的信息越多、攻击的效果越弱, 则所确定的安全模型越强, 反之越弱.
- 困难假设的强弱: 通常搜索类假设强于判定类假设, 平均情形 (average-case) 假设强于最坏情形 (worst-case) 假设.
- 归约质量的优劣: 笼统的说,  $(t_2, \epsilon_2)$  越接近  $(t_1, \epsilon_1)$ , 归约的质量越高. 在归约算法的运行时间和优势函数两个指标中, 通常更关注后者. 定义归约松紧因子  $r = \epsilon_2/\epsilon_1$ , 如果  $r$  是一个常数, 称归约是紧的; 如果  $r$  是一个可察觉的函数 (noticeable function), 称归约多项式松弛的, 归约有效; 如果  $r$  是一个可忽略函数, 称归约超多项式松弛, 归约无效.

### 注记 2.1

阿基米德曾说过:“给我一个支点, 我能撬起地球!” 可证明安全方法与这句名言有共通之处, 地球可以理解为待证明方案的安全性, 支点和杠杆可以理解为归约式证明方法, 而施加在杠杆上的力可以理解为困难性假设. 如果支点在困难性假设和方案安全性正中, 代表归约最优, 方案的安全性可以紧归约到假设困难性上; 如果力臂过短, 则代表归约松弛, 困难性假设无法有意义的保证密码方案的安全性.

## 2.2.1 如何书写安全性证明

很多初学者对方案/协议的安全性有隐约的直觉, 但是很难写出严格精确的证明. 密码学中的安全性证明如同吉他中的大横按, 是横亘在所有初学者面前的一个障碍.

本小节将以极为精炼的方式归纳总结安全性证明的构建方式. 给出安全性证明大致有两种外在形式, 分别是单一归约和游戏序列.

单一归约适用于密码方案/协议仅依赖单一困难问题的简单形式. 拟基于惟一困难假设  $\mathcal{P}$  证明密码方案/协议

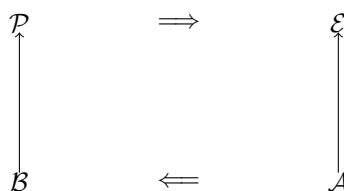


图 2.3: 单一归约

$\mathcal{E}$  的安全性时, 证明的方式是构建如图 2.3 所示的交换图表, 即首先假设存在 PPT 的敌手  $\mathcal{A}$  打破  $\mathcal{E}$  的安全性, 再利用  $\mathcal{A}$  构造算法  $\mathcal{B}$  打破困难假设  $\mathcal{P}$ . 构造  $\mathcal{B}$  的方法通常是令  $\mathcal{B}$  在方案  $\mathcal{E}$  的安全游戏中模拟挑战者的角色, 模拟的方式是将困难问题的实例直接嵌入到安全游戏的参数中. 此时, 基于  $\mathcal{P}$  的困难性便可得到任意 PPT 敌手针对  $\mathcal{E}$  的优势函数  $\text{Adv}_{\mathcal{A}}^{\mathcal{E}}(\kappa) \leq \text{negl}(\kappa)$  的结论.

### 注记 2.2

需要特别指出, 单一归约的适用范围有限, 仅适用于困难问题单一且能够直接嵌入安全游戏的场景, 这就要求安全游戏的目标和困难问题的类型必须相同, 比如同是计算性或者判定性. 有时, 在困难问题单一的情况下, 我们 also 需按照下面介绍的游戏序列方式组织证明. 如章节中的 ElGamal PKE 的证明. [add ref here](#)

对于基于多个困难问题的密码方案/协议, 即待证明的定理形如  $\mathcal{P}_1 + \dots + \mathcal{P}_n \Rightarrow \mathcal{E}$ , 就难以使用单一归约进行证明了. Shoup [Sho04] 针对该情形, 系统地提出了“游戏序列”的方式组织证明. 游戏序列的证明框架如下:

1. 引入一系列游戏, 记为  $\text{Game}_0, \dots, \text{Game}_m$ . 敌手在  $\text{Game}_i$  中成功的事件记作  $S_i$ , 优势基准为  $t$ , 则敌手在  $\text{Game}_i$  的优势函数为:

$$\text{Adv}_{\mathcal{A}}^{\text{Game}_i} = |\Pr[S_i] - t|$$

通常情况下  $\text{Game}_0$  刻画原始真实的安全游戏,  $\text{Game}_m$  刻画最终游戏且  $\text{Adv}_{\mathcal{A}}^{\text{Game}_m} = |\Pr[S_m] - t| = \text{negl}(\kappa)$ , 即敌手在  $\text{Game}_m$  中的优势函数是可忽略的.

2. 证明对于所有的  $i \in [m]$  均有  $|\Pr[S_i] - \Pr[S_{i-1}]| \leq \text{negl}(\kappa)$ ;
3. 通过混合论证 (hybrid argument) 得出  $\text{Adv}_{\mathcal{A}}^{\text{Game}_m} = |\Pr[S_0] - t| = \text{negl}(\kappa)$  的结论.

对于同一密码方案/协议, 在使用游戏序列进行证明时存在多种可能得游戏序列组织方式. 尽管在游戏序列的设定没有严格的规定, 但有以下两个经验准则:

- 相邻游戏的差异需最小化, 下一个游戏与上一个游戏仅有一个差异为宜;
- 差异应易于分析.

相邻游戏之间的差异通常有以下三种类型:

1. 差异源于不可区分的分布;
2. 差异基于某特定事件是否发生;
3. 差异仅是概念上调整, 为后续分析做铺垫.

对于第一类差异,  $\text{Game}_i$  和  $\text{Game}_{i+1}$  的变化可以归结为分布的不可区分性, 如  $Z_0 \approx Z_1$ , 其中  $Z_0$  和  $Z_1$  是两个分布. 换言之, 存在归约算法  $B$ , 以  $Z_0$  为输入时, 可以完美模拟敌手在  $\text{Game}_i$  中的视图; 以  $Z_1$  为输入时, 可以完美模拟敌手在  $\text{Game}_{i+1}$  中的视图. 我们令  $\text{View}_i$  表述敌手在  $\text{Game}_i$  中的视图. 在上下文清晰没有歧义时, 也常用  $\text{Game}_i$  直接代指敌手的视图.

- 当  $Z_0 \approx_s Z_1$  时, 利用复合引理 (composition lemma) 可以立刻得出任意敌手在两个游戏中的输出统计不可区分, 进而得出  $|\Pr[S_0] - \Pr[S_1]| \leq \text{negl}(\kappa)$
- 当  $Z_0 \approx_c Z_1$  时, 如果敌手成功这一事件  $B$  可准确判定, 则同样可以得出  $|\Pr[S_0] - \Pr[S_1]| \leq \text{negl}(\kappa)$ , 论证过程如下图 2.4 所示,  $B$  在事件  $S$  发生时输出 “1”, 否则输出 “0”. 根据游戏定义, 有  $\Pr[B(Z_0)] = \Pr[S_i]$ ,  $\Pr[B(Z_1)] = \Pr[S_{i+1}]$ , 因此有:

$$|\Pr[S_i] - \Pr[S_{i+1}]| = |\Pr[B(Z_0) = 1] - \Pr[B(Z_1) = 1]| \leq \text{negl}(\kappa)$$

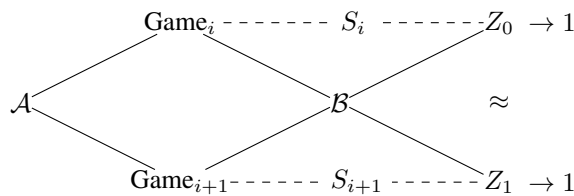


图 2.4: 基于分布不可区分的游戏演变

**注记 2.3**


许多学术论文在证明的过程中为了简便往往先证明敌手在两个相邻游戏中的视图不可区分, 再据此得出敌手优势函数的差是可忽略这一结论. 在多数情形, 这种论证并无问题, 但需要特别小心的是在视图计算不可区分时, 必须同时确保归约算法能够准确判定敌手成功事件才能够确保证明严谨. 在有些特殊情形, 归约算法无法有效判定敌手是否在游戏中成功, 此时归约算法无法利用敌手成功概率差异打破底层区分性假设, 从而导致归约失效. 请读者参考文献 [HLL16] 加深对该证明技术细节的理解和掌握.

第二类差异取决于某个特定事件是否发生, 即定义在同一概率空间的两个相邻游戏  $\text{Game}_i$  和  $\text{Game}_{i+1}$  仅在某特定事件  $F$  发生时存在差异, 在  $F$  不发生时完全一致, 概率描述如下:

$$S_i \wedge \neg F = S_{i+1} \wedge \neg F$$

为了分析敌手在相邻游戏  $\text{Game}_i$  和  $\text{Game}_{i+1}$  中的优势函数差, 需要以下的“差异引理”(difference lemma):

**引理 2.1 (Difference Lemma)**

令  $A, B, F$  是定义在同一概率空间中的事件, 如果  $A \wedge \neg F = B \wedge \neg F$ , 那么则有  $|\Pr[A] - \Pr[B]| \leq \Pr[F]$ . 

**证明** 差异引理的证明如下, 仅需使用古典概率中简单的缩放技巧:

$$\begin{aligned} |\Pr[A] - \Pr[B]| &= |\Pr[A \wedge F] + \Pr[A \wedge \neg F] - \Pr[B \wedge F] - \Pr[B \wedge \neg F]| // \text{全概率展开} \\ &= |\Pr[A \wedge F] - \Pr[B \wedge F]| // \text{化简} \\ &\leq \max\{\Pr[A \wedge F], \Pr[B \wedge F]\} \leq \Pr[F] // \text{缩放} \end{aligned}$$

证毕! □

根据差异引理, 若需证明  $|\Pr[S_i] - \Pr[S_{i+1}]| \leq \text{negl}(\kappa)$ , 仅需证明  $\Pr[F] \leq \text{negl}(\kappa)$ , 证明细分为以下两种情形:

- $F$  发生的概率取决于敌手的计算能力, 如敌手找到哈希函数的碰撞或者成功伪造消息认证码. 该情形需要建立安全归约, 即若  $F$  发生, 则存在敌手打破困难问题  $X_i$ .
- $F$  发生的概率与敌手的计算能力无关. 该情形仅需纯粹的信息论论证 (information-theoretic argument).

第三类差异称为桥接差异. 在分析游戏序列之间的差异时, 有时需要引入桥接步骤对某个变量的生成方式以等价的方式重新定义, 以确保差异分析的良好定义. 桥接步骤引入的差异仅是挑战者侧的概念性变化, 敌手侧的视图完全相同, 因此  $\Pr[S_i] = \Pr[S_{i+1}]$ . 桥接步骤看似可有可无, 实则必要, 若不引入必要的桥接步骤, 则会使得证明跳跃难以理解、游戏序列间的差异无法精确分析.

## 2.3 困难问题

密码学中常见的困难问题可大致分为数论类和格类, 其中前者是代数问题, 后者是数的几何问题, 这也正是格类困难问题具备抗量子攻击的原因之一.

数论类的假设又可进一步分为整数分解类和离散对数类两个分支. 本章首先介绍常见的数论类假设, 再介绍常见的格类困难问题.

### 2.3.1 整数分解类假设

整数分解类假设定义在群  $\mathbb{Z}_N^*$  上. 其中:

$$\mathbb{Z}_N^* \stackrel{\text{def}}{=} \{b \in \{1, \dots, N-1\} \mid \gcd(b, N) = 1\}$$

也即  $\mathbb{Z}_N^*$  是整数集合  $\{1, \dots, N-1\}$  中所有与  $N$  互质元素构成的子集, 在模乘运算  $ab \stackrel{\text{def}}{=} [ab \bmod N]$  下构成交换群.

以下令 **GenModulus** 是多项式时间算法, 其以安全参数  $1^\kappa$  为输入, 输出  $(N = pq, p, q)$ , 其中  $p$  和  $q$  (以压倒性概率) 是两个  $\kappa$ -bit 的素数.

#### 定义 2.1 (整数分解假设)

整数分解问题指分解大整数在平均意义下是困难的. 我们称整数分解假设相对于 **GenModulus** 成立当且仅当对于任意 PPT 敌手:

$$\Pr[\mathcal{A}(N) = (p', q') \text{ s.t. } p'q' = N] \leq \text{negl}(\kappa)$$

上述概率建立在敌手  $\mathcal{A}$  和  $\text{GenModulus}(1^\kappa) \rightarrow (N, p, q)$  的随机带上.



尽管整数分解假设历经百年的分析攻击仍然健壮, 但是它并不直接蕴含高效的密码系统. 这就引发密码科技工作者研究与整数分解问题困难性相关问题的研究. 1978 年, Rivest, Shamir 和 Adleman 提出了 RSA 问题.

令 **GenRSA** 是多项式时间算法, 其以安全参数  $1^\kappa$  为输入, 输出两个  $\kappa$ -bit 素数的乘积  $N$  作为模数, 同时输出正整数  $(e, d)$  满足  $ed \equiv 1 \pmod{\phi(N)}$ .

#### 定义 2.2 (RSA 假设)

RSA 问题指在平均意义下求解  $\mathbb{Z}_N^*$  的  $e$  次方根是困难的. 我们称 RSA 假设相对于 **GenModulus** 成立当且仅当对于任意 PPT 敌手:

$$\Pr[\mathcal{A}(N, e, y) = x \text{ s.t. } x^e = y \bmod N] \leq \text{negl}(\kappa)$$

上述概率建立在敌手  $\mathcal{A}$ 、 $\text{GenRSA}(1^\kappa) \rightarrow (N, e, d)$  和随机选取  $x \in \mathbb{Z}_N^*$  随机带上.



#### 注记 2.4

RSA 问题刻画的是在不知晓  $\phi(N)$  的情况下计算  $\mathbb{Z}_N^*$  中随机元素的  $e$  次方根是困难的. 容易看出, 如果敌手能够打破整数分解问题, 则可以通过分解  $N$  求出  $\phi(N)$ , 进而通过 Fermat 小定理计算  $e$  次方根. 因此, 整数分解问题难于 RSA 问题, 整数分解假设弱于 RSA 假设. 两个假设是否等价仍然未知.



给定群  $\mathbb{G}$ , 称  $y \in \mathbb{G}$  是一个二次剩余当且仅当  $\exists x \in \mathbb{G} \text{ s.t. } x^2 = y$ .  $x$  成为  $y$  的平方根. 如果一个元素不是二次剩余则称其为二次非剩余. 在 **abelian** 群中, 二次剩余构成子群.

首先考察群  $\mathbb{Z}_p^*$  中的二次剩余, 其中  $p$  是素数. 定义函数  $\text{sq}_p: \mathbb{Z}_p^* \rightarrow \mathbb{Z}_p^*$  为  $\text{sq}_p(x) \stackrel{\text{def}}{=} [x^2 \bmod p]$ . 当  $p$  是大于 2 的素数时,  $\text{sq}_p$  是 2-对-1 函数, 因此立刻可知  $\mathbb{Z}_p^*$  中恰好一半元素是二次剩余. 记模  $p$  的二次剩余集合为  $\mathcal{QR}_p$ , 模



$p$  的二次非剩余集合为  $\mathcal{QNR}_p$ , 我们有:

$$|\mathcal{QR}_p| = |\mathcal{QNR}_p| = \frac{|\mathbb{Z}_p^*|}{2} = \frac{p-1}{2}$$

定义元素  $x \in \mathbb{Z}_p^*$  模  $p$  的 Jacobi 符号如下:

$$\mathcal{J}_p(x) \stackrel{\text{def}}{=} \begin{cases} +1 & \text{if } x \in \mathcal{QR}_p \\ -1 & \text{if } x \in \mathcal{QNR}_p \end{cases}$$

再考察群  $\mathbb{Z}_N^*$  中的二次剩余, 其中  $N$  是两个互异素数  $p$  和  $q$  的乘积. 由中国剩余定理可知:  $\mathbb{Z}_N^* \simeq \mathbb{Z}_p^* \times \mathbb{Z}_q^*$ , 令  $y \mapsto (y_p, y_q)$  表示上述同构映射给出的分解, 易知  $y$  是模  $N$  的二次剩余当且仅当  $y_p$  和  $y_q$  分别是模  $p$  和模  $q$  的二次剩余. 定义函数  $\text{sq}_N : \mathbb{Z}_N^* \rightarrow \mathbb{Z}_N^*$  为  $\text{sq}_N(x) \stackrel{\text{def}}{=} [x^2 \bmod N]$ , 当  $N$  为互异素数乘积时,  $\text{sq}_N$  是 4-对-1 函数. 记模  $N$  的二次剩余集合为  $\mathcal{QR}_N$ , 由  $\mathcal{QR}_N$  与  $\mathcal{QR}_p \times \mathcal{QR}_q$  之间的一一对应关系可知:


$$\frac{|\mathcal{QR}_N|}{|\mathbb{Z}_N^*|} = \frac{|\mathcal{QR}_p| \cdot |\mathcal{QR}_q|}{|\mathbb{Z}_p^*|} = \frac{1}{4}$$

从二次剩余的角度可以对  $\mathbb{Z}_N^*$  中的元素进行如下的划分: (i)  $\mathbb{Z}_N^*$  可以划分为相同大小的  $\mathcal{J}_N^{+1}$  和  $\mathcal{J}_N^{-1}$  (Jacobi 符号分别为 1 和 -1); (ii)  $\mathcal{J}_N^{+1}$  有可以划分为  $\mathcal{QR}_N$  和  $\mathcal{QNR}_N^{+1}$ , 其中  $\mathcal{QNR}_N^{+1} \stackrel{\text{def}}{=} \{x \in \mathbb{Z}_N^* \mid x \notin \mathcal{QR}_N \wedge \mathcal{J}_N(x) = +1\}$ .

### 定义 2.3 (模二次剩余假设)

模二次剩余 (QR, quadratic residue) 假设指  $\mathcal{QR}_N$  上的均匀分布与  $\mathcal{QNR}_N^{+1}$  上的均匀分布计算不可区分. 我们称模二次剩余假设相对于 GenModulus 成立当且仅当对于任意 PPT 敌手:


$$|\Pr[\mathcal{A}(N, y_0) = 1] - \Pr[\mathcal{A}(N, y_1) = 1]| \leq \text{negl}(\kappa)$$

上述概率建立在敌手  $\mathcal{A}$ 、 $\text{GenModulus}(1^\kappa) \rightarrow (N, p, q)$  和随机选取  $y_0 \in \mathcal{QR}_N, y_1 \in \mathcal{QNR}_N^{+1}$  的随机带上. 

与 QR 假设应用紧密相关的技术细节是如何对  $\mathcal{QR}_N$  和  $\mathcal{QNR}_N^{+1}$  进行高效的均匀采样.

- 对  $\mathcal{QR}_N$  进行均匀采样较为简单: 仅需随机选取  $x \in \mathbb{Z}_N^*$  再令  $y := x^2 \bmod N$  即可. 注意到  $x^2 \bmod N$  是一个 4-to-1 的正则函数, 因此当  $x \xleftarrow{\mathcal{R}} \mathbb{Z}_N^*$  时, 输出  $y$  服从  $\mathcal{QR}_N$  上的均匀分布.
- 对  $\mathcal{QNR}_N^{+1}$  进行均匀采样稍显复杂, 当  $N$  的分解未知时如何均匀采样未知. 我们可以借助辅助信息  $z \in \mathcal{QNR}_N^{+1}$  完成采样, 即随机选取  $x \in \mathbb{Z}_N^*$ , 输出  $y := z \cdot x^2 \bmod N$ . 可以验证, 当  $x \xleftarrow{\mathcal{R}} \mathbb{Z}_N^*$  时, 输出  $y$  服从  $\mathcal{QNR}_N^{+1}$  上的均匀分布.


### 注记 2.5

显然, 整数分解问题难于二次剩余判定问题, 因此整数分解假设弱于模二次剩余判定假设. 两个假设是否等价仍然未知. 

### 定义 2.4 (模平方根假设)

模平方根 (SQR, square root) 假设指对  $\mathcal{QR}_N$  中的随机元素求平方根是困难的. 我们称模平方根假设相对于 GenModulus 成立当且仅当对于任意 PPT 敌手:

$$\Pr[\mathcal{A}(N, y) = x \text{ s.t. } x^2 = y \bmod N] \leq \text{negl}(\kappa)$$

上述概率建立在敌手  $\mathcal{A}$ 、 $\text{GenModulus}(1^\kappa) \rightarrow (N, p, q)$  和随机选取  $y \in \mathcal{QR}_N$  的随机带上. 

令  $p$  和  $q$  是两个互异的模 4 余 3 的素数, 则称  $N = pq$  是 Blum 整数. 我们有以下推论:

### 命题 2.1

当  $N$  是 Blum 整数时, 则每个模  $N$  的二次剩余有且仅有一个平方根是二次剩余.

上述推论保证了当  $N$  是 Blum 整数时, 函数  $f_N \stackrel{\text{def}}{=} [x^2 \bmod N]$  构成  $\mathcal{QR}_N$  上的置换. 这一性质在构造加密方案时至关重要.

### 注记 2.6

模平方根假设等价于整数分解假设, 即在未知  $N$  分解的情况下求模平方根与分解  $N$  一样困难.

综上, 整数分解类问题的困难性关系如图 2.5 所示:

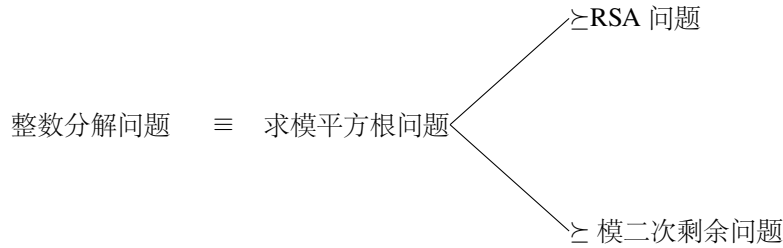


图 2.5: 整数分解类问题的困难性关系

## 2.3.2 离散对数类假设

离散对数类假设定义在循环群  $\mathbb{G}$  中. 令  $\text{GenGroup}$  是多项式时间算法, 其以安全参数  $1^\kappa$  为输入, 输出  $q$  阶循环群  $\mathbb{G} = \langle g \rangle$  的描述, 其中,  $q$  是  $\kappa$ -bit 的整数, 简记为  $(\mathbb{G}, q, g) \leftarrow \text{GenGroup}(1^\kappa)$ . 为了行文方便, 本书中假设  $\mathbb{G}$  为加法群, 用 “ $\cdot$ ” 表示群运算. 由循环群的定义可知,  $\mathbb{G}$  中的元素为  $\{g^0, g^1, \dots, g^{q-1}\}$ . 因此, 对于任意  $h \in \mathbb{G}$  存在唯一的  $x \in \mathbb{Z}_q$  使得  $g^x = h$ , 我们称  $x$  是  $h$  相对于生成元  $g$  的离散对数并记为  $x = \log_g h$ , 这里称其为离散对数强调其取值均为非负整数, 有别于标准算术对数的取值为实数.

### 定义 2.5 (离散对数假设)

离散对数问题指在平均意义下求解群元素的离散对数是困难的. 我们称离散对数 (DLOG) 假设相对于  $\text{GenGroup}$  成立当且仅当对于任意 PPT 敌手:

$$\Pr[\mathcal{A}(\mathbb{G}, q, g, h) = \log_g h] \leq \text{negl}(\kappa)$$

上述概率建立在敌手  $\mathcal{A}$ 、 $\text{GenGroup}(1^\kappa) \rightarrow (\mathbb{G}, q, g)$  和随机采样  $h \in \mathbb{G}$  的随机带上.

显然, 离散对数假设说明了  $x \mapsto g^x$  是从  $\mathbb{Z}_q$  到  $\mathbb{G}$  的单向函数. 单向函数能够蕴含的密码方案有限, 下面介绍与离散对数假设相关的其它假设, 它们能够作为更多密码方案的安全基础. 这类困难假设的起源于 Diffie 和 Hellman [DH76] 在 1976 年的划时代论文, 后来被称为 Diffie-Hellman 假设. 为了叙述方便, 我们首先定义 DH 函数  $\text{DH}_g : \mathbb{G}^2 \rightarrow \mathbb{G}$ ,

$$\text{DH}_g(h_1, h_2) \stackrel{\text{def}}{=} g^{\log_g h_1 \cdot \log_g h_2}$$

Diffie-Hellman 类假设可细分为两类, 一类是计算性 Diffie-Hellman (CDH) 问题, 一类是判定性 Diffie-Hellman (DDH) 问题. 下面依次介绍.

**定义 2.6 (CDH 假设)**

CDH 问题指在平均意义下计算  $\text{DH}_g$  函数是困难的. 我们称 *CDH* 假设相对于  $\text{GenGroup}$  成立当且仅当对于任意 PPT 敌手:

$$\Pr[\mathcal{A}(\mathbb{G}, q, g, g^a, g^b) = g^c] \leq \text{negl}(\kappa)$$

上述概率建立在敌手  $\mathcal{A}$ 、 $\text{GenGroup}(1^\kappa) \rightarrow (\mathbb{G}, q, g)$  和随机采样  $a, b \in \mathbb{Z}_q$  的随机带上.

**定义 2.7 (DDH 假设)**

对于四元组  $(g, g^a, g^b, g^c)$ , 如果  $g^c = \text{DH}_g(g^a, g^b)$  也即  $ab = c \bmod q$ , 则称其为 DH 元组. DDH 假设刻画的是随机 DH 元组和随机四元组是计算不可区分的. 我们称 *DDH* 假设相对于  $\text{GenGroup}$  成立当且仅当对于任意 PPT 敌手:

$$|\Pr[\mathcal{A}(\mathbb{G}, q, g, g^a, g^b, g^{ab}) = 1] - \Pr[\mathcal{A}(\mathbb{G}, q, g, g^a, g^b, g^c) = 1]| \leq \text{negl}(\kappa)$$

上述概率建立在敌手  $\mathcal{A}$ 、 $\text{GenGroup}(1^\kappa) \rightarrow (\mathbb{G}, q, g)$  和随机采样  $a, b, c \in \mathbb{Z}_q$  的随机带上.



离散对数类问题的困难性关系如图 2.6 所示:

$$\text{离散对数问题} \succeq \text{CDH 问题} \succeq \text{DDH 问题}$$

图 2.6: 离散对数类问题的困难性关系

**注记 2.7**

注意到任何  $q$  阶循环群均与  $\mathbb{Z}_q$  是同构的, 而  $\mathbb{Z}_q$  上的离散对数问题是容易的. 因此在实例化循环群  $\mathbb{G}$  必须谨慎审慎, 这也从一个方面说明离散对数类问题的困难性与底层代数结构的具体特性 (如群的表示) 紧密相关. 对于  $\mathbb{G}$  的实例化, 通常既可以选择  $\mathbb{F}_{p^k}^*$  的素数阶乘法子群, 也可以选择椭圆曲线上的素数阶乘法群. 另外强调一点, 存在这样的循环群  $\mathbb{G}$  (如双线性映射群) 使得离散对数、CDH 假设成立, 而 DDH 假设不成立.



### 2.3.3 格类假设

1997 年, Shor [Sho97] 给出了数论类问题 (包括整数分解类和离散对数类) 的有效量子算法. 在未来, 如果大规模量子计算机研制成功, 则数论类假设将不再成立. 迄今为止, 尚未有针对格基困难问题的有效量子算法, 通用的量子算法仅相对非量子算法有些许优势. 目前普遍的共识是格基困难问题具备抗量子安全能力, 这正是该类问题倍受关注的主要原因.

本小节中将介绍两个主要的平均意义下的格基困难问题, 短整数解问题和带误差学习问题. 需要提前说明的是, 格基困难问题的困难性与参数的选取密切相关, 因此格基问题的描述相比数论类问题要复杂得多.

Ajtai [Ajt96] 在 1996 年的开创性论文中正式提出了短整数解 (short integer solution, SIS) 问题. SIS 问题不仅可以作为所有 Minicrypt 世界中密码组件的安全基础, 包括单向函数、身份鉴别协议、数字签名, 还可以用来构造抗碰撞哈希函数. 非正式的, SIS 问题指在给定许多较大的有限加法群中随机选取的元素, 找到足够“短”的整系数组合使得其和是 0 是困难的. SIS 问题由以下参数刻画:

- 正整数  $n$  和  $q$ , 用于刻画加法群  $\mathbb{Z}_q^n$ ;
- 正实数  $\beta$ , 用于刻画解向量的长度;
- 正整数  $m$ , 用于表征群元素的个数.

其中  $n$  是主要的参数 (如:  $n \geq 100$ ),  $q > \beta$  通常设定为关于  $n$  的小多项式.

**定义 2.8 (短整数解假设 (SIS<sub>n,q,β,m</sub>))**

我们称 SIS 假设成立当且仅当对于任意 PPT 敌手:

$$\Pr[\mathcal{A}(\mathbf{a}_1, \dots, \mathbf{a}_m) = \mathbf{z} \neq \mathbf{0} \in \mathbb{Z}^m \text{ s.t. } \sum_{i=1}^m \mathbf{a}_i z_i = \mathbf{0} \in \mathbb{Z}_q^n \wedge \|\mathbf{z}\| \leq \beta] \leq \text{negl}(\kappa)$$

上述概率建立在敌手  $\mathcal{A}$  和随机选取  $\mathbf{a}_i \in \mathbb{Z}^m$  的随机带上。



以上定义中  $m$  个  $\mathbb{Z}_q^n$  上的随机向量可以按列向量的方式组成矩阵  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ . 因此, SIS 假设实质上在要求找到函数  $f_{\mathbf{A}}(\mathbf{z}) := \mathbf{A}\mathbf{z}$  的短整数非零向量原像是困难的.

下面简单讨论参数选取与问题困难性之间的关联:

- 如果不对  $\|\mathbf{z}\|$  进行限制, 那么可以轻易利用 Gaussian 消元法找到一个整数解. 同时, 我们必须要求  $\beta < q$ , 否则  $\mathbf{z} = (q, 0, \dots, 0) \in \mathbb{Z}^m$  即构成一个合法的非平凡解.
- 注意到任何关于矩阵  $\mathbf{A}$  的短整数解可通过补 0 平凡地延展为关于矩阵  $[\mathbf{A} \mid \mathbf{A}']$  的解. 换言之, SIS 问题的困难性随着  $m$  的增大变得容易. 对应的, SIS 问题的困难性随着  $n$  增加变得困难.
- 向量范数界  $\beta$  和向量  $\mathbf{a}_i$  的个数  $m$  必须足够大以保证解的存在性. 令  $\bar{m}$  是大于  $n \log q$  的最小正整数, 则我们必须有  $\beta > \sqrt{\bar{m}}$  和  $m \geq \bar{m}$ . 不失一般性, 不妨假设  $m = \bar{m}$ , 则存在超过  $q^n$  个向量  $\mathbf{x} \in \{0, 1\}^m$ , 根据鸽巢原理, 则必有  $\mathbf{x} \neq \mathbf{x}'$  使得  $\mathbf{A}\mathbf{x} = \mathbf{A}\mathbf{x}' \in \mathbb{Z}_q^n$ , 从而它们的差值  $\mathbf{z} = \mathbf{x} - \mathbf{x}' \in \{0, \pm 1\}^m$  是范数小于  $\beta$  的短整数解.
- 上述的鸽巢原理论证事实上蕴含更多深意: 函数族  $\{f_{\mathbf{A}} : \{0, 1\}^m \rightarrow \mathbb{Z}_q^n\}$  基于 SIS 假设是抗碰撞的. 若不然, 给定关于  $f_{\mathbf{A}}$  的一对碰撞  $\mathbf{x}, \mathbf{x}' \in \{0, 1\}^m$ , 则立刻诱导出关于  $\mathbf{A}$  的一个短整数解.

SIS 问题可以被理解为在以下特定  $q$  元  $m$  维整数格中的平均意义短向量问题 (short-vector problem, SVP), 该整数格的定义为:

$$\mathcal{L}^\perp(\mathbf{A}) \stackrel{\text{def}}{=} \{\mathbf{z} \in \mathbb{Z}^m : \mathbf{A}\mathbf{z} = \mathbf{0} \in \mathbb{Z}_q^n\} \supseteq q\mathbb{Z}^m$$

从编码的角度理解,  $\mathbf{A}$  扮演着格/码字  $\mathcal{L}^\perp(\mathbf{A})$  校验矩阵的角色. SIS 问题的困难性指对于随机选取的  $\mathbf{A}$ , 找到一个短的码字是困难的.

Regev [Reg05] 在 2005 年的开创性论文中提出了另一个平均意义下的重要格基困难问题—带误差学习问题 (learning with errors, LWE). LWE 问题与 SIS 问题互相对偶, 能够蕴含 Minicrypt 之外的密码体制.

在正式定义 LWE 问题之前, 首先引入 LWE 分布的概念. 称向量  $\mathbf{s} \in \mathbb{Z}_q^n$  为秘密, LWE 分布  $A_{\mathbf{s}, \chi}$  定义在  $\mathbb{Z}_q^n \times \mathbb{Z}_q$  上, 采样算法为随机选取  $\mathbf{a} \in \mathbb{Z}_q^n$ , 选取  $e \leftarrow \chi$ , 输出  $(\mathbf{a}, b = \langle \mathbf{s}, \mathbf{a} \rangle + e \bmod q)$ .

LWE 问题有两个版本, 其中搜索版本要求给定 LWE 采样求解秘密, 判定版本要求区分 LWE 采样和随机采样. LWE 问题由以下参数刻画:

- 正整数  $n$  和  $q$ , 和 SIS 问题一样, 用于刻画加法群  $\mathbb{Z}_q^n$ ;
- 正整数  $m$  表征采样的个数, 通常选取的足够大以保证秘密的惟一性;
- $\mathbb{Z}$  上的误差分布  $\chi$ , 通常的选取是宽度为  $\alpha q$  的离散 Gaussian 分布, 其中  $\alpha < 1$  称为相对错误率.

**定义 2.9 (搜索 LWE 假设)**

搜索 LWE 问题指给定  $m$  个  $A_{\mathbf{s}, \chi}$  的独立随机采样, 求解秘密向量  $\mathbf{s}$  是困难的. 我们称搜索 LWE 假设成立当且仅当对于任意 PPT 敌手:

$$\Pr[\mathcal{A}(\{\mathbf{a}_i, b\}_{i=1}^m \leftarrow A_{\mathbf{s}, \chi}) = \mathbf{s}] \leq \text{negl}(\kappa)$$

上述概率建立在敌手  $\mathcal{A}$ 、随机选取  $\mathbf{s} \in \mathbb{Z}_q^n$  和采样  $A_{\mathbf{s}, \chi}$  的随机带上。



**定义 2.10 (判定 LWE 假设)**

判定 LWE 问题指区分  $m$  个独立采样是来自  $A_{\mathbf{s}, \chi}$  分布还是随机分布是困难的. 我们称判定 LWE 假设成立当且仅当对于任意 PPT 敌手:

$$|\Pr[\mathcal{A}(\{\mathbf{a}_i, b\}_{i=1}^m \leftarrow A_{\mathbf{s}, \chi}) = 1] - \Pr[\mathcal{A}(\{\mathbf{a}_i, b\}_{i=1}^m \leftarrow U_{\mathbb{Z}_q^n \times \mathbb{Z}_q}) = 1]| \leq \text{negl}(\kappa)$$

上述概率建立在敌手  $\mathcal{A}$ 、随机选取  $\mathbf{s} \in \mathbb{Z}_q^n$  和采样  $A_{\mathbf{s}, \chi}$  以及  $U_{\mathbb{Z}_q^n \times \mathbb{Z}_q}$  的随机带上.

**注记 2.8**

LWE 问题时 LPN 问题 (learning parities with noise) 的一般化. 在 LPN 问题中,  $q = 2$ ,  $\chi$  为  $\{0, 1\}$  上的 Bernoulli 分布.



下面简单讨论参数选取与问题困难性之间的关联:

- 如果没有误差分布  $\chi$ , 则 LWE 问题的搜索版本和判定版本均可利用 Gaussian 消元法快速求解.
- 和 SIS 问题类似, 可以用矩阵的语言更简洁的描述 LWE 问题: (i) 将  $m$  个向量  $\mathbf{a}_i \in \mathbb{Z}_q^n$  汇聚为矩阵  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ ; (ii) 将  $m$  个  $b_i \in \mathbb{Z}_q$  汇聚为向量  $\mathbf{b} \in \mathbb{Z}_q^n$ , 因此对于 LWE 采样我们有:

$$\mathbf{b}^t = \mathbf{s}^t \mathbf{A} + \mathbf{e}^t (\text{mod } q),$$

其中  $\mathbf{e} \leftarrow \chi^m$ .

LWE 问题可以被理解为在以下特定  $q$  元  $m$  维整数格中的平均意义有界距离解码问题 (bounded-distance decoding problem, BDD), 该整数格的定义为:

$$\mathcal{L}(\mathbf{A}) \stackrel{\text{def}}{=} \{\mathbf{A}^t \mathbf{s} : \mathbf{s} \in \mathbb{Z}_q^n\} + q\mathbb{Z}^m$$

从编码的角度理解,  $\mathbf{A}$  扮演着格/码字  $\mathcal{L}(\mathbf{A})$  生成矩阵的角色. 对于 LWE 采样,  $\mathbf{b}$  与格中的惟一向量/码字相近, 搜索版本要求计算秘密向量  $\mathbf{s}$ , 即根据带误差的码字进行解码. 对于随机采样,  $\mathbf{b}$  以大概率远离格  $\mathcal{L}(\mathbf{A})$  中所有向量. SIS 问题的困难性指对于随机选取的  $\mathbf{A}$ , 找到一个短的码字是困难的.

## 2.4 复杂性理论初步

在复杂性理论中, 困难问  $P$  通常定义在  $L \subseteq X$  上,  $X$  是所有实例的集合,  $L$  是  $X$  中满足特定性质的一个子集. 我们称  $P$  是可高效判定/求解的, 如果存在确定性时间的 Turing 机  $M$  满足:

$$x \in L \iff M(x) = 1$$

所有可高效判定/求解问题的合集组成  $\mathcal{P}$  复杂性类.

**笔记** 实例集合  $X$  的学术术语是词 (words), 子集  $L$  对应的学术术语是语言 (language). 术语的来源是一个自然的类比: 不妨设世界上所有的词汇构成一个集合, 那么汉语、英语、法语、德语、C++ 语言、Rust 语言等多种多样的语言自然构成了这个集合的各个子集. 通常, 称语言内的元素为 Yes 实例, 语言外的元素为 No 实例.

密码学中的困难问题可以分为计算和判定两类:

- 计算类 (也称搜索类) 问题要求计算出问题的解: 如 RSA 问题、离散对数问题、计算 Diffie-Hellman 问题、短整数解问题等.
- 判定类问题要求判定是或否: 如二次剩余问题、判定 Diffie-Hellman 问题、判定 LWE 问题等.

从解空间的角度理解, 判定问题可以看做计算问题的特例, 即输出解为 1 比特. 通常, 同一个问题的计算版本难于判定版本, 对应的计算假设弱于判定假设.

困难的二元关系是对密码学中各种计算类困难问题的抽象.

### 定义 2.11 (二元关系 (binary relation))

令  $L \subseteq X$  是一个  $\mathcal{NP}$  语言.  $L$  由二元关系  $R_L : X \times W$  定义, 其中  $W$  之证据集合:

$$x \in L \iff \exists w \in W \text{ s.t. } (x, w) \in R_L$$

如果  $R_L$  满足如下两个性质, 则称其是困难的 (hard):

- 易采样 (easy to sample):  $\exists$  PPT 算法  $\text{SampR}$  对关系  $R_L$  进行随机采样, 其以公开参数  $pp$  为输入, 输出“实例-证据”元组  $(x, w) \in R_L$ .
- 难抽取 (hard to extract):  $\forall$  PPT 敌手  $\mathcal{A}$ :

$$\Pr[(x, \mathcal{A}(x) = w') \in R_L : (x, w) \leftarrow \text{SampR}(pp)] \leq \text{negl}(\kappa)$$

**笔记** 单向函数自然诱导了一个困难的二元关系. 易采样的性质由单向函数的定义域可高效采样和单向函数可高效求值两点保证, 难抽取的性质由单向函数的单向性保证.

**问题任务:** 计算/搜索      **解空间:**  $\{0, 1\}^{\text{poly}(\kappa)}$

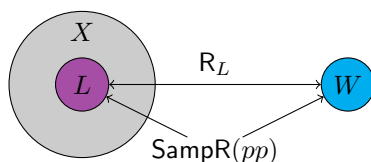


图 2.7: 计算类困难问题图示

子集成员判定问题 (SMP, Subset Membership Problem) 则是密码学中各种判定类问题的抽象.

**定义 2.12 (子集成员判定问题)**

令  $L \subset X$  是一个语言, 公开参数是  $pp$ . 定义以 3 个 PPT 采样算法:

- $\text{SampAll}(pp)$ : 输出  $X$  中的随机元素.
- $\text{SampYes}(pp)$ : 输出  $L$  中的随机元素, 即随机 Yes 实例.
- $\text{SampNo}(pp)$ : 输出  $X \setminus L$  中的随机元素, 即随机 No 实例.

SMP 问题有两种类型:

- Type 1:  $U_X \approx_c U_L$
- Type 2:  $U_{X \setminus L} \approx_c U_L$

**注记 2.9**

定义  $\rho = |L|/|X|$  为语言  $L$  相对于  $X$  的密度. 容易证明:

- 当  $\rho = \text{negl}(\kappa)$  时:  $\text{Type 1} \iff \text{Type 2}$
- 当  $\rho$  已知时:  $\text{Type 2} \Rightarrow \text{Type 1}$ 
  - 归约的方法是对给定分布和  $U_L$  分布根据  $\rho$  进行加权重构: 如果给定分布是  $U_{X \setminus L}$ , 则重构结果  $U_X$ ; 如果给定分布是  $U_L$ , 则重构结果仍为  $U_L$ . 因此, Type 2 的实例可以归约到 Type 1 的实例.



问题任务: 区分/判定      解空间:  $\{0, 1\}$

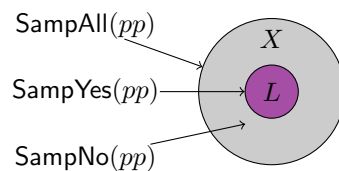


图 2.8: 判定类计算问题示例



## 2.5 信息论工具

### 2.5.1 熵的概念

Shannon 在 1948 年开创了信息论这一全新领域, 其工作的开创性不啻于 Einstein 关于引力场理论的工作, 一举奠定了学科的基础, 解答了最初所有最重要的问题。

Shannon 的信息论关注的重点是“消息”和它们在(有噪)信道中的传播. 容易直观理解的是, 消息所包含的信息量取决于消息令人感到意外的程度(如果消息平平无奇, 那么它包含的信息为 0). 相比而言, 稍显反直觉的是随机消息包含最多的信息。

信息论中最重要的概念是熵(entropy). 熵量化了期望的意义下消息包含的信息量, 单位通常是比特. 从概率论的角度看, 熵是对随机变量不确定性的测度. 以下令  $X$  是定义在  $\Omega$  上的随机变量。

#### 定义 2.13 (熵)

$X$  的熵刻画了平均意义下  $X$  取值的(不)可预测性:

$$H(X) = - \sum_{\omega \in \Omega} \Pr[X = \omega] \log \Pr[X = \omega]$$



#### 注记 2.10

一个消息的熵就是消息所包含信息的比特数, 用编码的语言刻画, 就是编码该消息所需的最短比特数.



密码方案/协议的安全性分析均是针对恶意敌手展开的. 敌手单次正确预测某随机变量(如私钥)值的概率与密码方案/协议的安全性紧密相关. 显然, 敌手的最佳策略是猜测最大似然值. 在本章中, 我们用大写字母  $X$  表示随机变量, 用小写字母  $x$  表示  $X$  的取值, 用花体字母  $\mathcal{X}$  表示  $X$  的支撑集。

一个随机变量  $X$  的最大可预测性是  $\max_{\omega \in \Omega} \Pr[X = \omega]$ . 最大可预测性对应最小熵(min-entropy), 严格定义如下:

#### 定义 2.14 (最小熵)

$X$  的最小熵刻画了  $X$  的最大可预测性:

$$H_{\infty}(X) = -\log \left( \max_{\omega \in \Omega} \Pr[X = \omega] \right)$$



#### 注记 2.11

最小熵可以看做“最坏情形”(worst-case) 的熵.



在很多场景中, 随机变量  $X$  与另一随机变量  $Y$  相关, 并且敌手知晓  $Y$  的取值. 因此, Dodis 等 [Dod+08] 引入了平均最小熵(average min-entropy) 来刻画  $X|Y$  的(不)可预测性:

$$\tilde{H}_{\infty}(X|Y) = -\log \left( \mathbb{E}_{y \leftarrow Y} \left[ 2^{H_{\infty}(X|Y=y)} \right] \right) = -\log \left( \mathbb{E}_{y \leftarrow Y} \left[ \max_{\omega \in \Omega} \Pr[X = \omega | Y = y] \right] \right) \quad (2.1)$$

以下浅释平均最小熵的定义直觉. 考虑一对变量  $X$  和  $Y$ (两者可能相关). 如果敌手知晓  $Y$  的取值  $y$ , 则  $X$  在敌手视角中的可预测性是  $\max_x \Pr[X = x | Y = y]$ . 在平均的意义下(对  $Y$  做期望), 敌手成功预测  $X$  的概率为  $\mathbb{E}_{y \leftarrow Y} [\max_x \Pr[X = x | Y = y]]$ .

平均最小熵的定义在对  $Y$  做加权平均的前提下( $Y$  的取值不受敌手控制)测度  $X$  最坏情形下的可预测性(敌手知晓  $y$  后对  $X$  的预测是恶意行为). 一个微妙的细节是平均最小熵的定义(2.1)先对预测成功的概率做期望后再取对数, 那能否交换  $\log$  和  $\mathbb{E}$  的次序呢? 定义平均最小熵  $\mathbb{E}_{y \leftarrow Y} [H_{\infty}(X|Y = y)]$  是否合理呢? 交换次序后的定义



失去了原本的意义. 考虑以下的例子, 令  $X$  和  $Y$  都是定义在  $\Omega = \{0, 1\}^{1000}$  上的随机变量,  $Y$  是  $\Omega$  上的随机分布, 当  $Y$  的取值  $y$  的首 bit 为 0 时,  $X$  的取值与  $y$  相同, 否则随机分布. 因此对于  $Y$  的半数取值  $y$ ,  $H_\infty(X|Y=y) = 0$ , 对另外半数取值,  $H_\infty(X|Y=y) = 1000$ , 所以  $\mathbb{E}_{y \leftarrow Y}[H_\infty(X|Y=y)] = 500$ . 然而, 声称  $X$  具有 500 比特的安全性显然不符合逻辑. 事实上, 知晓  $Y$  取值  $y$  的敌手直接输出  $y$ , 既能够以大于  $1/2$  的概率猜对  $X$  的取值. 平均最小熵标准的定义准确刻画了至少  $1/2$  的可预测性, 因为  $\tilde{H}_\infty(X|Y)$  略小于 1. 我们也可以从数学的角度解释如下,  $\mathbb{E}$  是线性算子, 而  $\log$  是非线性算子, 因此次序交换后意义不同.

平均最小熵和最小熵之间存在何种关系呢? Dodis 等 [Dod+08] 证明了如下的 cChaining Lemma, 建立了两者之间的关系, 给出了平均最小熵的一个下界.

### 引理 2.2 (Chaining Lemma)

令  $X, Y$  和  $Z$  是三个随机变量(可任意相关), 其中  $Y$  的支撑集包含至多  $2^r$  个元素. 我们有  $\tilde{H}_\infty(X|(Y, Z)) \geq H_\infty(X|Z) - r$ . 特别的, 当  $Z$  为空时, 上述不等式简化为:  $\tilde{H}_\infty(X|Y) \geq H_\infty(X) - r$ .



## 2.5.2 随机性提取

随机性是密码学的主旋律, 几乎所有已知密码方案/协议都离不开均匀随机采样. 然而, 均匀无偏的完美信源并不易得, 很多场景下存在的是有偏的弱信源. 如何在信源有偏的情况下进行均匀随机采样呢? 这就是随机性提取器所要完成的工作.

### 定义 2.15 (强随机性提取器)

令  $X$  是最小熵  $H_\infty(X) \geq n$  的随机变量,  $\text{ext} : \mathcal{X} \times \mathcal{S} \rightarrow \mathcal{Y}$  是一个可高效计算的函数. 我们称  $\text{ext}$  是对信源  $X$  的  $(n, \epsilon)$ -强随机性提取器当且仅当以下成立:

$$\Delta((\text{ext}(X, S), S), (Y, S)) \leq \epsilon,$$

其中  $S$  是定义在  $\mathcal{S}$  上的均匀随机变量,  $Y$  是定义在  $\mathcal{Y}$  上的均匀随机变量.



类比于平均最小熵和最小熵之间的关系, 当信源  $X$  与另一变量  $Z$  相关时, 我们需要引入平均强随机性提取器来对信源  $X$  进行萃取.

### 定义 2.16 (平均强随机性提取器)

令  $(X, Z)$  是满足约束  $\tilde{H}_\infty(X|Z) \geq n$  的任意变量对,  $\text{ext} : \mathcal{X} \times \mathcal{S} \rightarrow \mathcal{Y}$  是一个可高效计算的函数. 我们称  $\text{ext}$  是对信源  $X$  的平均意义  $(n, \epsilon)$ -强随机性提取器当且仅当以下成立:

$$\Delta((\text{ext}(X, S), S, Z), (Y, S, Z)) \leq \epsilon,$$

其中  $S$  是定义在  $\mathcal{S}$  上的均匀随机变量,  $Y$  是定义在  $\mathcal{Y}$  上的均匀随机变量.



Dodis 等 [Dod+08] 的 Leftover Hash Lemma(剩余哈希引理) 证明了任何强随机性提取性在适当的参数设定下都是平均强随机性提取器. 作为一个特例, Dodis 等证明了任何一族一致哈希函数 (universal hash functions) 都是平均强随机性提取器.

### 引理 2.3 (Leftover Hash Lemma)

令  $X$  和  $Z$  是满足约束  $\tilde{H}_\infty(X|Z) \geq n$  的任意变量对,  $\mathcal{H} = \{h_s : \mathcal{X} \rightarrow \mathcal{Y}\}_{s \leftarrow S}$  是一族一致哈希函数. 那么当  $n \geq \log |\mathcal{Y}| + 2 \log(1/\epsilon)$  时,  $\text{ext}(x, s) := h_s(x)$  是  $(n, \epsilon)$ -平均强随机性提取器.



## 2.6 公钥加密基本安全模型

### 2.6.1 公钥加密方案

公钥加密的概念由 Diffie 和 Hellman [DH76] 在 1976 年的划时代论文中正式提出, 其与对称加密的最大不同在于每个用户自主生成一对密钥, 公钥用于加密、私钥用于解密, 发送方仅需知晓接收方的公钥即可向接收方发送密文。

#### 定义 2.17 (公钥加密方案)

正式的, 公钥加密方案由以下的四个多项式时间组成:

- $\text{Setup}(1^\kappa)$ : 系统生成算法以安全参数  $1^\lambda$  为输入, 输出系统公开参数  $pp$ , 其中  $pp$  包含对公钥空间  $PK$ 、私钥空间  $SK$ 、明文空间  $M$  和密文空间  $C$  的描述. 该算法由可信第三方生成并公开, 系统中的所有用户共享, 所有算法均将  $pp$  作为输入. 当上下文明确时, 常常为了行文简洁省去  $pp$ .
- $\text{KeyGen}(pp)$ : 密钥生成算法以系统公开参数  $pp$  为输入, 输出一对公/私钥对  $(pk, sk)$ , 其中公钥公开, 私钥秘密保存.
- $\text{Encrypt}(pk, m; r)$ : 加密算法以公钥  $pk \in PK$ 、明文  $m \in M$  为输入, 输出密文  $c \in C$ .
- $\text{Decrypt}(sk, c)$ : 解密算法以私钥  $sk \in SK$  和密文  $c \in C$  为输入, 输出明文  $m \in M$  或者  $\perp$  表示密文非法. 解密算法通常为确定性算法.

**正确性.** 该性质保证公钥加密的功能性, 即使用私钥可以正确恢复出对应公钥加密的密文. 正式的, 对于任意明文  $m \in M$ , 有:

$$\Pr[\text{Decrypt}(sk, \text{Encrypt}(pk, m)) = m] = 1 - \text{negl}(\kappa). \quad (2.2)$$

公式 (2.2) 的概率建立在  $\text{Setup}(1^\kappa) \rightarrow pp$ 、 $\text{KeyGen}(pp) \rightarrow (pk, sk)$  和  $\text{Encrypt}(pk, m) \rightarrow c$  的随机带上. 如果上述概率严格等于 1, 则称公钥加密方案满足完美正确性.

#### 注记 2.12

通常基于数论假设的公钥加密方案满足完美正确性, 而格基方案由于底层困难问题的误差属性, 解密算法存在可忽略的误差.

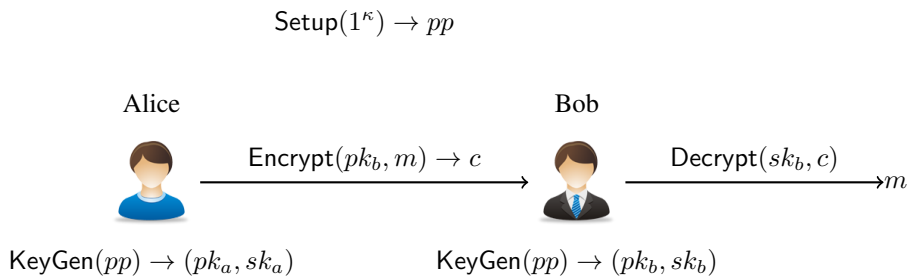


图 2.9: 公钥加密方案示意图


**安全性.** 定义公钥加密方案敌手  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$  的优势函数如下:

$$\text{Adv}_{\mathcal{A}}(\kappa) = \Pr \left[ \beta' = \beta : \begin{array}{l} pp \leftarrow \text{Setup}(1^\lambda); \\ (pk, sk) \leftarrow \text{KeyGen}(pp); \\ (m_0, m_1, \text{state}) \leftarrow \mathcal{A}_1^{\text{O}_{\text{decrypt}}(\cdot)}(pp, pk); \\ \beta \xleftarrow{\mathcal{R}} \{0, 1\}; \\ c^* \leftarrow \text{Encrypt}(pk, m_\beta); \\ \beta' \leftarrow \mathcal{A}_2^{\text{O}_{\text{decrypt}}(\cdot)}(pp, pk, \text{state}, c^*); \end{array} \right] - \frac{1}{2},$$

在上述定义中,  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$  表示敌手  $\mathcal{A}$  可划分为两个阶段, 划分界线是接收到挑战密文  $c^*$  前后,  $state$  表示  $\mathcal{A}_1$  向  $\mathcal{A}_2$  传递的信息, 记录部分攻击进展.  $\mathcal{O}_{\text{decrypt}}(\cdot)$  表示解密谕言机, 其在接收到密文  $c$  的询问后输出  $\text{Decrypt}(sk, c)$ . 如果任意的 PPT 敌手  $\mathcal{A}$  在上述游戏中的优势函数均为可忽略函数, 则称公钥加密方案是 IND-CPA 安全的; 如果任意的 PPT 敌手在阶段 1 可自适应访问  $\mathcal{O}_{\text{decrypt}}(\cdot)$  的情形下仍仅具有可忽略优势, 则称公钥加密方案是 IND-CCA1 安全的; 如果任意的 PPT 敌手在阶段 1 和阶段 2 均可自适应访问  $\mathcal{O}_{\text{decrypt}}(\cdot)$  的情形下仍仅具有可忽略优势, 则称公钥加密方案是 IND-CCA2 或 IND-CCA 安全的.

以下阐述公钥加密安全性定义的一些细微之处:

- 自适应的含义是敌手的攻击行为可根据学习到的知识动态调整, 如我们称敌手能够自适应的访问解密谕言机指敌手可以根据历史询问结果发起新的询问. 简而言之, 自适应性极大的增强了敌手的攻击能力.
- IND-CCA 安全性远强于 IND-CCA1 和 IND-CPA 安全性, 这是因为敌手可以在观察到挑战密文  $c^*$  后有针对性的发起更加有威胁的解密询问.
- $(m_0, m_1)$  由敌手任意选择, 从而巧妙精准的刻画了密文不泄漏明文任何一比特信息的直觉.
- 为了避免定义无意义, 在 IND-CCA 的安全游戏中禁止敌手在第二阶段向  $\mathcal{O}_{\text{decrypt}}(\cdot)$  询问挑战密文  $c^*$ .

 **笔记** 对于密码方案, 给出恰当的安全性定义非常重要: 一方面安全性定义必须足够强以刻画现实中存在的攻击, 另一方面安全性定义不能过强使得无法构造满足其的密码方案. 公钥加密的安全性定义是逐渐演化的.

上世纪 70 年代, Diffie 和 Hellman 提出了公钥加密的概念, 随后 Rivest、Shamir 和 Adleman 构造出了首个公钥加密方案——RSA 加密. 在这一阶段, 公钥加密的安全性仅具备符合直觉的单向性, 即在平均意义下从密文中恢复出明文是计算困难的. 到了上世纪 80 年代, 人们逐渐认识到单向性并不能满足应用需求, 这是因为对于单向安全的公钥加密方案, 敌手有可能从密文恢复出明文的部分信息, 而在应用中, 由于数据来源的多样性和不确定性, 明文的每一比特都可能包含关键的机密信息 (比如股票交易指令中的“买”或“卖”).

1982 年, Goldwasser 和 Micali [GM82] 指出单向安全的不足, 提出了语义安全性 (semantic security). 语义安全性的直观含义是密文对敌手求解明文没有帮助. 严格定义颇为精妙, 定义的形式是基于模拟的, 即敌手掌握密文的视角可以由一个 PPT 的模拟器在计算意义下模拟出来. 语义安全性可以看做 Shannon 完美安全性在计算意义的推广放松, 然而在论证的时候稍显笨重.

Goldwasser 和 Micali 给出了另一个等价的定义 (等价性的证明参见 Dodis 和 Ruhl 的短文 [DR99]), 即选择明文攻击下的不可区分性 (IND-CPA, indistinguishability against chosen-plaintext attack). IND-CPA 安全定义的直觉是密文在计算意义下不泄漏明文的任意一比特信息, 即对任意两个明文对应的密文分布是计算不可区分的, 其中选择明文攻击刻画了公钥公开特性使得任意敌手均可通过自行加密获得任意明文对应密文这一事实. 使用 IND-CPA 安全进行安全论证相比语义安全要便捷很多, 因此被广为采用.

注意到 IND-CPA 安全仅考虑被动敌手, 即敌手只窃听信道上的密文. 1990 年, Naor 和 Yung [NY90] 认为敌手有能力发起一系列主动攻击, 比如重放密文、修改密文等, 进而提出选择密文攻击 (CCA, chosen-ciphertext attack) 刻画这一系列主动攻击行为, 即敌手可以自适应的获取指定密文对应的明文. Naor 和 Yung 考虑了两种选择密文攻击, 一种是弱化版本, 称为午餐时间攻击 (lunch-time attack), 含义是敌手只能在极短的时间窗口 (收到挑战密文之前) 进行选择密文攻击; 另一种是标准版本, 敌手可以长时间窗口 (收到挑战密文前后) 进行选择密文攻击.

1998 年, Bleichenbacher [Ble98] 展示了针对 PKCS#1 标准中公钥加密方案的有效选择密文攻击, 实证了关于选择密文安全的研究并非杞人忧天. Shoup [Sho] 进一步深入探讨了选择密文安全的重要性与必要性. 从此, IND-CCA 安全成为了公钥加密方案的事实标准.

事实上, 公钥加密还存在另外一种更符合直觉的安全定义, 大意是对于明文空间中的随机明文进行加密, 敌手正确猜测出加密明文的概率与随机猜测的正确概率相近. 我们称这种安全性为消息恢复选择明文安全 (message-recovery-CPA security), 严格定义如下:

消息恢复选择明文安全. 令  $M$  是明文空间. 定义公钥加密方案敌手  $\mathcal{A}$  的优势函数如下:

$$\text{Adv}_{\mathcal{A}}(\lambda) = \Pr \left[ \begin{array}{l} pp \leftarrow \text{Setup}(1^\kappa); \\ (pk, sk) \leftarrow \text{KeyGen}(pp); \\ M' = \{m_1, \dots, m_n\} \subseteq M \leftarrow \mathcal{A}, |M'| \geq 2; \\ i \xleftarrow{\mathcal{R}} [n]; \\ c^* \leftarrow \text{Encrypt}(pk, m_i); \\ i' \leftarrow \mathcal{A}(pk, M', c^*); \end{array} \right] - \frac{1}{n}$$

如果任意的 PPT 敌手  $\mathcal{A}$  在上述游戏中的优势函数均为可忽略函数, 则称公钥加密方案是 Message-Recovery-CPA 安全的.

### 定理 2.1

公钥加密的 IND-CPA 与 Message-Recovery-CPA 安全性是等价的.

### 证明

Message-Recovery-CPA  $\Rightarrow$  IND-CPA. 容易看出, 当限定  $|M'| = 2$  时, MessageRecovery-CPA 蕴含 IND-CPA. 令  $\mathcal{A}$  是针对 IND-CPA 安全的敌手, 以下展示如何基于  $\mathcal{A}$  构造  $\mathcal{B}$  打破 MessageRecovery-CPA 安全.  $\mathcal{B}$  选择两个互异的明文  $m_0, m_1$ , 提交  $M' = \{m_0, m_1\}$ , 获得  $c^* \leftarrow \text{Encrypt}(pk, m_\beta)$  作为挑战.  $\mathcal{B}$  将  $c^*$  发送给  $\mathcal{A}$ , 并将  $\mathcal{A}$  的输出作为自己的输出发送给 MessageRecovery-CPA 的挑战者. 显然,  $\mathcal{B}$  完美的模拟了 IND-CPA 安全实验,  $\text{Adv}_{\mathcal{B}}(\kappa) = \text{Adv}_{\mathcal{A}}(\kappa)$ .

IND-CPA  $\Rightarrow$  Message-Recovery-CPA. 我们通过以下的游戏序列完成证明:

**Game 0.** 对应标准的 MessageRecovery-CPA 安全实验. 在挑战阶段,  $\mathcal{CH}$  在收到敌手选择的  $M'$  后, 随机选择  $i \xleftarrow{\mathcal{R}} [n]$ , 发送  $c^* \leftarrow \text{Encrypt}(pk, m_i)$  to  $\mathcal{A}$ . 根据定义, 我们有:

$$\text{Adv}_{\mathcal{A}}(\kappa) = |\Pr[S_0] - 1/|M'|||$$

**Game 1.** 与标准的 Message-Recovery-CPA 安全实验基本相同, 唯一的不同是在挑战阶段收到  $M'$  后,  $\mathcal{CH}$  随机选择  $i \xleftarrow{\mathcal{R}} [n], j \xleftarrow{\mathcal{R}} [n]$ , 发送  $c^* \leftarrow \text{Encrypt}(pk, m_j)$  to  $\mathcal{A}$ . 因为  $i$  在信息论意义下隐藏于  $\mathcal{A}$ , 因此我们有:

$$\Pr[S_1] = |1/|M'|||$$

此处需要特别强调上述概率仅建立在  $\mathcal{CH}$  随机选择  $i$  的随机带上.

最后, 我们证明 IND-CPA 安全保证了  $|\Pr[S_0] - \Pr[S_1]| = \text{negl}(\kappa)$ .

令  $\mathcal{B}$  是针对 IND-CPA 安全的敌手.  $\mathcal{B}$  随机选择  $i \xleftarrow{\mathcal{R}} [n], j \xleftarrow{\mathcal{R}} [n]$ , 令  $m_0 = m_i, m_1 = m_j$ , 将  $(m_0, m_1)$  提交给它的挑战者并获得关于  $m_\beta$  的挑战密文  $c^*$ .  $\mathcal{B}$  发送挑战密文  $c^*$  给  $\mathcal{A}$ ,  $\mathcal{A}$  输出对  $i$  的猜测  $i'$ . 如果  $i' = i$ ,  $\mathcal{B}$  输出 0. 否则,  $\mathcal{B}$  输出对  $\beta$  的随机猜测. 当  $\beta = 0$  时,  $\mathcal{B}$  向  $\mathcal{A}$  完美的模拟了 Game<sub>0</sub>, 因此正确输出  $\beta$  的概率是  $\Pr[S_0]$ ; 当  $\beta = 1$  时,  $\mathcal{B}$  向  $\mathcal{A}$  完美的模拟了 Game<sub>1</sub>, 因此正确输出  $\beta$  的概率是  $(1 - \Pr[S_1])$ ; 我们有:

$$\begin{aligned} \text{Adv}_{\mathcal{B}}(\kappa) &= |\Pr[\beta = 0] \Pr[S_0] + \Pr[\beta = 1](1 - \Pr[S_1]) - 1/2| \\ &= \left| \frac{1}{2}(\Pr[S_0] - \Pr[S_1]) \right| \end{aligned}$$

IND-CPA 的安全性保证了  $\text{Adv}_{\mathcal{B}}(\kappa)$  可忽略, 因此  $|\Pr[S_0] - \Pr[S_1]| \leq \text{negl}(\kappa)$ . 综上所述我们有  $\text{Adv}_{\mathcal{A}}(\kappa) = \text{negl}(\kappa)$ .

以上展示了 Message-Recovery-CPA 和 IND-CPA 的双向蕴含, 定理得证.

### 2.6.1.1 基本性质

**同态性.** 公钥加密方案的正确性隐式保证了解密算法自然诱导出从密文空间  $C$  到明文空间  $M$  的一个映射  $\phi = \text{Dec}(sk, \cdot)$ . 如果  $\phi$  具备同态性, 则第三方可对密文进行相应的公开计算, 得到的密文与对明文施加同样计算所得结果对应. 正式的, 令  $\mathcal{C} = \{f\}$  是从  $M^n \rightarrow M$  的某个电路族, 其中  $n$  是正整数;  $\text{Eval}$  为密文求值算法, 以公钥  $pk$ 、 $f \in \mathcal{C}$  和密文向量  $\mathbf{c} = (c_1, \dots, c_n)$  为输入, 输出  $c' \in C$ , 记作  $c' \leftarrow \text{Eval}(pk, f, \mathbf{c})$ . 如果对于任意  $f \in \mathcal{C}$  和任意明文  $\mathbf{m} = (m_1, \dots, m_n) \in M^n$  以下公式成立:


$$\Pr \left[ \begin{array}{l} (pk, sk) \leftarrow \text{KeyGen}(1^\kappa); \\ \text{Dec}(sk, c') = f(\mathbf{m}) : \mathbf{c} \leftarrow \text{Enc}(pk, \mathbf{m}); \\ c' \leftarrow \text{Eval}(pk, f, \mathbf{c}); \end{array} \right] = 1$$

则称公钥加密方案是  $\mathcal{C}$ -同态的,  $\mathcal{C}$  刻画了同态所支持的公开计算类型. 两种常见的同态类型如下:

- 部分同态 (partially homomorphic): 不失一般性, 若明文空间  $M$  为加法群, 密文空间  $C$  为乘法群, 若  $\mathcal{C}$  仅包含  $M^2 \rightarrow M$  的群运算, 则称加密方案是部分同态或者加法同态的. 此时同态性刻画如下:

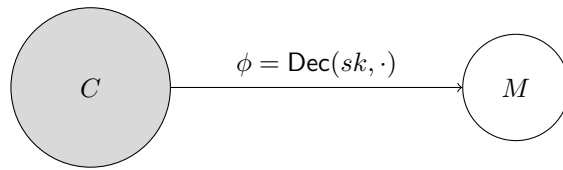
$$\Pr \left[ \text{Dec}(sk, c_1 \cdot c_2) = f(m_1, m_2) : \begin{array}{l} (pk, sk) \leftarrow \text{KeyGen}(1^\kappa); \\ c_1 \leftarrow \text{Enc}(pk, m_1), c_2 \leftarrow \text{Enc}(pk, m_2); \end{array} \right] = 1$$

- 全同态 (fully homomorphic): 若  $\mathcal{C}$  包含了  $M^n \rightarrow M$  的所有多项式时间可计算函数, 则称方案是全同态的.

 **笔记** 几乎所有公钥加密方案都构建在代数性质良好的结构上, 且大部分方案均天然满足部分同态, 如 RSA、ElGamal、Goldwasser-Micali、Benaloh、Paillier、Sander-Young-Yung、Boneh-Goh-Nissim、Ishai-Paskin 等. 在 RSA 公钥加密方案横空出世仅一年后, Rivest、Adleman 和 Dertouzos [RAD78] 即提出了全同态公钥加密的概念. 直到 31 年后, 才由 Gentry [Gen09] 通过引入理想格构造给出首个全同态加密方案的构造. 自此突破之后, 全同态加密迅猛发展, 理论成果百花齐放, 效率不断提升, 成为了隐私保护技术中重要且实用的密码学工具. 感兴趣的读者请参阅 Halevi 的综述文章 [Hal17].

#### 注记 2.13

对于密码方案和协议, 安全性和功能效率之间通常存在权衡关系 (trade-off). 对于公钥加密方案, CPA 安全性与同态性可以共存, 而更强的 CCA 安全性与同态性之间就存在冲突, 无法兼得. 在现实世界中应用公钥加密方案时, 需根据应用场景的具体需求在安全性和功能效率之间做出恰当的选择, 切不可教条.



### 2.6.2 密钥封装机制

主流的公钥加密方案基于数论或者格基困难问题构造. 基于数论问题的公钥加密方案因需要进行高精度算术运算导致加解密速率较低, 基于格基困难问题的公钥加密方案存在公钥和密文尺寸较大的问题. 而对称加密方案因其功能简单, 仅需异或等逻辑运算即可完成, 且硬件支持良好 (如定制的指令), 因此在较公钥加密具有较大的性能优势, 在加密长明文的场景下更为显著.

如何解决公钥加密在加密长消息时的性能短板呢? 解决思路是混合加密 (hybrid encryption), 朴素的实现方式是 PKE+SKE, 如图 2.10 所示:

1. 发送方首先随机选择对称密钥  $k$ , 调用公钥加密算法用接收方的公钥  $pk$  加密  $k$  得到  $c$ , 再调用对称加密算法用  $k$  加密明文  $m$  得到  $c_1$ , 最终的密文  $(c, c_1)$ .



2. 接收方在接收到密文  $(c, c')$  后, 首先使用私钥  $sk$  解密  $c$  恢复对称密钥  $k$ , 再使用  $k$  解密  $c'$ .

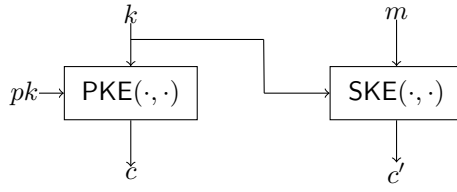


图 2.10: 混合加密: PKE+SKE

混合加密方法既保留了公钥加密的功能性, 同时性能几乎与对称加密相当, 因此是公钥加密加密长明文时的通用范式. Cramer 和 Shoup [CS02] 观察到公钥加密在混合加密范式中起到的关键作用是发送方向接收方传输对称密钥, 而传递的方式并非必须是加解密. 基于该观察, Cramer 和 Shoup 提出了“密钥封装-数据封装”范式, 简称为 KEM-DEM(key/data-encapsulation mechanism), 该范式可以看作是混合加密的另一种实现方式, 如图 2.11 所示. 顾名思义, KEM-DEM 范式包含 KEM 和 DEM 两个组件, DEM 可以粗略的等同为对称加密, KEM 是该范式的核心. 简言之, KEM 与 PKE 的不同在于发送方不再先显式选择对称密钥再加密, 而是封装一个随机的对称密钥.

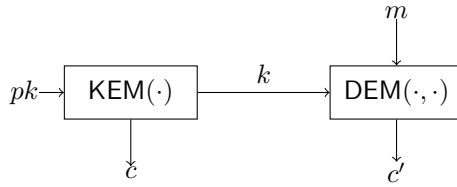


图 2.11: 混合加密: KEM+DEM

#### 定义 2.18 (密钥封装机制)

正式的, KEM 由以下的四个多项式时间组成:

- $\text{Setup}(1^\kappa)$ : 系统生成算法以安全参数  $1^\lambda$  为输入, 输出系统公开参数  $pp$ , 其中  $pp$  包含对公钥空间  $PK$ 、私钥空间  $SK$ 、对称密钥空间  $K$  和密文空间  $C$  的描述. 该算法由可信第三方生成并公开, 系统中的所有用户共享, 所有算法均将  $pp$  作为输入. 当上下文明确时, 常常为了行文简洁省去  $pp$ .
- $\text{KeyGen}(pp)$ : 密钥生成算法以系统公开参数  $pp$  为输入, 输出一对公/私钥对  $(pk, sk)$ , 其中公钥公开, 私钥秘密保存.
- $\text{Encaps}(pk; r)$ : 封装算法以公钥  $pk \in PK$  为输入, 输出对称密钥  $k \in K$  和封装密文  $c \in C$ .
- $\text{Decaps}(sk, c)$ : 解封装算法以私钥  $sk \in SK$  和密文  $c \in C$  为输入, 输出对称密钥  $k \in K$  或者  $\perp$  表示封装密文非法. 解封装算法通常为确定性算法.

#### 注记 2.14

在 KEM 中, 对称密钥  $k$  起到的作用是在发送方和接收方之间建立安全的会话信道, 因此也常称为会话密钥.

**正确性.** 该性质保证 KEM 的功能性, 即使用私钥可以正确恢复出封装密文所封装的会话密钥. 正式的, 对于任意会话密钥  $k \in K$ , 有:

$$\Pr[\text{Decaps}(sk, c) = k : (c, k) \leftarrow \text{Encaps}(pk)] = 1 - \text{negl}(\kappa). \quad (2.3)$$

公式 (2.3) 的概率建立在  $\text{Setup}(1^\kappa) \rightarrow pp$ 、 $\text{KeyGen}(pp) \rightarrow (pk, sk)$  和  $\text{Encaps}(pk) \rightarrow (c, k)$  的随机带上. 如果上述概率严格等于 1, 则称 KEM 方案满足完美正确性.

安全性. 定义 KEM 敌手  $\mathcal{A}$  的优势函数如下:

$$\text{Adv}_{\mathcal{A}}(\kappa) = \Pr \left[ \begin{array}{l} pp \leftarrow \text{Setup}(1^\lambda); \\ (pk, sk) \leftarrow \text{KeyGen}(pp); \\ (c^*, k_0^*) \leftarrow \text{Encaps}(pk), k_1^* \xleftarrow{R} K; \\ \beta \xleftarrow{R} \{0, 1\}; \\ \beta' \leftarrow \mathcal{A}^{\mathcal{O}_{\text{decaps}}(\cdot)}(pp, pk, c^*, k_\beta^*); \end{array} \right] - \frac{1}{2},$$

在上述定义中,  $\mathcal{O}_{\text{decaps}}(\cdot)$  表示解封封装谕言机, 其在接收到密文  $c$  的询问后输出  $\text{Decaps}(sk, c)$ . 如果任意的 PPT 敌手  $\mathcal{A}$  在上述游戏中的优势函数均为可忽略函数, 则称 KEM 方案是 IND-CPA 安全的; 如果任意的 PPT 敌手在可自适应访问  $\mathcal{O}_{\text{decaps}}(\cdot)$  的情形下仍仅具有可忽略优势, 则称 KEM 方案是 IND-CCA 安全的.

下面给出安全性定义的一些注记:

- KEM 的安全游戏中分阶段定义敌手不再必要, 因为挑战密文的生成不受敌手控制, 正是这点不同使得 KEM 的安全性定义要比 PKE 的安全性定义简单.
- 为了避免定义无意义, 在 IND-CCA 的安全游戏中禁止敌手向  $\mathcal{O}_{\text{decaps}}(\cdot)$  询问挑战密文  $c^*$ .

补充 DEM 部分和 KEM+DEM 的安全性组合

### 2.6.3 两类混合加密范式的比较

以上两种混合方法的共性都是首先生成对称密钥, 再利用对称密钥加密明文, 因此效率方面的差异体现在第一阶段. PKE+SKE 范式的非对称部分是先选择一个随机的密钥  $k$ , 再使用 PKE 对其加密得到  $c$ , 而 KEM+DEM 范式的非对称部分是两步并做一步完成. 如果使用 PKE-SKE 范式, 密文  $c$  必然存在密文扩张, 这是由概率加密的本质决定的; 而如果使用 KEM+DEM 的方法, 密文  $c$  相比  $k$  可能不存在扩张, 原因是此时  $c$  是对  $k$  的封装, 而非加密. 综上, 使用 KEM 代替 PKE, 不仅能够缩减整体密文尺寸, 也能够提升效率.


#### 注记 2.15

通常 KEM 要比 PKE 构造简单, 这是因为 KEM 可以看作功能受限的 PKE, 因为其只允许加密随机的明文. 

相比效率提升, KEM-DEM 的理论价值更大. 首先, KEM-DEM 范式实现了对 PKE 的功能解耦, 将 PKE 中的非对称内核抽取出来凝练为 KEM, 意义如下:

- KEM-DEM 范式极大简化了 PKE 的可证明安全. 我们只需证明 KEM 和 DEM 满足一定性质即可. 对比安全模型即可发现, 对于 PKE 有 CPA/CCA1/CCA 三个依次增强的安全性, 而 KEM 只有 CPA/CCA 两个依次增强的安全性. 最关键的是: 在 PKE 中敌手  $\mathcal{A}$  对挑战密文  $c^*$  有一定的控制能力, 而 KEM 中  $c^*$  完全由挑战者控制, 这一区别使得 KEM 安全证明中的归约算法更容易设计.
- KEM-DEM 范式有助于简化 PKE 的设计. 该范式将 PKE 的设计任务简化为对应的 KEM, 在后面的章节中可以看到, 在设计高等级安全的 PKE 时, 仅需设计满足相应安全性的 KEM 即可.
- KEM-DEM 范式有助于洞悉 PKE 本质. 该范式揭示了构造 PKE 的核心机制在于构造 KEM. 后续的章节揭示了 KEM 的本质是公开可求值的伪随机函数, 是伪随机函数在 minicrypt 中的对应. 认识到这一点后, 不仅可以几乎将所有公钥加密的构造统一在同一框架下, 还可以将 SKE 和 PKE 的构造在伪随机函数的视角下实现高度统一.

#### 注记 2.16

目前, 格基的 KEM 设计仍是 PKE-SKE 的方式, 显得不够灵巧纯粹, 如何设计精巧纯粹的格基 KEM 是很有挑战意义的研究课题. 

### 第三章 经典公钥加密方案回顾



## 3.1 基于数论问题的经典方案

### 3.1.1 Goldwasser-Micali PKE

Goldwasser 和 Micali [GM84] 在 1984 年基于 QR 假设构造出首个可证明安全的公钥加密方案. 该方案仅能加密一比特消息, 设计的思想可类比编码: 当明文为 0 时, 随机选取二次剩余元素作为密文; 当明文为 1 时, 随机选取 Jacobi 符号为 +1 的非二次剩余元素作为密文.

#### 构造 3.1 (Goldwasser-Micali PKE)

- $\text{Setup}(1^\kappa)$ : 全局参数生成  $pp$ , 包含对明文空间  $M = \{0, 1\}$  的描述.
- $\text{KeyGen}(pp)$ : 从  $pp$  中解析出  $\kappa$ , 运行  $\text{GenModulus}(1^\kappa) \rightarrow (N, p, q)$ , 随机选取  $z \in \mathcal{QR}_N^{+1}$ , 输出公钥  $pk = (N, z)$  和私钥  $sk = (p, q)$ .
- $\text{Encrypt}(pk, m)$ : 以公钥  $pk = (N, z)$  和明文  $m \in \{0, 1\}$  为输入, 随机选择  $x \xleftarrow{R} \mathbb{Z}_N^*$ , 输出密文  $c = z^m \cdot x^2 \bmod N$ .
- $\text{Decrypt}(sk, c)$ : 以私钥  $sk = (p, q)$  和密文  $c$  为输入, 利用私钥判定  $c$  是否是模  $N$  的二次剩余. 若是, 输出 0; 否则输出 1.

Goldwasser-Micali PKE 的正确性显然, 安全性由以下定理保证.

#### 定理 3.1

如果 QR 假设成立, 那么 Goldwasser-Micali PKE 是 IND-CPA 安全的.

**证明** 令  $S_i$  表示敌手在  $\text{Game}_i$  中成功概率. 以游戏序列的方式组织证明如下:

$\text{Game}_0$ : 该游戏是标准的 IND-CPA 游戏, 挑战者  $\mathcal{CH}$  和敌手  $\mathcal{A}$  交互如下:

- 初始化:  $\mathcal{CH}$  运行  $\text{Setup}(1^\kappa)$  生成公开参数  $pp$ , 同时运行  $\text{KeyGen}(pp)$  生成公私钥对  $pk = (N, z)$  和  $sk = (p, q)$ .  $\mathcal{CH}$  将  $(pp, pk)$  发送给  $\mathcal{A}$ .
- 挑战:  $\mathcal{A}$  选择  $m_0, m_1 \in \mathbb{G}$  并发送给  $\mathcal{CH}$ .  $\mathcal{CH}$  选择随机比特  $\beta \in \{0, 1\}$ , 随机选择  $x \in \mathbb{Z}_N^*$ , 计算  $c^* = z^{m_\beta} \cdot x^2 \bmod N$  并发送给  $\mathcal{A}$ .
- 猜测:  $\mathcal{A}$  输出对  $\beta$  的猜测  $\beta'$ .  $\mathcal{A}$  成功当且仅当  $\beta' = \beta$ .

根据定义, 我们有:

$$\text{Adv}_{\mathcal{A}}(\kappa) = |\Pr[S_0] - 1/2|$$

$\text{Game}_1$ : 与  $\text{Game}_0$  的唯一不同在于密钥对的生成方式,  $\mathcal{CH}$  将  $pk$  中元素  $z$  的选取由 Jacobi 符号为 +1 的随机非二次剩余元素切换为随机二次剩余元素. 在  $\text{Game}_1$  中, 无论  $m_\beta$  是 0 还是 1, 密文分布均是  $\mathcal{QR}_N$  上的均匀分布, 完美掩盖了  $\beta$  的信息. 因此, 即使对于拥有无穷计算能力的敌手, 我们也有:

$$\text{Adv}_{\mathcal{A}}(\kappa) = |\Pr[S_1] - 1/2| = 0$$

#### 引理 3.1

如果 QR 假设成立, 那么对于任意 PPT 敌手我们均有  $|\Pr[S_0] - \Pr[S_1]| \leq \text{negl}(\kappa)$ .

**证明** 证明的思路是反证. 若存在 PPT 敌手  $\mathcal{A}$  在  $\text{Game}_0$  和  $\text{Game}_1$  中成功的概率之差不可忽略, 则可构造出 PPT 算法  $\mathcal{B}$  打破 QR 困难问题. 令  $\mathcal{B}$  的 QR 挑战实例为  $(N, z)$ ,  $\mathcal{B}$  的目标是区分挑战实例  $z$  选自  $\mathcal{QR}_N^{+1}$  还是  $\mathcal{QR}_N$  上的均匀分布. 为此  $\mathcal{B}$  扮演 IND-CPA 游戏中的挑战者与  $\mathcal{A}$  交互如下:

- 初始化:  $\mathcal{B}$  根据它的挑战实例生成  $pp$ , 令  $pk = (N, z)$ , 将  $(pp, pk)$  发送给  $\mathcal{A}$ .
- 挑战:  $\mathcal{A}$  选择  $m_0, m_1 \in \mathbb{G}$  并发送给  $\mathcal{B}$ .  $\mathcal{B}$  随机选择  $\beta \xleftarrow{R} \{0, 1\}$ , 随机选取  $x \in \mathbb{Z}_N^*$  设置  $c^* = z^{m_\beta} \cdot x^2$  并发送给  $\mathcal{A}$ .

- 猜测:  $\mathcal{A}$  输出对  $\beta$  的猜测  $\beta'$ . 如果  $\beta' = \beta$ ,  $\mathcal{B}$  输出 1.

对上述交互分析可知, 如果  $z \xleftarrow{R} \mathcal{QNR}_N^{+1}$ , 那么  $\mathcal{B}$  完美的模拟了  $\text{Game}_0$ ; 如果  $z \xleftarrow{R} \mathcal{QR}_N$ , 那么  $\mathcal{B}$  完美的模拟了  $\text{Game}_1$ . 因此,  $\mathcal{B}$  解决 QR 挑战的优势  $\text{Adv}_{\mathcal{B}}^{\text{QR}}(\kappa) = |\Pr[S_0] - \Pr[S_1]|$ . 如果 QR 假设成立, 我们有  $|\Pr[S_0] - \Pr[S_1]| \leq \text{negl}(\kappa)$ .

综上, 定理得证. □

### 3.1.2 Rabin PKE

Rabin [Rab79] 在 1979 年基于 SQR 假设构造出  $\mathcal{QR}_N$  上的单向陷门置换  $f_N \stackrel{\text{def}}{=} [x^2 \bmod N]$ , 称为 Rabin TDP. 可以证明, 最低有效位 (lsb, least significant bit) 函数是 Rabin TDP 的 hardcore 谓词. 基于 Rabin TDP, 可以构造公钥加密方案如下:

#### 定义 3.1 (Rabin PKE)

- $\text{Setup}(1^\kappa)$ : 全局参数生成  $pp$ , 包含对明文空间  $M = \{0, 1\}$  的描述.
- $\text{KeyGen}(pp)$ : 从  $pp$  中解析出  $\kappa$ , 运行  $\text{GenModulus}(1^\kappa) \rightarrow (N, p, q)$ , 其中  $N$  是 Blum 整数. 输出公钥  $pk = N$  和私钥  $sk = (p, q)$ .
- $\text{Encrypt}(pk, m)$ : 以公钥  $pk = N$  和明文  $m \in \{0, 1\}$  为输入, 随机选择  $x \xleftarrow{R} \mathcal{QR}_N$ , 计算  $c_0 = x^2 \bmod N$ , 计算  $c_1 = m \oplus \text{lsb}(x)$ , 输出  $c = (c_0, c_1)$  作为密文.
- $\text{Decrypt}(sk, c)$ : 以私钥  $sk = (p, q)$  和密文  $c = (c_0, c_1)$  为输入, 计算  $x$  满足  $x^2 = c_0 \bmod N$ , 输出  $m' = c_1 \oplus \text{lsb}(x)$ .

Rabin PKE 的正确性由  $f_N \stackrel{\text{def}}{=} [x^2 \bmod N]$  是陷门置换这一事实保证, IND-CPA 安全性由陷门置换的单向性保证.

## 3.2 基于离散对数类问题的经典方案

### 3.2.1 ElGamal PKE

1985 年, ElGamal [ELG85] 基于 Diffie-Hellman 构造了 ElGamal PKE 方案. 该方案设计简洁精巧, 对后续的研究有深远的影响.

#### 定义 3.2 (ElGamal PKE)

- $\text{Setup}(1^\kappa)$ : 运行  $\text{GenGroup}(1^\kappa) \rightarrow (\mathbb{G}, q, g)$ , 输出  $pp$  包含循环群描述, 同时包含对公钥空间  $PK = \mathbb{G}$ 、私钥空间  $SK = \mathbb{Z}_q$ 、明文空间  $M = \mathbb{G}$  和密文空间  $C = \mathbb{G}^2$ .
- $\text{KeyGen}(pp)$ : 随机选取  $sk \in \mathbb{Z}_q$  作为私钥, 计算公钥  $pk := g^{sk}$ .
- $\text{Encrypt}(pk, m)$ : 以公钥  $pk$  和明文  $m \in \mathbb{G}$  为输入, 随机选择  $r \xleftarrow{\mathbb{R}} \mathbb{Z}_q$ , 计算  $c_0 = g^r, c_1 = pk^r \cdot m$ , 输出密文  $c = (c_1, c_2) \in C$ .
- $\text{Decrypt}(sk, c)$ : 以私钥  $sk$  和密文  $c = (c_0, c_1)$  为输入, 输出  $m' := c_1/c_0^{sk}$ .

**正确性.** 以下公式 3.1 说明方案具有完美正确性:

$$m' = c_1/c_0^{sk} = pk^r \cdot m / (g^r)^{sk} = m \quad (3.1)$$

#### 定理 3.2

如果 DDH 假设成立, 那么 ElGamal PKE 是 IND-CPA 安全的.

**证明** 令  $S_i$  表示敌手在  $\text{Game}_i$  中成功概率. 以游戏序列的方式组织证明如下:

$\text{Game}_0$ : 该游戏是标准的 IND-CPA 游戏, 挑战者  $\mathcal{CH}$  和敌手  $\mathcal{A}$  交互如下:

- 初始化:  $\mathcal{CH}$  运行  $\text{Setup}(1^\kappa)$  生成公开参数  $pp$ , 同时运行  $\text{KeyGen}(pp)$  生成公私钥对  $(pk, sk)$ .  $\mathcal{CH}$  将  $(pp, pk)$  发送给  $\mathcal{A}$ .
- 挑战:  $\mathcal{A}$  选择  $m_0, m_1 \in \mathbb{G}$  并发送给  $\mathcal{CH}$ .  $\mathcal{CH}$  选择随机比特  $\beta \in \{0, 1\}$ , 随机选择  $r \in \mathbb{Z}_q$ , 计算  $c^* = (g^r, pk^r \cdot m_\beta)$  并发送给  $\mathcal{A}$ .
- 猜测:  $\mathcal{A}$  输出对  $\beta$  的猜测  $\beta'$ .  $\mathcal{A}$  成功当且仅当  $\beta' = \beta$ .

根据定义, 我们有:

$$\text{Adv}_{\mathcal{A}}(\kappa) = |\Pr[S_0] - 1/2|$$

$\text{Game}_1$ : 与  $\text{Game}_0$  的唯一不同在于挑战密文的生成方式,  $\mathcal{CH}$  不再计算  $pk^r$  作为会话密钥掩蔽  $m_\beta$ , 而是随机选取  $z \xleftarrow{\mathbb{R}} \mathbb{Z}_q$ , 用  $g^z$  作为会话密钥掩蔽  $m_\beta$ . 挑战密文  $c^* = (g^r, g^z \cdot m_\beta)$ . 在  $\text{Game}_1$  中, 由于  $r$  和  $z$  均有挑战者从  $\mathbb{Z}_q$  中独立随机选取, 因此挑战密文  $c^*$  在  $\mathbb{G} \times \mathbb{G}$  上均匀分布, 完美掩盖了  $\beta$  的信息. 因此, 即使对于拥有无穷计算能力的敌手, 我们也有:

$$\text{Adv}_{\mathcal{A}}(\kappa) = |\Pr[S_1] - 1/2| = 0$$

#### 引理 3.2

如果 DDH 假设成立, 那么对于任意 PPT 敌手我们均有  $|\Pr[S_0] - \Pr[S_1]| \leq \text{negl}(\kappa)$ .

**证明** 证明的思路是反证. 若存在 PPT 敌手  $\mathcal{A}$  在  $\text{Game}_0$  和  $\text{Game}_1$  中成功的概率差不可忽略, 则可构造出 PPT 算法  $\mathcal{B}$  打破 DDH 困难问题. 令  $\mathcal{B}$  的 DDH 挑战实例为  $(g, g^a, g^b, g^c)$ ,  $\mathcal{B}$  的目标是区分挑战实例是 DDH 四元组还是随机四元组. 为此  $\mathcal{B}$  扮演 IND-CPA 游戏中的挑战者与  $\mathcal{A}$  交互如下:

- 初始化:  $\mathcal{B}$  根据它的挑战实例生成  $pp$ , 令  $pk = g^a$ , 将  $(pp, pk)$  发送给  $\mathcal{A}$ . 注意,  $\mathcal{B}$  并不知晓  $a$  (这是符合逻辑的, 不然归约无意义).

- 挑战:  $\mathcal{A}$  选择  $m_0, m_1 \in \mathbb{G}$  并发送给  $\mathcal{B}$ .  $\mathcal{B}$  随机选择  $\beta \xleftarrow{\mathbb{R}} \{0, 1\}$ , 设置  $c^* = (g^b, g^c \cdot m_\beta)$  并发送给  $\mathcal{A}$ . 该设定隐式的设定  $r = b$ .
- 猜测:  $\mathcal{A}$  输出对  $\beta$  的猜测  $\beta'$ . 如果  $\beta' = \beta$ ,  $\mathcal{B}$  输出 1.

对上述交互分析可知, 如果  $c = ab$ , 那么  $\mathcal{B}$  完美的模拟了  $\text{Game}_0$ ; 如果  $c$  在  $\mathbb{Z}_q$  中随机选择, 那么  $\mathcal{B}$  完美的模拟了  $\text{Game}_1$ . 因此,  $\mathcal{B}$  解决 DDH 挑战的优势  $\text{Adv}_{\mathcal{B}}^{\text{DDH}}(\kappa) = |\Pr[S_0] - \Pr[S_1]|$ . 如果 DDH 假设成立, 我们有  $|\Pr[S_0] - \Pr[S_1]| \leq \text{negl}(\kappa)$ .

综上, 定理得证.  $\square$

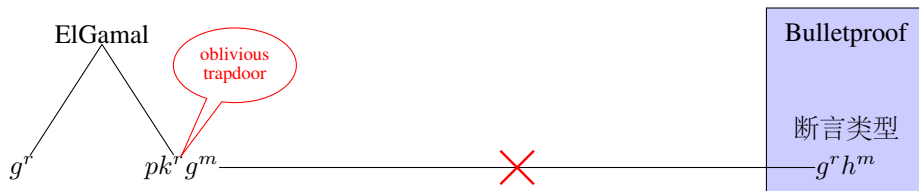


**笔记** [具有实际应用价值的同态] ElGamal PKE 构建在  $q$  阶循环群  $\mathbb{G} = \langle g \rangle$  上, 明文空间是  $\mathbb{G}$ , 使用公钥  $pk$  对明文  $m$  的加密所得密文为  $(g^r, pk^r \cdot m)$ . 容易验证, ElGamal PKE 相对于  $\mathbb{G}$  中的群运算 “ $\cdot$ ” 同态, 然而, 这种类型的同态并无实际意义, 现实应用中需要的是相对于  $\mathbb{Z}_q$  上的模加运算 “ $+$ ” 同态. 面向实际需求, ISO/IEC 标准化了 exponential ElGamal PKE 方案. 该方案同样构建在  $q$  阶循环群  $\mathbb{G} = \langle g \rangle$  上, 所不同的是明文空间设定为  $\mathbb{G}$  的自然同构  $\mathbb{Z}_q$ , 使用公钥  $pk$  对明文  $m$  加密时, 首先计算  $m$  的自然同构映射结果  $g^m$ , 再如常加密, 最终密文为  $(g^r, pk^r \cdot g^m)$ . 容易验证, exponential ElGamal 相对于  $\mathbb{Z}_q$  中的 “ $+$ ” 运算同态.

### 3.2.2 Twisted ElGamal PKE

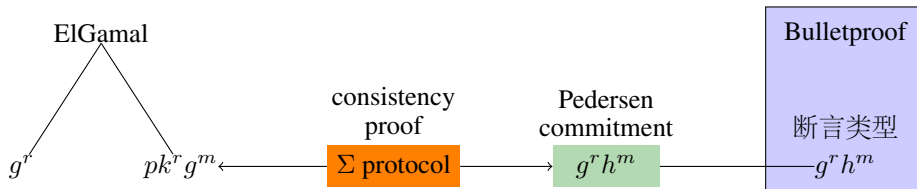
近半个世纪, 随着网络技术的飞速发展, 计算模式逐渐由集中式迁移分布式. 新型计算模式对加密方案的需求也从单一的机密性保护扩展到对隐私计算的支持. 上一节注记中提到的 Exponential ElGamal PKE 支持  $\mathbb{Z}_p$  上的模加运算 “ $+$ ” 同态, 适用于密态计算场景. 在区块链和机器学习等涉及恶意敌手的计算场景中, 还需要加法同态加密方案具有零知识证明友好的特性, 即易与零知识证明协议进行套接, 证明密文加密的明文满足声称的约束关系, 如在指定的区间内.

当前最高效的零知识区间范围证明系统是构建在离散对数群上的 Bulletproof [Bün+18], 其接受的断言类型为 Pedersen 承诺. 尽管 exponential ElGamal PKE 密文的第二项  $pk^r \cdot g^m$  也是 Pedersen 承诺的形式, 但是若证明者为密文发送方, 则其知晓承诺密钥  $(pk, g)$  之间的离散对数关系, 从而无法调用 Bulletproof 完成证明 (合理性得不到保证).

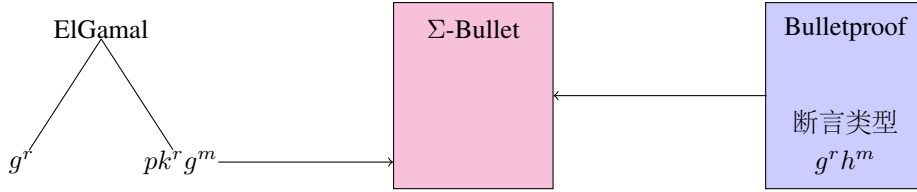


解决该问题有两种技术手段:

- 文献 [Fau+19] 中的方法: 证明者引入新的 Pedersen 承诺作为桥接, 并设计  $\Sigma$  协议证明新承诺的消息与明文的一致性, 再调用 Bulletproof 对桥接承诺进行证明. 该方法的缺点是需要引入桥接承诺的额外的  $\Sigma$  协议, 增大了证明和验证的开销.



- 文献 [Bün+20] 中的方法: 结合待证明的 ElGamal PKE 密文对 Bulletproof 进行重新设计, 使用特制的  $\Sigma$ -Bulletproof 完成证明. 该方法的缺点是需要对 Bulletproof 进行定制化的改动, 不具备模块化.



上述两种技术手段均存在不足. 针对这一问题, Chen 等 [Che+20] 对 exponential ElGamal PKE 进行变形扭转, 将封装密文与会话密钥的位置对调, 同时更改同构映射编码的基底, 得到 twisted ElGamal PKE. 新的加密方案与 exponential ElGamal PKE 的性能和安全性相当, 同样满足  $\mathbb{Z}_q$  上的模加同态; 特别的, 密文的第二部分恰好是标准的 Pedersen 承诺形态 (承诺密钥陷门未知), 因此可无缝对接 Bulletproof 等区间范围证明, 具备零知识证明友好的特性.

### 定义 3.3 (Twisted ElGamal PKE)

- $\text{Setup}(1^\kappa)$ : 运行  $\text{GenGroup}(1^\kappa) \rightarrow (\mathbb{G}, q, g)$ , 随机选取  $\mathbb{G}$  的另一生成元  $h$ , 输出  $pp$  包含循环群和  $h$  的描述, 同时包含对公钥空间  $PK = \mathbb{G}$ 、私钥空间  $SK = \mathbb{Z}_q$ 、明文空间  $M = \mathbb{Z}_q$  和密文空间  $C = \mathbb{G}^2$ .
- $\text{KeyGen}(pp)$ : 随机选取  $sk \in \mathbb{Z}_q$  作为私钥, 计算公钥  $pk := g^{sk}$ .
- $\text{Encrypt}(pk, m)$ : 以公钥  $pk$  和明文  $m \in \mathbb{Z}_q$  为输入, 随机选择  $r \xleftarrow{\mathbb{R}} \mathbb{Z}_q$ , 计算  $c_0 = pk^r$ ,  $c_1 = pk^r \cdot h^m$ , 输出密文  $c = (c_0, c_1) \in C$ .
- $\text{Decrypt}(sk, c)$ : 以私钥  $sk$  和密文  $c = (c_0, c_1)$  为输入, 输出  $m' := \log_h c_1 / c_0^{sk^{-1}}$ .



**正确性.** 以下公式 3.2 说明方案具有完美正确性:

$$c_1 / c_0^{sk^{-1}} = g^r \cdot h^m / (pk^r)^{sk^{-1}} = h^m \quad (3.2)$$

### 定理 3.3

如果 DDH 假设成立, 那么 twisted ElGamal PKE 是 IND-CPA 安全的.

证明与标准的 ElGamal PKE 证明类似, 我们留给作者作为练习.



**笔记** 为获得  $\mathbb{Z}_q$  上的加法同态, exponential ElGamal PKE 和 twisted ElGamal PKE 均将明文空间设定为  $\mathbb{Z}_q$ , 加密时必须先进行同构编码, 解密时则在最后需要进行解码. 解码的过程等同于求解离散对数, 因此为了确保解密高效, 必须将明文空间限制在较小的范围内, 如  $[2^{40}]$ .

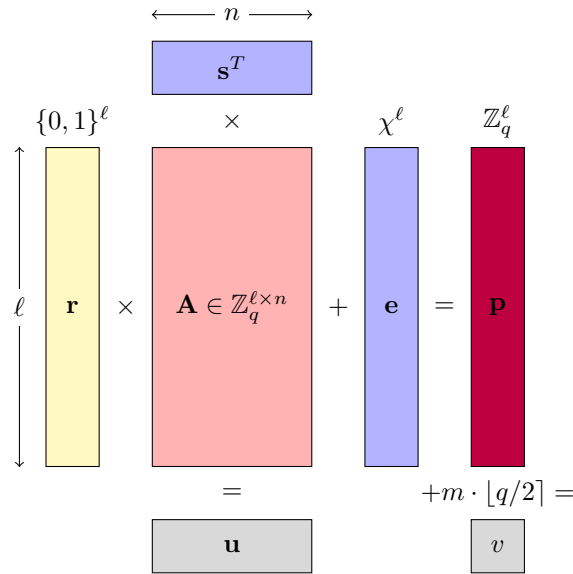


图 3.1: Regev PKE 加密方案示意图

### 3.3 基于格问题的经典方案

#### 3.3.1 Regev PKE

Regev [Reg05] 中提出了 LWE 困难问题, 并基于该问题构造了一个公钥加密方案, 称为 Regev PKE.

##### 定义 3.4 (Regev PKE)

- **Setup**( $1^\kappa$ ): 以安全参数  $\kappa$  为输入, 生成随机矩阵  $\mathbf{A} \in \mathbb{Z}_q^{\ell \times n}$  作为公开参数.
- **KeyGen**( $pp$ ): 以公开参数  $pp$  为输入, 随机选取向量  $\mathbf{s} \xleftarrow{\mathbb{R}} \mathbb{Z}_q^n$  作为私钥, 随机选取噪声向量  $\mathbf{e} \xleftarrow{\mathbb{R}} \chi^\ell$  (其中  $\chi^\ell = D_{\mathbb{Z}^\ell, r}$ ), 计算  $\mathbf{p} \leftarrow \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \in \mathbb{Z}_q^\ell$  作为公钥.
- **Encrypt**( $pk, m$ ): 以公钥  $pk = \mathbf{p}$  和明文  $m \in \{0, 1\}$  为输入, 随机选取向量  $\mathbf{r} \xleftarrow{\mathbb{R}} \{0, 1\}^\ell$  计算  $\mathbf{u}^T = \mathbf{r}^T \mathbf{A}$ ,  $v = \mathbf{r}^T \mathbf{p} + m \cdot \lfloor q/2 \rfloor$  作为密文.
- **Decrypt**( $\mathbf{s}, c$ ): 以私钥  $sk = \mathbf{s}$  和密文  $c = (\mathbf{u}, v)$ , 计算  $y = v - \mathbf{u}^T \mathbf{s} \in \mathbb{Z}_q$ , 若  $y$  接近 0 则输出 0, 若  $y$  接近  $\lfloor q/2 \rfloor$  则输出 1.



**正确性.** 观察以下等式:

$$\begin{aligned}
 y &= v - \mathbf{u}^T \mathbf{s} \\
 &= \mathbf{r}^T \mathbf{p} + m \cdot \lfloor q/2 \rfloor - \mathbf{r}^T \mathbf{A} \mathbf{s} \\
 &= \mathbf{r}^T (\mathbf{A} \mathbf{s} + \mathbf{e}) + m \cdot \lfloor q/2 \rfloor - \mathbf{r}^T \mathbf{A} \mathbf{s} \\
 &= \mathbf{r}^T \mathbf{e} + m \cdot \lfloor q/2 \rfloor
 \end{aligned}$$

由上述推导可知, 当累计误差  $|\langle \mathbf{r}, \mathbf{e} \rangle| \leq q/4$  时解密正确. 因此, 在参数选取时应令  $q$  的取值相对于误差分布  $\chi$  和  $\ell$  相对大. 比如, 当  $\chi = D_{\mathbb{Z}, r}$  是离散 Gaussian 分布时,  $\langle \mathbf{r}, \mathbf{e} \rangle$  是参数至多为  $r\sqrt{\ell}$  的亚 Gaussian 分布, 其尺寸小于  $r\sqrt{\ell \ln(1/\varepsilon)/\pi}$  的概率至少为  $1 - 2\varepsilon$ . 为了确保解密错误的概率可忽略, 可设定  $r = \Theta(\sqrt{n})$ ,  $q = \tilde{O}(n)$ , 对应 LWE 错误率  $\alpha = r/q = 1/\tilde{O}(n)$ .

**定理 3.4**

如果判定性 LWE 假设成立, 则 Regev PKE 是 IND-CPA 安全的.



**证明** 令  $S_i$  表示敌手在  $\text{Game}_i$  中成功概率. 以游戏序列的方式组织证明如下:

$\text{Game}_0$ : 该游戏是标准的 IND-CPA 游戏. 挑战者  $\mathcal{CH}$  和敌手  $\mathcal{A}$  交互如下:

- 初始化:  $\mathcal{CH}$  运行  $\text{Setup}(1^\kappa)$  生成公开参数  $\mathbf{A} \in \mathbb{Z}_q^{\ell \times n}$ , 同时生成公私钥对, 其中私钥  $sk$  为随机向量  $\mathbf{s} \in \mathbb{Z}_q^n$ , 公钥  $pk$  为  $\mathbf{p} = \mathbf{A}\mathbf{s} + \mathbf{e} \in \mathbb{Z}_q^\ell$ , 其中  $\mathbf{e} \leftarrow \chi^\ell$ .
- 挑战:  $\mathcal{A}$  选取  $(m_0, m_1)$  发送给  $\mathcal{CH}$ .  $\mathcal{CH}$  随机选取  $\mathbf{r} \xleftarrow{\mathbb{R}} \{0, 1\}^\ell$ ,  $\beta \xleftarrow{\mathbb{R}} \{0, 1\}$ , 计算  $\mathbf{u} = \mathbf{r}^T \mathbf{A}$ ,  $v = \mathbf{r}^T \mathbf{p} + m_\beta \cdot \lfloor q/2 \rfloor$ , 发送  $(\mathbf{u}, v)$  给  $\mathcal{A}$  作为挑战密文.
- 猜测:  $\mathcal{A}$  输出对  $\beta$  的猜测  $\beta'$ .  $\mathcal{A}$  成功当且仅当  $\beta = \beta'$ .

根据定义, 我们有:

$$\text{Adv}_{\mathcal{A}}(\kappa) = |\Pr[S_0] - 1/2|$$

$\text{Game}_1$ : 与  $\text{Game}_0$  惟一不同的是  $\mathcal{CH}$  生成公钥的方式由计算  $\mathbf{A}\mathbf{s} + \mathbf{e}$  变为随机选取  $\mathbb{Z}_q^\ell$  上的向量. 在  $\text{Game}_1$  中,  $\vec{\mathbf{A}} = \mathbf{A} \mid \mathbf{p}$  是  $\mathbb{Z}_q^{\ell \times n}$  上的随机矩阵, 容易验证  $f_{\vec{\mathbf{A}}}(\mathbf{r}) = \mathbf{r}^T \vec{\mathbf{A}}$  是从  $\{0, 1\}^\ell$  到  $\mathbb{Z}_q^{n+1}$  的 universal hash, 由参数选取  $\ell > n \log q$  和剩余哈希引理 (leftover hash lemma) 可知, 函数的输出服从  $\mathbb{Z}_q^{n+1}$  上的均匀分布. 因此, 挑战密文完美掩盖了  $\beta$  的信息. 因此, 即使对于拥有无穷计算能力的敌手, 我们也有:

$$\text{Adv}_{\mathcal{A}}(\kappa) = |\Pr[S_1] - 1/2| = 0$$

**断言 3.1**

如果判定性 LWE 假设成立, 那么对于任意 PPT 敌手均有  $|\Pr[S_0] - \Pr[S_1]| \leq \text{negl}(\kappa)$ .



**证明** 证明的思路是反证. 若存在 PPT 敌手  $\mathcal{A}$  在  $\text{Game}_0$  和  $\text{Game}_1$  中成功的概率差不可忽略, 则可构造出 PPT 算法  $\mathcal{B}$  打破 LWE 困难问题. 令  $\mathcal{B}$  的 LWE 挑战实例为  $(\mathbf{A}, \mathbf{p})$ ,  $\mathcal{B}$  的目标是区分挑战实例是随机采样还是 LWE 采样. 为此  $\mathcal{B}$  扮演 IND-CPA 游戏中的挑战者与  $\mathcal{A}$  交互如下:

- 初始化:  $\mathcal{B}$  发送  $(\mathbf{A}, \mathbf{p})$  给  $\mathcal{A}$ . 该操作将  $pk$  隐式地设定为  $\mathbf{p}$ .
- 挑战:  $\mathcal{A}$  选取  $(m_0, m_1)$  发送给  $\mathcal{CH}$ .  $\mathcal{CH}$  随机选取  $\mathbf{r} \xleftarrow{\mathbb{R}} \{0, 1\}^\ell$ ,  $\beta \xleftarrow{\mathbb{R}} \{0, 1\}$ , 计算  $\mathbf{u} = \mathbf{r}^T \mathbf{A}$ ,  $v = \mathbf{r}^T \mathbf{p} + m_\beta \cdot \lfloor q/2 \rfloor$ , 发送  $(\mathbf{u}, v)$  给  $\mathcal{A}$  作为挑战密文.
- 猜测:  $\mathcal{A}$  输出对  $\beta$  的猜测  $\beta'$ . 如果  $\beta = \beta'$ ,  $\mathcal{B}$  输出 1.

对上述交互分析可知, 如果  $\mathbf{p}$  是 LWE 采样, 那么  $\mathcal{B}$  完美模拟了  $\text{Game}_0$ ; 如果  $\mathbf{p}$  是随机采样, 那么  $\mathcal{B}$  完美模拟了  $\text{Game}_1$ . 因此,  $\mathcal{B}$  解决 LWE 挑战的优势  $\text{Adv}_{\mathcal{B}}^{\text{LWE}}(\kappa) = |\Pr[S_0] - \Pr[S_1]|$ . 如果 LWE 假设成立, 我们有  $|\Pr[S_0] - \Pr[S_1]| \leq \text{negl}(\kappa)$ . □

综上, 定理得证. □

**注记 3.1**

Regev PKE 和 Goldwasser-Micali PKE 在设计上有异曲同工之处, 均采用的是有损加密思想, 即公钥存在正常和有损这两种计算不可区分的类型, 正常公钥生成的密文可以正确解密, 而有损公钥生成的密文丢失了明文的全部信息. 在安全性证明时, 利用两种类型公钥的计算不可区分性以及有损加密的性质, 即可完成 IND-CPA 安全的论证. ♠

**3.3.2 GPV PKE**

Gentry, Peikert 和 Vaikuntanathan [GPV08] 同样基于 LWE 假设设计出一个 PKE 方案, 称为 GPV PKE.

GPV PKE 由以下 4 个多项式时间算法组成:



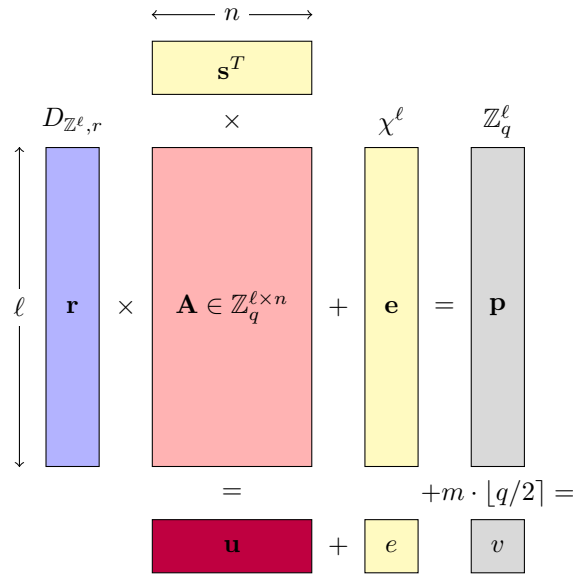


图 3.2: GPV PKE 加密方案示意图

- **Setup**( $1^\kappa$ ): 以安全参数  $\kappa$  为输入, 生成随机矩阵  $\mathbf{A} \in \mathbb{Z}_q^{\ell \times n}$  作为公开参数.
- **KeyGen**( $pp$ ): 以公开参数  $pp$  为输入, 随机选取噪声向量  $\mathbf{r} \xleftarrow{\mathcal{R}} D_{\mathbb{Z}^\ell, r}$  作为私钥, 计算  $\mathbf{u}^T \leftarrow \mathbf{r}^T \mathbf{A}$  作为公钥. 从编码的角度,  $\mathbf{u}$  可以理解  $\mathbf{e}$  相对于  $\mathbf{A}$  的 syndrome.
- **Encrypt**( $pk, m$ ): 以公钥  $pk = \mathbf{u}$  和明文  $m \in \{0, 1\}$  为输入, 随机选取向量  $\mathbf{s} \xleftarrow{\mathcal{R}} \mathbb{Z}_q^n$  和  $\mathbf{e} \xleftarrow{\mathcal{R}} \chi^\ell$ , 随机选取  $e \xleftarrow{\mathcal{R}} \chi$ , 计算  $\mathbf{p} = \mathbf{A}\mathbf{s} + \mathbf{e} \in \mathbb{Z}_q^\ell$ ,  $v = \mathbf{u}^T \mathbf{s} + e + m \cdot \lfloor q/2 \rfloor$  作为密文.
- **Decrypt**( $\mathbf{r}, c$ ): 以私钥  $sk = \mathbf{r}$  和密文  $c = (\mathbf{p}, v)$ , 计算  $y = v - \mathbf{r}^T \mathbf{p} \in \mathbb{Z}_q$ , 若  $y$  接近 0 则输出 0, 若  $y$  接近  $\lfloor q/2 \rfloor$  则输出 1.

**正确性.** 观察以下等式:

$$\begin{aligned}
 y &= v - \mathbf{r}^T \mathbf{p} \\
 &= \mathbf{u}^T \mathbf{s} + e + m \cdot \lfloor q/2 \rfloor - \mathbf{r}^T (\mathbf{A}\mathbf{s} + \mathbf{e}) \\
 &= \mathbf{u}^T \mathbf{s} + e + m \cdot \lfloor q/2 \rfloor - \mathbf{u}^T \mathbf{s} - \mathbf{r}^T \mathbf{e} \\
 &= m \cdot \lfloor q/2 \rfloor + e - \mathbf{r}^T \mathbf{e}
 \end{aligned}$$

由上述推导可知, 当累计误差  $|\langle \mathbf{e} - \mathbf{r}^T \mathbf{e} \rangle| \leq q/4$  时解密正确. 通过恰当的参数选择, 可确保累计误差以接近 1 的绝对优势概率小于等于  $q/4$ , 更多细节请参考 [GPV08]

### 定理 3.5

如果判定性 LWE 假设成立, 则 GPV PKE 是 IND-CPA 安全的.

**证明** 令  $S_i$  表示敌手在  $\text{Game}_i$  中成功概率. 以游戏序列的方式组织证明如下:

**Game<sub>0</sub>:** 该游戏是标准的 IND-CPA 游戏. 挑战者  $\mathcal{CH}$  和敌手  $\mathcal{A}$  交互如下:

- **初始化:**  $\mathcal{CH}$  运行 **Setup**( $1^\kappa$ ) 生成公开参数  $\mathbf{A} \in \mathbb{Z}_q^{\ell \times n}$ , 同时生成公私钥对, 其中私钥  $sk$  为随机向量  $\mathbf{r} \in D_{\mathbb{Z}^\ell, r}$ , 公钥  $pk$  为  $\mathbf{u} = \mathbf{r}^T \mathbf{A}$ .
- **挑战:**  $\mathcal{A}$  选取  $(m_0, m_1)$  发送给  $\mathcal{CH}$ .  $\mathcal{CH}$  随机选取  $\mathbf{s} \xleftarrow{\mathcal{R}} \mathbb{Z}_q^n$ , 随机选取  $\mathbf{e} \xleftarrow{\mathcal{R}} \chi^\ell$  和  $e \xleftarrow{\mathcal{R}} \chi$ ,  $\beta \xleftarrow{\mathcal{R}} \{0, 1\}$ , 计算  $\mathbf{p} = \mathbf{A}\mathbf{s} + \mathbf{e} \in \mathbb{Z}_q^\ell$ ,  $v = \mathbf{u}^T \mathbf{s} + e + m_\beta \cdot \lfloor q/2 \rfloor$  作为密文. 发送  $(\mathbf{u}, v)$  给  $\mathcal{A}$  作为挑战密文.
- **猜测:**  $\mathcal{A}$  输出对  $\beta$  的猜测  $\beta'$ .  $\mathcal{A}$  成功当且仅当  $\beta = \beta'$ .



根据定义, 我们有:

$$\text{Adv}_{\mathcal{A}}(\kappa) = |\Pr[S_0] - 1/2|$$

**Game<sub>1</sub>**: 与 **Game<sub>0</sub>** 惟一不同的是  $\mathcal{CH}$  生成公钥的方式由计算  $\mathbf{u}^T = \mathbf{r}^T \mathbf{A}$  变为随机选取  $\mathbf{u} \xleftarrow{\mathbf{R}} \mathbb{Z}_q^n$  上的向量. 在 **Game<sub>1</sub>** 中,  $(\mathbf{A}, \mathbf{p} = \mathbf{A}\mathbf{s} + \mathbf{e}, \mathbf{u}, \mathbf{u}^T \mathbf{s} + x)$  恰好构成  $\ell + 1$  个 **LWE** 采样结果. 有 **LWE** 假设立刻可知, 敌手在 **Game<sub>1</sub>** 中的视角计算意义下隐藏了  $\beta$  的信息, 因此基于 **LWE** 假设有:

$$\text{Adv}_{\mathcal{A}}(\kappa) = |\Pr[S_1] - 1/2| \leq \text{negl}(\kappa)$$

### 断言 3.2

对于任意的敌手  $\mathcal{A}$  (即使拥有无穷计算能力), 均有  $|\Pr[S_0] - \Pr[S_1]| \leq \text{negl}(\kappa)$



**证明** 根据  $\ell \geq 2n \log q$  的参数选择可知, 公钥  $\mathbf{u}$  的分布与  $\mathbb{Z}_q^n$  上的均匀分布统计不可区分, 因此敌手在 **Game<sub>0</sub>** 和 **Game<sub>1</sub>** 中的视图统计不可区分, 从而  $|\Pr[S_0] - \Pr[S_1]| \leq \text{negl}(\kappa)$ . □

综上, 定理得证. □

### 注记 3.2

**Regev PKE** 和 **GPV PKE** 在形式上相似, 构造使用了相同的元素  $\mathbf{A}, \mathbf{s}, \mathbf{r}, \mathbf{e}, \mathbf{p}, \mathbf{u}, v$ , 但用途含义不完全相同, 构造互为对偶. **Regev PKE** 中,  $\mathbf{p}$  为公钥,  $(\mathbf{s}, \mathbf{e})$  为私钥,  $\mathbf{u}$  为密文; **GPV PKE** 中  $\mathbf{p}$  为密文,  $(\mathbf{s}, \mathbf{e})$  为加密随机数,  $\mathbf{u}$  为公钥. 感兴趣的读者可以参考 Micciancio [Mic10] 了解更多格密码学中的对偶性. **Regev PKE** 中, 公钥空间是稀疏的; **GPV PKE** 中, 公钥空间是稠密的, 正是利用公钥空间的稠密性, 我们可以借助随机谕言机将 **GPV PKE** 升级为身份加密方案 **GPV IBE**. ♠

## 第四章 通用构造方法

### 内容提要

- ❑ 单向陷门函数类
- ❑ 哈希证明系统类
- ❑ 可提取哈希证明系统类
- ❑ 程序混淆类
- ❑ 统一构造框架

## 4.1 单向陷门函数类

### 4.1.1 单向陷门函数

单向陷门函数 (TDF) 是单向函数 (OWF) 在 Cryptomania 中的对应, 简言之, 其正向计算容易, 逆向计算困难但在有陷门信息辅助时同样容易.

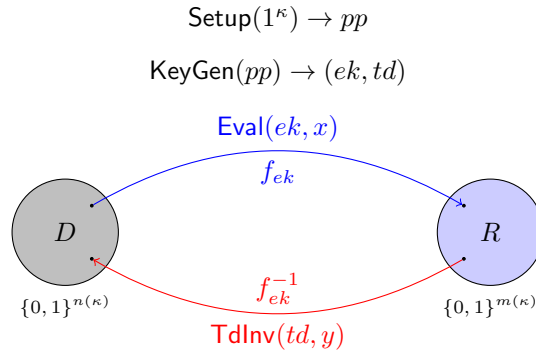
#### 定义 4.1 (单向陷门函数)

TDF 由以下四个多项式时间算法组成:

- **Setup**( $1^\kappa$ ): 系统生成算法以安全参数  $\kappa$  为输入, 输出公开参数  $pp$ , 其中  $pp$  包含对定义域  $D$ , 值域  $R$ , 求值公钥空间  $EK$ 、求逆陷门空间  $TD$  和单向陷门函数族  $f: EK \times D \rightarrow R$  的描述. 换言之,  $f$  是由求值公钥索引的函数族. 不失一般性,  $D$  支持高效的随机采样, 即存在 PPT 算法 **SampDom** 可以从  $D$  中随机选取一个元素. 在多数情况下,  $D$  和  $R$  是与求值公钥无关的 (该性质也被称为 index-independent), 但在有些情形下,  $D$  和  $R$  是由求值公钥索引的空间簇. 为了叙述简洁, 以下均假设  $D$  和  $R$  是单一空间. 空间簇的情形由单一集合的情形自然推广得到.
- **KeyGen**( $pp$ ): 以公共参数  $pp$  为输入, 输出密钥对  $(ek, td)$ , 其中  $ek$  为求值公钥,  $td$  为求逆陷门.
- **Eval**( $ek, x$ ): 以求值公钥  $ek$  和定义域元素  $x \in D$  为输入, 输出  $y \leftarrow f_{ek}(x)$ .
- **TdInv**( $td, y$ ): 以求逆陷门  $td$  和值域元素  $y \in R$  为输入, 输出  $x \in D$  或特殊符号  $\perp$  指示  $y$  不存在原像.

定义以下两条性质:

- 单射:  $\forall ek$ , 称  $f_{ek}$  是单射的当且仅当  $x \neq x' \Rightarrow f_{ek}(x) \neq f_{ek}(x')$ .
- 置换:  $\forall ek, \text{Img}(f_{ek}) = D = R$ .



**正确性.**  $\forall \kappa \in \mathbb{N}, \forall pp \leftarrow \text{Setup}(1^\kappa), \forall (ek, td) \leftarrow \text{KeyGen}(pp)$  和  $\forall x \in D$  和  $y = \text{Eval}(ek, x)$ , 有:

$$\Pr[\text{TdInv}(td, y) \in f_{ek}^{-1}(y)] = 1$$

**单向性.** 定义单向陷门函数敌手  $\mathcal{A}$  的优势函数如下:

$$\text{Adv}_{\mathcal{A}}(\kappa) = \Pr \left[ \begin{array}{l} pp \leftarrow \text{Setup}(1^\kappa); \\ (ek, td) \leftarrow \text{KeyGen}(pp); \\ x^* \xleftarrow{\mathbb{R}} D, y^* \leftarrow \text{Eval}(ek, x^*); \\ x \leftarrow \mathcal{A}(pp, ek, y^*) \end{array} : x \in f_{ek}^{-1}(y^*) \right],$$

如果对于任意的 PPT 敌手  $\mathcal{A}$ , 其优势函数均是可忽略的, 则称该陷门函数是单向的.

## 注记 4.1

1. 不失一般性, 假定  $D$  和  $R$  均存在经典表示, 分别是  $\{0, 1\}^{n(\kappa)}$  和  $\{0, 1\}^{m(\kappa)}$ , 其中  $n(\cdot)$  和  $m(\cdot)$  是关于  $\kappa$  的多项式函数. 容易验证, 长度函数不能过大, 如果  $n(\cdot)$  或  $m(\cdot)$  是超多项式函数, 则函数无法高效计算; 长度函数也不能过小, 如果  $n(\cdot)$  或  $m(\cdot)$  是亚线性函数, 则函数不可能满足单向性.
2. 在抽象定义中, 只限定了  $\text{TDFInv}(td, \cdot)$  在输入为像集元素时返回原像, 而未限定其输入为非像集元素时的行为. 在具体构造时,  $\text{TDFInv}(td, \cdot)$  在输入为非像集元素时的行为往往需要精心设定, 以方便安全性证明.
3. 在单向性的定义中, 敌手  $\mathcal{A}$  仅观察到  $ek$  和  $y^*$  的信息.  $x^* \xleftarrow{R} D$  可以放宽至  $x^*$  选自  $D$  上的高最小熵分布, 即  $H_\infty(x^*) \geq \omega(\log \kappa)$ .

在展示如何使用单向陷门函数构造公钥方案之前, 先展示一个基于单向陷门置换的构造. 该构造并不安全, 但对得到正确的构造很有启发意义.

## 构造 4.1 (朴素的 TDF-based PKE(不安全))

- $\text{Setup}(1^\kappa)$ : 运行  $\text{TDP.Setup}(1^\kappa)$  生成公开参数  $pp$ , 其中明文空间和密文空间均为单向陷门置换的定义域  $D$ .
- $\text{KeyGen}(pp)$ : 运行  $\text{TDP.KeyGen}(pp) \rightarrow (ek, td)$ , 其中  $ek$  作为加密公钥,  $td$  作为解密私钥.
- $\text{Encrypt}(ek, m)$ : 以公钥  $ek$  和明文  $m \in D$  为输入, 运行  $\text{TDP.Eval}(ek, m)$  计算  $c \leftarrow f_{ek}(m)$  作为密文.
- $\text{Decrypt}(td, c)$ : 以私钥  $td$  和密文  $c \in D$  为输入, 运行  $\text{TDP.TdInv}(td, c)$  计算  $m \leftarrow f_{ek}^{-1}(c)$  恢复明文.

上述构造来自 Diffie 和 Hellman 的经典论文 [DH76], 原始的 RSA 公钥加密方案就是该构造的具体实例化. 该构造的想法直观, 利用单向陷门置换将明文转化为密文, 同时利用陷门可以求逆从密文中恢复明文. 但其仅仅满足较弱的 OW-CPA 安全, 并不满足现在公认的最低要求 IND-CPA 安全, 因此其也被成为公钥加密的 textbook 构造<sup>1</sup>. 朴素构造不满足 IND-CPA 安全的根本原因是加密算法是确定型的而非概率型的, 因此敌手可以通过“加密-比较”即可打破 IND-CPA 安全. 因此, 强化朴素构造的第一步是选择定义域中的随机元素  $x$ , 计算其函数值  $f_{ek}(x)$  作为封装密文, 再用  $x$  作为会话密钥掩蔽明文. 强化构造仍然不满足 IND-CPA 安全性, 原因是  $f_{ek}(\cdot)$  是公开可计算函数, 其函数值泄漏了原像信息, 使得原像在敌手的视角中不再伪随机. 针对性的强化方法是计算  $x$  的 hardcore function 值作为会话密钥.

以下首先介绍 hardcore function 的概念.

## 定义 4.2 (hardcore function)

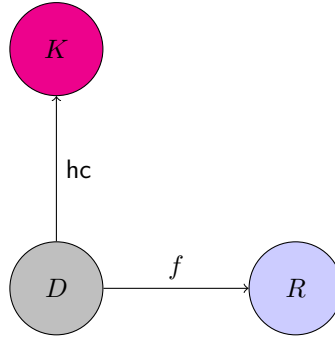
称多项式时间可计算的确性型函数  $hc: D \rightarrow K$  是函数  $f: D \rightarrow R$  的 hardcore function 当且仅当:

$$(f(x^*), hc(x^*)) \approx_c (f(x^*), U_K)$$

其中概率空间建立在  $x^* \xleftarrow{R} D$  的随机带上. 以安全实验的方式可如下定义, 即对于任意 PPT 敌手  $\mathcal{A}$ , 其安全优势可忽略:

$$\text{Adv}_{\mathcal{A}}(\kappa) = \Pr \left[ \beta' = \beta : \begin{array}{l} pp \leftarrow \text{Setup}(1^\kappa); \\ x^* \xleftarrow{R} D, y^* \leftarrow f(x^*); \\ k_0^* \leftarrow hc(x^*), k_1^* \xleftarrow{R} K, \beta \xleftarrow{R} \{0, 1\}; \\ \beta' \leftarrow \mathcal{A}(pp, ek, y^*, k_\beta^*); \end{array} \right] - \frac{1}{2}$$

<sup>1</sup>textbook 指其仅适合作为以科普为目的的教学.

**定理 4.1 (Goldreich-Levin Theorem)**

如果  $f : \{0, 1\}^n \rightarrow \{0, 1\}^m$  是单向函数, 那么  $GL(x) = \bigoplus_{i=1}^n x_i r_i$  是从  $\{0, 1\}^n$  到  $\{0, 1\}$  的单比特输出硬核函数, 或称为硬核谓词.

**注记 4.2**

Goldreich-Levin 定理是现代密码学中极为重要的结论, 它的意义在于通过显式构造硬核函数, 建立起单向性与伪随机性之间的关联. 从另一个角度理解, GL 硬核谓词可以看做一个计算意义下的随机性提取器, 对  $x|f(x)$  的计算熵进行随机性提取, 萃取出伪随机的一比特. 还需要特别说明的是, 到目前为止尚不知晓如何针对任意单向函数  $f$  设计一个确定型的硬核谓词. GL 是相对于  $g(x, r) := f(x)||r$  的硬核谓词, 或者可以将  $r \xleftarrow{R} \{0, 1\}^n$  理解为硬核谓词的描述, 将 GL 理解为  $f$  的随机性硬核谓词. 本书中采用第二种观点.

另一方面, GL 硬核谓词是通用的 (universal), 即构造对于任意单向函数均成立. 强通用性的代价是效率较低, 输出仅是单比特. 当单向函数具有特殊的结构 (如函数是置换) 或者依赖额外困难假设 (如判定性假设、差异输入程序混淆假设) 时, 存在更高效的构造.



下面我们展示如何基于单射的单向陷门函数构造构造 KEM 方案.

**构造 4.2 (TDF-based KEM)**

- **Setup( $1^\kappa$ ):** 运行  $\text{TDF.Setup}(1^\kappa)$  生成公开参数  $pp$ .  $pp$  中不仅包含单向陷门函数  $f_{ek} : D \rightarrow R$  的描述, 还包括相应硬核函数  $hc : D \rightarrow K$  的描述. KEM 方案的密文空间是 TDF 的定义域  $D$ , 密钥空间是硬核函数的值域  $K$ .
- **KeyGen( $pp$ ):** 运行  $\text{TDF.KeyGen}(pp) \rightarrow (ek, td)$ , 其中  $ek$  作为封装公钥  $pk$ ,  $td$  作为解封装私钥  $sk$ .
- **Encaps( $pk, m$ ):** 以公钥  $pk = ek$  为输入, 随机选取  $x \xleftarrow{R} D$ , 运行  $\text{TDF.Eval}(ek, x)$  计算  $c \leftarrow f_{ek}(m)$  作为封装密文, 计算  $k \leftarrow hc(x)$  作为会话密钥.
- **Decaps( $sk, c$ ):** 以私钥  $sk = td$  和密文  $c$  为输入, 运行  $\text{TDF.TdInv}(td, c)$  计算  $x \leftarrow f_{ek}^{-1}(c)$ , 输出  $k \leftarrow hc(x)$ .



**正确性.** 由单向陷门函数的单射性质和求逆算法的正确性可知, 上述 KEM 构造满足正确性.

**定理 4.2**

如果  $f_{ek}$  是一族单射单向陷门函数, 那么上述 KEM 构造是 IND-CPA 安全的.



**证明** 证明的思路是单一归约, 即若存在敌手  $\mathcal{A}$  打破 KEM 方案的 IND-CPA 安全性, 则存在敌手  $\mathcal{B}$  打破  $hc$  的伪随机性, 进而与  $f_{ek}$  的单向性矛盾. 令  $\mathcal{B}$  的挑战实例为  $(pp, ek, y^*, k_\beta^*)$ , 其中  $pp$  为单射单向陷门函数的公开参数,  $ek$  为随机生成的求值密钥,  $y^* \leftarrow f_{ek}(x^*)$  是随机选取原像  $x^*$  的像,  $k_0^* \leftarrow hc(x^*)$ ,  $k_1^* \xleftarrow{R} K$ .  $\mathcal{B}$  的目标是判定  $\beta = 0$  抑或  $\beta = 1$ .  $\mathcal{B}$  与  $\mathcal{A}$  交互如下:

- **初始化:**  $\mathcal{B}$  根据  $pp$  生成 KEM 方案的公开参数, 并设定公钥  $pk := ek$ , 将  $(pp, ek)$  发送给  $\mathcal{A}$ .
- **挑战:**  $\mathcal{B}$  设定  $c^* := y^*$ , 将  $(c^*, k_\beta^*)$  发送给  $\mathcal{A}$ .

- 猜测:  $\mathcal{A}$  输出  $\beta'$ ,  $\mathcal{B}$  将  $\beta'$  转发给它自身的挑战者.

容易验证,  $\mathcal{B}$  完美地模拟了 KEM 方案中的挑战者,  $\mathcal{B}$  成功当且仅当  $\mathcal{A}$  成功. 因此我们有:

$$\text{Adv}_{\mathcal{A}}^{\text{KEM}}(\kappa) = \text{Adv}_{\mathcal{B}}^{\text{hc}}(\kappa)$$

由  $f_{ek}$  的单向性可知, hc 伪随机, 从 KEM 构造满足 IND-CPA 安全性. □

以上的结果展示了单射单向陷门函数蕴含 IND-CPA 的公钥加密. 需要确认 TDF 与 CCA PKE 之间是否存在分离一个自然的问题是, 单向陷门函数需要满足何种性质才能蕴含 IND-CCA 的公钥加密. 以下, 我们按照时间先后顺序依次介绍单向陷门函数的三个增强版本, 并展示如何基于这些增强版本的单向陷门函数构造 IND-CCA 的公钥加密.

### 4.1.2 有损陷门函数

天之道, 损有余而补不足, 是故虚胜实, 不足胜有余.

— 宋·黄裳《九阴真经》

理想世界中的镜中月 and 水中花体现的是信息完美复刻, 而现实世界中更多的现象体现的却是信息有损, 如拍照、录音, 无论设备和手段多么先进, 都无法做到完美复刻信源信息, 只能做到尽可能的高保真. 单射函数可以形象的理解为理想世界中信息无损的编码过程, 那么什么形式的函数刻画了现实世界中信息有损的编码过程呢? Peikert 和 Waters [PW08] 正是基于上述的思考, 在 2008 年开创性提出了有损陷门函数的概念. 简言之, 有损陷门函数有两种模式, 即正常和有损模式. 在单射模式下, 函数是单射的, 像完全保留了原像的全部信息; 在有损模式下, 函数是有损的, 像在信息论意义下丢失了原像的部分信息. 两种模式之间的关联是计算不可区分.

#### 定义 4.3 (有损陷门函数 LTDF)

有损陷门函数 LTDF 由  $n$  和  $\tau$  两个参数刻画, 包含以下五个多项式时间算法:

- $\text{Setup}(1^\kappa)$ : 系统生成算法以安全参数  $\kappa$  为输入, 输出公开参数  $pp$ , 其中  $pp$  包含对定义域  $X$  和值域  $Y$  的描述. 其中  $|X| = 2^{n(\kappa)}$ .
- $\text{GenInjective}(pp)$ : 以公共参数  $pp$  为输入, 输出密钥对  $(ek, td)$ , 其中  $ek$  为求值公钥,  $td$  为求逆陷门. 该算法输出的  $ek$  定义了从  $X$  到  $Y$  的单射函数  $f_{ek}$ , 拥有对应  $td$  可以对  $f_{ek}$  进行高效求逆.
- $\text{GenLossy}(pp)$ : 以公共参数  $pp$  为输入, 输出密钥对  $(ek, \perp)$ , 其中  $ek$  为求值公钥,  $\perp$  表示陷门不存在无法求逆. 该算法输出的  $ek$  定义了从  $D$  到  $R$  的有损函数  $f_{ek}$ , 像集的大小至多为  $2^{\tau(\kappa)}$ .
- $\text{Eval}(ek, x)$ : 以求值公钥  $ek$  和定义域元素  $x \in X$  为输入, 输出  $y \leftarrow f_{ek}(x)$ .
- $\text{TdInv}(td, y)$ : 以求逆陷门  $td$  和值域元素  $y \in Y$  为输入, 输出  $x \in X$  或特殊符号  $\perp$  指示  $y$  不存在原像.

模式不可区分性.  $\text{GenInjective}(pp)$  和  $\text{GenLossy}(pp)$  的第一个输出构成的分布在计算意义下不可区分, 即任意 PPT 敌手无法判定求值公钥  $ek$  属于单射模式还是有损模式. ♣

相比常规的单向陷门函数, 有损陷门函数额外具备一个计算不可区分的有损模式, 这正是其威力的来源. 在利用有损陷门函数设计密码方案/协议时, 通常按照如下的步骤:

1. 在单射模式下完成密码方案/协议的功能性构造 (功能性通常需要函数单射可逆)
2. 在有损模式下完成密码方案/协议的安全性论证 (论证通常在信息论意义下进行)
3. 利用单射模式和有损模式的计算不可区分性证明密码方案/协议在正常模式下计算安全性.

细心的读者可能已经发现了有损陷门函数的定义中并没有显式的要求函数在单射模式下具备单向性, 这是因为单射和有损模式的计算不可区分性已经隐式的保证了这一点. 以下进行严格证明, 具体展示应用有损陷门函数设计密码方案/协议的过程.

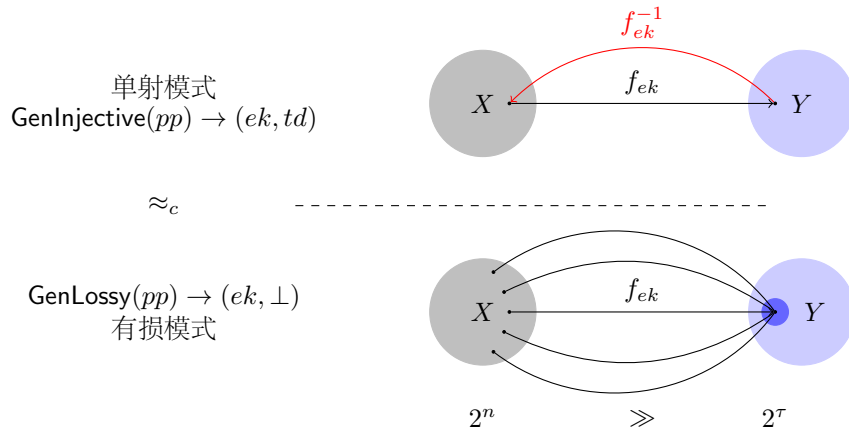


图 4.1: 有损陷门函数 (LTDF) 示意图

**定理 4.3**

令  $\mathcal{F}$  是一族  $(n, \tau)$ -LTDF, 当  $n - \tau \geq \omega(\log \lambda)$  时,  $\mathcal{F}$  的单射模式构成一族单射单向陷门函数.

**证明** 证明通过游戏序列组织.

**Game<sub>0</sub>**: 该游戏是标准的单射单向陷门函数安全游戏. 挑战者  $\mathcal{CH}$  和敌手  $\mathcal{A}$  交互如下:

- 初始化:  $\mathcal{CH}$  运行  $pp \leftarrow \text{Setup}(\kappa)$ ,  $(ek, td) \leftarrow \text{GenInjective}(pp)$ , 发送  $(pp, ek)$  给  $\mathcal{A}$ .
- 挑战阶段:  $\mathcal{CH}$  随机选择  $x^* \xleftarrow{R} X$ , sends  $y^* \leftarrow f_{ek}(x^*)$  给  $\mathcal{A}$ .
- 猜测阶段:  $\mathcal{A}$  输出  $x'$ ,  $\mathcal{A}$  赢得游戏当且仅当  $x' = x^*$ .

根据定义, 我们有:

$$\text{Adv}_{\mathcal{A}}(\kappa) = \Pr[S_0]$$

**Game<sub>1</sub>**: 该游戏与上一个游戏完全相同, 唯一不同的是将单射模式切换到有损模式

- 初始化:  $\mathcal{CH}$  运行  $(ek, \perp) \leftarrow \text{GenLossy}(pp)$  生成求值公钥  $ek$ .

根据定义, 我们有:

$$\text{Adv}_{\mathcal{A}}(\kappa) = \Pr[S_1]$$

**断言 4.1**

单射和有损两种模式的计算不可区分性保证了  $|\Pr[S_0] - \Pr[S_1]| \leq \text{negl}(\kappa)$ .

**证明** 我们利用反证法完成归约论证: 若  $|\Pr[S_0] - \Pr[S_1]|$  不可忽略, 则可构造出 PPT 的敌手  $\mathcal{B}$  打破模式的不可区分性.  $\mathcal{B}$  在收到模式不可区分性的挑战  $(pp, ek)$  后, 将  $(pp, ek)$  发送给  $\mathcal{A}$ , 随后随机选取  $x^* \xleftarrow{R} X$ , 计算并发送  $y^* \leftarrow f_{ek}(x^*)$  给  $\mathcal{A}$ . 当收到  $\mathcal{A}$  的输出  $x'$  后, 若  $x' = x^*$ ,  $\mathcal{B}$  输出 '1', 否则输出 '0'. 分析可知, 当  $ek$  来自单射模式时,  $\mathcal{B}$  完美的模拟了 **Game<sub>0</sub>**; 当  $ek$  来自有损模式时,  $\mathcal{B}$  完美的模拟了 **Game<sub>1</sub>**. 因此, 我们有:

$$|\Pr[\mathcal{B}(ek) = 1 : ek \leftarrow \text{GenInjective}(pp)] - \Pr[\mathcal{B}(ek) = 1 : ek \leftarrow \text{GenLossy}(pp)]| = |\Pr[S_0] - \Pr[S_1]|$$

其中  $pp \leftarrow \text{Setup}(1^\kappa)$ . □

**断言 4.2**

对于任意的敌手  $\mathcal{A}$  (即使拥有无穷计算能力), 其在 **Game<sub>1</sub>** 中的优势也是可忽略的. □

**证明** **Game<sub>1</sub>** 处于有损模式, 因此由 Chaining Lemma 2.2 可知,  $x^*$  的平均条件最小熵  $\tilde{H}_\infty(x^*|y^*) \geq n - \tau \geq \omega(\log \kappa)$ , 从而断言得证. □



综合以上, 定理得证! □

#### 注记 4.3

有损陷门函数相比标准单向陷门函数多了有损模式, 也正因为如此, 其具有标准单向陷门函数很多不具备的优势。

在安全方面, 根据上述论证容易验证只要参数设置满足一定约束, 则有损(陷门)函数在泄漏模型下仍然安全. 具体的, 在敌手获得关于原像任意长度为  $\ell$  有界泄漏的情形下, 只要  $n - \tau - \ell \geq \omega(\log \kappa)$ , 则单向性依然成立. 因此, 有损(陷门)函数是构造抗泄漏单向函数的重要工具 [Kom16; CWZ18].

在效率方面, 令  $\mathcal{H}$  是一族从  $X$  到  $\{0, 1\}^{m(\kappa)}$  的对独立哈希函数族 (pairwise-independent hash family), 只要  $n - \tau - m \geq \omega(\log \kappa)$ , 那么从  $\mathcal{H}$  中随机选择的  $h$  即构成单向函数的多比特输出 hardcore function. 论证的方式是应用 Leftover Hash Lemma 2.3 和对独立哈希函数族构成强随机性提取器的事实, 得到 hardcore function 输出和均匀随机输出不可区分的结论. ♠

有损陷门函数还有一个非平凡的扩展, 称为全除一 (ABO, All-But-One) 有损陷门函数. 简言之, ABO-TDF 存在一个分支集合 (branch set), 记为  $B$ . 求值密钥  $ek$  和分支值  $b \in B$  共同定义了从  $X$  到  $Y$  的函数  $f_{ek,b}$ , 该函数当且仅当  $b$  等于某特定一个分支值时有损, 在其它分支均单射可逆. 严格定义如下:

#### 定义 4.4 (全除一有损陷门函数)

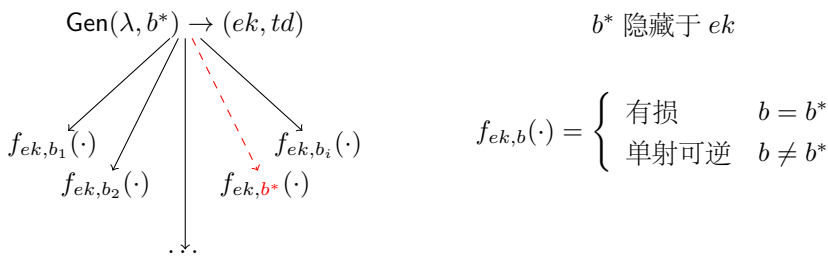
ABO-TDF 由  $n$  和  $\tau$  两个参数刻画, 包含以下五个多项式时间算法:

- $\text{Setup}(1^\kappa)$ : 系统生成算法以安全参数  $\kappa$  为输入, 输出公开参数  $pp$ , 其中  $pp$  包含对定义域  $X$ 、值域  $Y$  和分支集合  $B$  的描述. 其中  $|X| = 2^{n(\kappa)}$ .
- $\text{Gen}(pp, b^*)$ : 以公共参数  $pp$  和给定分支值  $b^* \in B$  为输入, 输出密钥对  $(ek, td)$ , 其中  $ek$  为求值公钥,  $td$  为求逆陷门. 该算法输出的  $ek$  和分支值  $b \in B$  定义了从  $X$  到  $Y$  的函数  $f_{ek,b}$ . 当  $b \neq b^*$  时,  $f_{ek,b}$  单射且拥有对应  $td$  可高效求逆; 当  $b = b^*$  时,  $f_{ek,b^*}$  有损, 像集的大小至多为  $2^{\tau(\kappa)}$ ,  $b^*$  因此称为有损分支.
- $\text{Eval}(ek, b, x)$ : 以求值公钥  $ek$ 、分支值  $b \in B$  和定义域元素  $x \in X$  为输入, 输出  $y \leftarrow f_{ek,b}(x)$ .
- $\text{TdInv}(td, b, y)$ : 以求逆陷门  $td$ 、分支值  $b \in B$  和值域元素  $y \in Y$  为输入, 输出  $x \in X$  或特殊符号  $\perp$  指示  $y$  不存在原像.

有损分支隐藏性. 该性质刻画的安全要求是求值公钥不泄漏有损分支的信息. 严格定义类似公钥加密的不可区分安全或是承诺的隐藏性, 即  $\forall b_0, b_1 \in B$ , 我们有:


$$\text{Gen}(pp, b_0) \approx_c \text{Gen}(pp, b_1)$$

其中  $pp \leftarrow \text{Setup}(1^\kappa)$ . ♣



#### 注记 4.4

ABO-TDF 可以理解为 LTDF 的扩展, 分支集合由  $\{0, 1\}$  延拓至  $\{0, 1\}^b$ . LTDF 已经有较为丰富的应用, 如 IND-CPA 的公钥加密方案、不经意传输、抗碰撞哈希函数等; LTDF 与 ABO-TDF 结合有着更强的应用, 如

IND-CCA 的公钥加密方案. IND-CCA 的公钥加密方案构造原理蕴含在如何基于 LTDF 和 ABO-LTDF 构造更高级的单向陷门函数中 (将在章节中阐述), 为了避免重复, 此处不再详述. 

以下展示如何给出 LTDF 和 ABO-TDF 的具体构造. 构造的难点是需要巧妙设计密钥对生成算法, 使其可以工作在单射可逆和有损两个模式, 且两种模式在计算意义下不可区分. 设计的思路是令定义域  $X$  是向量空间, 输入  $x$  是向量空间中的元素, 求值公钥  $ek$  是刻画线性变换的矩阵, 函数求值  $f(ek, x)$  的过程就是对输入进行线性变换, 当  $ek$  满秩时, 函数单射可逆; 当  $ek$  非满秩时, 函数有损. 隐藏  $ek$  工作模式的思路则是对其“加密”. 我们称上述技术路线为矩阵式方法.

下面展示矩阵式构造的一个具体例子, 以剥丝抽茧的方式阐明设计思想和关键技术.

**隐藏矩阵生成.** 最简单的满秩矩阵是单位阵, 最简单的非满秩矩阵是全零阵, 两者之间差异显著, 为了保证计算不可区分性, 思路是生成一个伪随机的隐藏矩阵 (concealer matrix)  $\mathbf{M}$  对其加密. 我们期望  $\mathbf{M}$  满足如下结构:  $\mathbf{M}$  的所有行向量均处于同一个一维子空间, 后面可以看到子空间的描述将作为陷门信息使用. 具体的, 隐藏矩阵生成算法  $\text{GenConcealMatrix}(n)$  细节如下:

1. 随机选择  $\mathbf{r} = (r_1, \dots, r_n) \xleftarrow{\mathbb{R}} \mathbb{Z}_p^n$  和  $\mathbf{s} = (s_1, \dots, s_m, 1) \xleftarrow{\mathbb{R}} \mathbb{Z}_p^n \times \{1\}$
2. 计算张量积  $\mathbf{V} = \mathbf{r} \otimes \mathbf{s} = \mathbf{r}^t \mathbf{s} \in \mathbb{Z}_p^{n \times (n+1)}$

$$\mathbf{V} = \left( \begin{array}{cccc|c} r_1 s_1 & r_1 s_2 & \dots & r_1 s_n & r_1 \\ r_2 s_1 & r_2 s_2 & \dots & r_2 s_n & r_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ r_n s_1 & r_n s_2 & \dots & r_n s_n & r_n \end{array} \right)$$


3. 输出  $\mathbf{M} = g^{\mathbf{V}} \in \mathbb{G}^{n \times (m+1)}$  作为隐藏矩阵,  $\mathbf{s}$  作为陷门信息.

$$\mathbf{M} = \left( \begin{array}{cccc|c} g^{r_1 s_1} & g^{r_1 s_2} & \dots & g^{r_1 s_n} & g^{r_1} \\ g^{r_2 s_1} & g^{r_2 s_2} & \dots & g^{r_2 s_n} & g^{r_2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ g^{r_n s_1} & g^{r_n s_2} & \dots & g^{r_n s_n} & g^{r_n} \end{array} \right)$$


算法前两步的作用是生成特定结构: 通过张量积确保  $\mathbf{V}$  中所有行向量均处于向量  $(s_1, \dots, s_n, 1)$  张成的一维子空间中. 当前向量定义在有限域  $\mathbb{F}_p$  上, 而  $ek$  矩阵不可以定义在有限域  $\mathbb{F}_p$  上, 否则存在高效的算法判定  $ek$  对应的矩阵是否满秩. 令  $\mathbb{G}$  是  $p$  阶循环群, 其中 DDH 假设成立. 可以证明, 如果  $ek$  矩阵定义在  $\mathbb{G}$  上, 那么满秩和非满秩无法有效判定. 因此, 算法的第三步利用从  $\mathbb{F}_p$  到  $\mathbb{G}$  的同构映射  $\phi: t \rightarrow g^t$  将  $\mathbf{V}$  中的所有元素从  $\mathbb{F}_p$  提升到  $\mathbb{G}$  中.

#### 注记 4.5

如果将  $\mathbf{s}$  截断为  $\mathbf{s}' = (s_1, \dots, s_n)$ , 那么  $g^{\mathbf{r} \otimes \mathbf{s}'} = (g^{r_i \cdot s_j}) \in \mathbb{G}^{n \times n}$  恰好是 Naor-Reingold 基于 DDH 假设的伪随机合成器构造 (pseudorandom synthesizer)

- 伪随机合成器  $f(r, s)$  是满足如下性质的函数: 令  $r_1, \dots, r_n$  和  $s_1, \dots, s_m$  独立随机分布, 当输入  $(r, s)$  取遍  $(r_i, s_j)$  组合时, 输出伪随机.
- Naor 和 Reingold 证明了从  $\mathbb{Z}_p \times \mathbb{Z}_p$  映射到  $\mathbb{G}$  的函数  $f(r, s) = g^{rs}$  是基于 DDH 假设的伪随机合成器. 

#### 引理 4.1

如果 DDH 假设成立, 那么由  $\text{GenConcealMatrix}(n)$  生成的矩阵  $\mathbf{M} = g^{\mathbf{V}}$  在  $\mathbb{G}^{n \times (n+1)}$  上伪随机. 

**证明** 证明的过程分为两个步骤, 我们首先在一行上从左至右逐个列元素进行混合论证, 证明其与  $\mathbb{G}^{n+1}$  上的随机向量计算不可区分, 再利用该结论从上到下逐行进行混合论证, 从而证明隐藏矩阵  $\mathbf{M}$  在  $\mathbb{G}^{n \times (n+1)}$  上伪随机分布.

- 逐列论证: 令  $r \xleftarrow{R} \mathbb{Z}_p$ ,  $s \xleftarrow{R} \mathbb{Z}_p^n$ ,  $t \xleftarrow{R} \mathbb{Z}_p^n$ , 证明如下两个分布计算不可区分:

$$(g^s, g^r, y = g^{r \cdot s}) \approx_c (g^s, g^r, y = g^t)$$

证明的方法是设计如下的游戏序列进行混合论证:

$$\begin{aligned} \text{Hyb}_0 &: g^s \quad g^{r s_1} \quad \dots \quad g^{r s_n} \quad g^r \\ \text{Hyb}_1 &: g^s \quad g^{t_1} \quad \dots \quad g^{r s_n} \quad g^r \\ \text{Hyb}_j &: g^s \quad g^{t_1} \quad g^{t_j} \quad g^{r s_n} \quad g^r \\ \text{Hyb}_n &: g^s \quad g^{t_1} \quad \dots \quad g^{t_n} \quad g^r \end{aligned}$$

基于 DDH 假设, 可以证明任意两个相邻的游戏中定义的分布簇均计算不可区分, 利用 hybrid lemma 立刻可得:  $\text{Hyb}_0 \approx_c \text{Hyb}_1$ .

- 逐行论证: 基于上述结果, 我们再逐行变换, 每次将一行替换成  $\mathbb{G}^{n+1}$  上的随机向量, 再次利用 hybrid lemma 即可证明

$$(g^s, \mathbf{M}) \approx_c (g^s, U_{\mathbb{G}^{n \times (n+1)}}) \quad (4.1)$$

综上,  $\mathbf{M}$  在  $\mathbb{G}^{n \times (n+1)}$  上伪随机分布. □

#### 注记 4.6

公式 (4.1) 事实上证明了比引理更强的结果, 即在敌手观察到  $g^s$  的情形下,  $\mathbf{M}$  仍与  $\mathbb{G}^{n \times (n+1)}$  上随机矩阵计算不可区分. 在以上两个步骤的证明过程中, 横向的归约损失是  $n$ , 纵向的归约损失为  $n$ , 因此证明的总归约损失是  $n^2$ . 可以利用 DDH 类假设的随机自归约性质 (random self-reducibility) 将归约损失降为  $n$  (to be confirmed).

以下首先展示基于 DDH 假设的 LTDF 构造.

#### 构造 4.3 (DDH-based LTDF)

- Setup( $1^\kappa$ ): 运行  $\text{GenGroup}(1^\kappa) \rightarrow (\mathbb{G}, g, p)$ , 其中  $\mathbb{G}$  是一个阶为素数  $p$  的循环群, 生成元为  $g$ . 输出  $pp = (\mathbb{G}, g, p)$ .  $pp$  还包括了定义域  $X = \{0, 1\}^n$  和值域  $Y = \mathbb{G}$  的描述.
- GenInjective( $n$ ): 运行  $\text{GenConcealMatrix}(n) \rightarrow (g^{\mathbf{V}}, \mathbf{s})$ , 输出  $g^{\mathbf{Z}} = g^{\mathbf{V} + \mathbf{I}'}$  作为公钥  $ek$ , 其中  $\mathbf{I}' \in \mathbb{Z}_p^{n \times (n+1)}$  由  $n$  阶单位阵在最右侧补上全零列扩展得来 (即  $(\mathbf{e}_1, \dots, \mathbf{e}_n, \mathbf{0})$ ), 输出  $\mathbf{s}$  作为函数的陷门  $td$ .

$$g^{\mathbf{Z}} = \left( \begin{array}{cccc|c} g^{r_1 s_1 + 1} & g^{r_1 s_2} & \dots & g^{r_1 s_n} & g^{r_1} \\ g^{r_2 s_1} & g^{r_2 s_2 + 1} & \dots & g^{r_2 s_n} & g^{r_2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ g^{r_n s_1} & g^{r_n s_2} & \dots & g^{r_n s_n + 1} & g^{r_n} \end{array} \right)$$

- GenLossy( $n$ ):  $\text{GenConcealMatrix}(n) \rightarrow g^{\mathbf{V}}$ , 输出  $g^{\mathbf{Z}} = g^{\mathbf{V}}$  作为公钥  $ek$ , 陷门  $td$  为  $\perp$ .

$$g^{\mathbf{Z}} = \left( \begin{array}{cccc|c} g^{r_1 s_1} & g^{r_1 s_2} & \dots & g^{r_1 s_n} & g^{r_1} \\ g^{r_2 s_1} & g^{r_2 s_2} & \dots & g^{r_2 s_n} & g^{r_2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ g^{r_n s_1} & g^{r_n s_2} & \dots & g^{r_n s_n} & g^{r_n} \end{array} \right)$$

- Eval( $ek, \mathbf{x}$ ): 以  $ek = g^{\mathbf{Z}}$  和  $\mathbf{x} \in \{0, 1\}^n$  为输入, 计算  $\mathbf{y} \leftarrow g^{\mathbf{xZ}} \in \mathbb{G}^{n+1}$ .
- TdInv( $td, \mathbf{y}$ ): 解析  $td = \mathbf{s} = (s_1, \dots, s_n)$ , 对每个  $i \in [n]$ , 计算  $a_i = y_i / y_{n+1}^{s_i}$  并输出  $x_i \in \{0, 1\}$  s.t.

$$a_i = g^{x_i}.$$

**定理 4.4**

基于 DDH 假设, 上述构造是一族  $(n, \log p)$ -LTDF.

**证明** 单射可逆模式的正确性由算法  $\text{TdInv}$  的正确性保证. 在有损模式下, 所有输出  $\mathbf{y}$  都具有  $g^{c\mathbf{s}}$  的形式, 其中  $c = \langle \mathbf{x}, \mathbf{r} \rangle \in \mathbb{Z}_p$ . 向量  $\mathbf{s}$  被  $ek$  固定, 因此  $\text{Img}(f_{ek}) \leq p$ .

单射可逆模式和有损模式的计算不可区分性由  $\text{GenConcealMatrix}$  输出的伪随机性 (引理 4.1) 保证.  $\square$

下面展示如何基于 DDH 假设构造 ABO-LTDF.

**构造 4.4 (DDH-based ABO-LTDF)**

- $\text{Setup}(1^\kappa)$ : 运行  $\text{GenGroup}(1^\kappa) \rightarrow (\mathbb{G}, g, p)$ , 其中  $\mathbb{G}$  是一个阶为素数  $p$  的循环群, 生成元为  $g$ . 输出  $pp = (\mathbb{G}, g, p)$ .  $pp$  还包括定义域  $X = \{0, 1\}^n$ 、值域  $Y = \mathbb{G}$  和分支集合  $B = \mathbb{Z}_p$  的描述.
- $\text{Gen}(pp, b^*)$ : 运行  $\text{GenConcealMatrix}(n) \rightarrow (g^{\mathbf{V}}, \mathbf{s})$ , 输出  $g^{\mathbf{Z}} = g^{\mathbf{V} - b^* \mathbf{I}'}$  作为公钥  $ek$ , 其中  $\mathbf{I}' = (\mathbf{e}_1, \dots, \mathbf{e}_n, \mathbf{0}) \in \mathbb{Z}_p^{n \times (n+1)}$ , 输出  $(b^*, \mathbf{s})$  作为陷门  $td$ .
- $\text{Eval}(ek, b, \mathbf{x})$ : 以  $ek = g^{\mathbf{Z}}$  和  $\mathbf{x} \in \{0, 1\}^n$  为输入, 计算  $\mathbf{y} \leftarrow g^{\mathbf{x}(\mathbf{Z} + b(\mathbf{e}_1, \dots, \mathbf{e}_n, \mathbf{0}))} \in \mathbb{G}^{n+1}$ , 记为  $y \leftarrow f(ek, b, \mathbf{x})$  或  $y \leftarrow f_{ek, b}(\mathbf{x})$ .
- $\text{TdInv}(td, b, \mathbf{y})$ : 解析  $td$  为  $\mathbf{s} = (s_1, \dots, s_n)$ , 对每个  $i \in [n]$ , 计算  $a_i = y_i / y_{n+1}^{s_i}$  并输出  $x_i \in \{0, 1\}$  s.t.  $a_i = g^{(b-b^*)x_i}$ .

$$\begin{aligned} & \text{Gen}(pp, b^*) \rightarrow (ek, \mathbf{s}) \\ & \text{GenConcealMatrix}(n) = g^{\mathbf{V}} \\ & \mathbf{x} \in \mathbb{Z}_2^n \times \left( \begin{array}{cccc|c} g^{r_1 s_1} & g^{r_1 s_2} & \dots & g^{r_1 s_n} & g^{r_1} \\ g^{r_2 s_1} & g^{r_2 s_2} & \dots & g^{r_2 s_n} & g^{r_2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ g^{r_n s_1} & g^{r_n s_2} & \dots & g^{r_n s_n} & g^{r_n} \end{array} \right) \begin{array}{l} -b^*(\mathbf{e}_1, \dots, \mathbf{e}_n, \mathbf{0}) \\ +b(\mathbf{e}_1, \dots, \mathbf{e}_n, \mathbf{0}) \\ \rightarrow \mathbf{y} \in \mathbb{G}^{n+1} \end{array} \\ & \text{DDH} \Rightarrow \approx_c U_{\mathbb{G}^{n \times (n+1)}} \end{aligned}$$

**定理 4.5**

基于 DDH 假设, 上述构造是一族分支集合为  $B = \mathbb{Z}_p$  的  $(n, \log p)$ -ABO-TDF.

**证明** 容易验证, 当  $b \neq b^*$  时,  $\mathbf{V} + (b - b^*)\mathbf{I}'$  矩阵满秩,  $f_{ek, b}$  单射且可高效求逆; 当  $b = b^*$  时, 矩阵  $\mathbf{V} + (b - b^*)\mathbf{I}'$  的秩为 1,  $\text{Img}(f_{ek, b}) \leq p$ . 有损分支隐藏性由  $\text{GenConcealMatrix}$  输出的伪随机性 (引理 4.1) 保证.  $\square$

**注记 4.7**

为了确保求逆算法的高效性, 以上构造有两个重要的设定: (1) 首先在  $\text{ConcealMatrix}$  设置了辅助列  $(g^{r_1}, \dots, g^{r_n})^T$ , 便于计算出  $a_i = g^{x_i}$ ; (2) 从  $a_i$  中计算  $x_i$  需要求解离散对数, 因此定义域  $X$  设定为  $\mathbb{Z}_2^n$ , 其中 2 可以进一步放宽至  $\kappa^{O(1)}$  (关于  $\kappa$  的多项式规模), 以保证可以在多项式时间完成离散对数求解.

**扩展与深化**

注意到在公钥加密的选择密文安全定义中敌手对解密谕言机的访问权限是全除一的, 由此可以看出全除一有损陷门函数的应用局限于“全除一”类安全的密码方案设计. Hofheinz [Hof12] 引入了全除多有损陷门函数, 将有损分支的数量从 1 扩展到  $\text{poly}(\kappa)$ , 并展示了其在选择打开选择密文安全 (selective opening chosen-ciphertext security) 中的应用. 在有损陷门函数的应用中, 我们通常期望有损模式下函数丢失的信息尽可能的多, 即像集尽

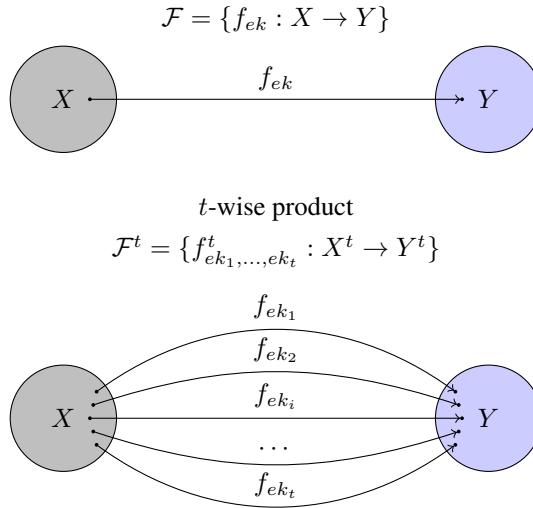
可能的小. 这是因为单射和有损模式的反差越大, 所蕴含的结果越强, 如更高的泄漏容忍能力、更紧的安全归约等. 但凡事有度, 物极必反, 在常规的一致归约 (universal reduction) 模型下, 有损模式的像集尺寸  $2^r$  不能过小, 至少是关于计算安全参数  $\kappa$  的超多项式规模, 否则 PPT 的敌手可以通过生日攻击有效的区分单射和有损模式. Zhandry [Zha16] 创造性的提出了极度有损函数 (ELF, extremely lossy functions). 在 ELF 中, 有损模式下函数的像集可以缩小至关于计算安全参数  $\kappa$  的多项式规模, 只要在指定 PPT 敌手的生日攻击能力之外即可. ELF 的有损模式之所以能够打破像集多项式界的关键在更为精细的个体归约 (individual reduction) 模型 [Den17] 下进行安全性证明. Zhandry 基于不可区分程序混淆给出了 ELF 的构造, 并展示了其强大的应用. 在无需求逆的应用场景中, 不仅不需要陷门, 甚至是单射的性质也可以弱化. 陈等 Chen-CT-RSA-2018 根据这一观察, 提出了规则有损函数 (RLF, regular lossy functions). 相比标准的 LTDF, RLF 将单射可逆模式放宽至规则有损, 即每个像的原像集合大小相同. 正是这一弱化, 使得 RLF 不仅有更加高效的具体构造, 也可由哈希证明系统通用构造得出, 并在抗泄漏密码学领域有着重要的应用.

### 4.1.3 相关积单向陷门函数

山重水复疑无路, 柳暗花明又一村.

— 宋·陆游《过山西村》

令  $\mathcal{F} = \{f_{ek} : X \rightarrow Y\}$  是一族单向陷门函数, 可以自然对  $\mathcal{F}$  进行  $t$  重延拓, 得到  $\mathcal{F}^t = \{f_{ek_1, \dots, ek_t}^t : X^t \rightarrow Y^t\}$ , 其中  $f_{ek_1, \dots, ek_t}^t(x_1, \dots, x_t) := (f_{ek_1}(x_1), \dots, f_{ek_t}(x_t))$ . 我们称  $\mathcal{F}^t$  为  $\mathcal{F}$  的  $t$  重积 ( $t$ -wise product).



#### 定理 4.6

如果  $\mathcal{F} = \{f_{ek}\}$  是一族单向函数, 那么它的  $t$  重积  $\mathcal{F}^t = \{f_{ek_1, \dots, ek_t}^t\}$  也是一族单向函数.

**证明** 证明的思路简述如下: 如果存在 PPT 的敌手  $\mathcal{A}$  打破  $\mathcal{F}^t$  的单向性, 那么其必然以不可忽略的优势对  $t$  个单向函数的实例  $f_{ek_i}(\cdot)$  求逆, 这显然与  $\mathcal{F}$  的单向性冲突, 因此得证.

#### 注记 4.8

上述定理在  $ek_1 = \dots = ek_t$  (即所有  $f_{ek_i}$  相同) 时仍然成立, 该情形恰好对应单向函数的单向性放大 (one-wayness amplification).

需要注意的是,  $f_{ek_1, \dots, ek_t}^t$  单向性成立的前提是各分量输入  $x_i$  独立随机采样, 而当各分量输入相关时, 单向性则未必成立, 这是因为多个像的分量交叉组合可能会泄漏原像的信息.

**构造 4.5 (反例构造)**

令  $\hat{f}_{ek} : X = \{0,1\}^n \rightarrow Y$  是一个单向函数, 构造一个新的函数  $f_{ek} : \{0,1\}^{2n} \rightarrow Y \parallel \{0,1\}^n$  如下:

$$f_{ek}(x_l \parallel x_r) := \hat{f}_{ek}(x_l) \parallel x_r$$

在上述构造中,  $f_{ek}$  以  $\hat{f}_{ek}$  为核, 因此如果  $\hat{f}_{ek}$  是单向的, 那么  $f_{ek}$  也是单向的. 考察 2 重积  $f_{ek_1, ek_2}^2$  在相关输入  $(x_1 = x_l \parallel x_r, x_2 = x_r \parallel x_l)$  下的行为:

$$f_{ek_1, ek_2}^2(x_1, x_2) := (f_{ek_1}(x_1), f_{ek_2}(x_2)) = \hat{f}_{ek_1}(x_l) \parallel x_r \parallel \hat{f}_{ek_2}(x_r) \parallel x_l$$

根据  $f_{ek}$  的设计,  $f_{ek_1, ek_2}^2$  的原像信息  $(x_1, x_2)$  可以从像中的  $(x_r, x_l)$  完全恢复出来, 因此在输入呈如上相关时并不满足单向性. 上述反例构造的精髓是设计具有特殊结构的单向函数.

反例 4.5 说明单向函数的  $t$  重积在输入相关时并不一定仍然单向. Alon 和 Rosen [RS09] 引入了相关积 (correlated products) 单向陷门函数, 定义如下: 要求函数的  $t$  重积在输入分量相关时仍然保持单向性

**定义 4.5 (相关积单向性)**

令  $\mathcal{F} : X \rightarrow Y$  是一族单向函数,  $\mathcal{C}_t$  是定义在  $X^t$  上的分布 (分量相关). 如果  $\mathcal{F}$  的  $t$  重积  $\mathcal{F}^t : X^t \rightarrow Y^t$  在  $\mathcal{C}_t$  相关积下仍然是单向的, (即对于任意 PPT 敌手  $\mathcal{A}$ , 其在如下的安全实验中优势是可忽略的)

$$\Pr \left[ \begin{array}{l} ek_i \leftarrow \text{Gen}(\kappa); \\ (x_1^*, \dots, x_t^*) \leftarrow \mathcal{C}_t; \\ y^* \leftarrow (f_{ek_1}(x_1^*), \dots, f_{ek_t}(x_t^*)); \\ x' \leftarrow \mathcal{A}(ek_1, \dots, ek_t, y^*); \end{array} \right]$$

则称  $\mathcal{F}$  是  $\mathcal{C}_t$  相关积安全的 (correlated-product secure). 该定义可以自然延拓到陷门函数场景.

在给出 CP-TDF 的定义后, 接下来需要研究的问题是分析什么样的  $\mathcal{F}$  在何种相关积下仍然单向. 本书中聚焦最为典型的一种  $\mathcal{C}_t$  相关积——均匀重复相关积  $\mathcal{U}_t$ , 即  $x_1 \stackrel{\mathcal{R}}{\leftarrow} X$  且  $x_1 = \dots = x_t$ . Rosen 和 Segev [RS09] 基于 LTDF 给出了 CP-TDF 的一个通用构造, 揭示了两者之间的联系.

**定理 4.7**

令  $\mathcal{F}$  是一族  $(n, \tau)$ -LTDF, 那么  $\mathcal{F}$  在相关积  $\mathcal{U}_t$  下仍然单向, 其中  $t \leq (n - \omega(\log \kappa)) / \tau$ .

**证明** 证明通过以下的游戏序列完成, 敌手在  $\text{Game}_i$  中成功的事件为  $S_i$ .

$\text{Game}_0$ : 对应真实的相关积单向性实验, 函数以单射模式运作

- $\mathcal{CH}$  独立运行  $\mathcal{F}.\text{GenInjective}(\kappa)$  算法  $t$  次, 生成  $ek = (ek_1, \dots, ek_t)$  并将其发送给  $\mathcal{A}$ .
- $\mathcal{CH}$  随机采样  $x^* \stackrel{\mathcal{R}}{\leftarrow} X$ , 计算  $y^* \leftarrow (f_{ek_1}(x^*), \dots, f_{ek_t}(x^*))$  并将  $y^*$  发送给  $\mathcal{A}$ .
- $\mathcal{A}$  输出  $x'$ , 当且仅当  $x' = x^*$  时成功.

根据定义, 我们有:

$$\text{Adv}_{\mathcal{A}}^{\text{Game}_0}(\kappa) = \Pr[S_0]$$

$\text{Game}_1$ : 与上一游戏相同, 区别在于函数切换到有损模式运作

- $\mathcal{CH}$  独立运行  $\mathcal{F}.\text{GenLossy}(\kappa)$  算法  $t$  次, 生成  $ek = (ek_1, \dots, ek_t)$  并将其发送给  $\mathcal{A}$ .

$$\text{Adv}_{\mathcal{A}}^{\text{Game}_1}(\kappa) = \Pr[S_1]$$

**断言 4.3**

基于 LTDF 的单射/有损模式不可区分性, 任意 PPT 敌手  $\mathcal{A}$  在  $\text{Game}_0$  和  $\text{Game}_1$  中的成功概率差可忽略.



**证明**  $\text{Game}_0$  和  $\text{Game}_1$  的差别在于  $(ek_1, \dots, ek_t)$  的生成模式. 基于 LTDF 的单射/有损模式不可区分性和 hybrid argument, 可以推出  $\text{Game}_0 \approx_c \text{Game}_1$ , 进而保证  $|\Pr[S_0] - \Pr[S_1]| \leq \text{negl}(\kappa)$ .

#### 断言 4.4

对于任意敌手  $\mathcal{A}$  (即使拥有无穷计算能力),  $\Pr[S_1] = \text{negl}(\lambda)$ .

**证明** 在  $\text{Game}_1$  中, 函数工作在有损模式, 因此像集的大小至多为  $2^{t\tau}$ , 由 chaining lemma 2.2 可知  $x^*$  的平均最小熵  $\tilde{H}_\infty(x^*|y^*) \geq n - t\tau$ . 根据定理前提条件中的参数选取, 有  $\tilde{H}_\infty(x^*|y^*) \geq \omega(\log \kappa)$ , 因此断言得证.  $\square$

综上, 我们有  $\Pr[S_0] \leq \text{negl}(\kappa)$ . 定理得证!  $\square$



**笔记** 追求简洁、消除冗余在科学和文学领域似乎都是真理. 然而, 正如知乎上一篇文章 [zhihu-essay] 所说: “尽管我们偏爱简洁, 但冗余让一切皆有可能”. 相关积单向函数的定义和构造就充分诠释了冗余的力量.

### 4.1.4 自适应单向陷门函数

他强由他强, 清风拂山冈; 他横由他横, 明月照大江; 他自狠来他自恶, 我自一口真气足.

— 达摩《九阳真经》


构造 4.2 仅具备 IND-CPA 安全性, 并不一定能够满足 IND-CCA 安全性. 这是因为底层的 TDF 可能具备诸如同态等优良的代数性质, 使得上层 PKE/KEM 方案具有可延展性. 从安全归约的角度分析, 归约算法无法对解密/解封装询问做出正确的应答. 基于以上分析, 一个自然的问题是: TDF 满足何种增强的性质才能够使得构造 4.2 满足 IND-CCA 安全性.

Kiltz, Mohassel 和 O'Neill [KMO10] 提出了自适应单向性 (adaptive one-wayness), 该性质要求 TDF 的单向性在敌手能够访问求逆谕言机的情况下仍然成立.

#### 定义 4.6 (自适应单向性)

令  $\mathcal{F}$  是一族陷门函数, 定义敌手  $\mathcal{A}$  的优势如下:

$$\Pr \left[ \begin{array}{l} pp \leftarrow \text{Setup}(\kappa); \\ (ek, td) \leftarrow \text{KeyGen}(pp); \\ x^* \xleftarrow{\mathcal{R}} X, y^* \leftarrow f_{ek}(x^*); \\ x' \leftarrow \mathcal{A}^{\mathcal{O}_{\text{inv}}}(ek, y^*); \end{array} \right]$$


其中  $\mathcal{O}_{\text{inv}}$  是求逆谕言机,  $\forall x \neq x^*, \mathcal{O}_{\text{inv}}(y) = \text{TdInv}(td, y)$ . 如果任意 PPT 敌手  $\mathcal{A}$  在上述安全试验中的优势均为  $\text{negl}(\kappa)$ , 那么则称  $\mathcal{F}$  是自适应单向的. 

为了方便在公钥加密场景中的应用, 引入自适应伪随机性如下.

#### 定义 4.7 (自适应伪随机性)

令  $\mathcal{F}$  是一族单向函数,  $\text{hc}$  是其 hardcore function. 定义敌手  $\mathcal{A}$  的优势如下:

$$\Pr \left[ \begin{array}{l} (ek, td) \leftarrow \mathcal{F}.\text{Gen}(\lambda); \\ x^* \xleftarrow{\mathcal{R}} X, y^* \leftarrow f_{ek}(x^*); \\ k_0^* \leftarrow \text{hc}(x^*), k_1^* \xleftarrow{\mathcal{R}} K, \beta \xleftarrow{\mathcal{R}} \{0, 1\}; \\ \beta' \leftarrow \mathcal{A}^{\mathcal{O}_{\text{inv}}}(ek, y^*, k_\beta^*); \end{array} \right] - \frac{1}{2}$$

其中  $\mathcal{O}_{\text{inv}}$  是求逆谕言机,  $\text{hc}$  是  $\mathcal{F}$  的 hardcore function. 如果任意 PPT 敌手  $\mathcal{A}$  在上述安全试验中的成功概率均为  $\text{negl}(\kappa)$ , 那么则称  $\text{hc}$  的是自适应伪随机的. 



**推论 4.1**

$\mathcal{F}$  的自适应单向性蕴含 hardcore function 的自适应伪随机性.



**证明** Goldreich-Levin 定理的证明可以平行推广到求逆谕言机  $\mathcal{O}_{\text{inv}}$  存在的情形下,  $\text{hc}(x^*)$  自适应伪随机性由  $x^*$  的自适应单向性保证.  $\square$

自适应单向陷门函数 (ATDF, adaptive TDF) 定义简洁, 威力强大, 将 ATDF 代入构造 4.2 中, 得到的 KEM 满足 IND-CCA 安全. 从安全归约的角度观察, ATDF 的自适应单向性是为 KEM 的 CCA 安全性量身定制的, 都是“全除一”类型的安全定义. 那么, 如何构造 ATDF 呢? 文献 [KMO10] 一方面基于实例独立 (instance-independent) 假设给出 ATDF 的具体构造, 一方面分别基于 LTDF 和 CP-TDF 给出了 ATDF 的两个通用构造.

以下我们聚焦 ATDF 的通用构造, 首先展示如何基于 LTDF 构造 ATDF. 构造的技术困难点在于 ATDF 的安全试验中挑战者  $\mathcal{CH}$  向敌手  $\mathcal{A}$  提供了“全除一”式解密谕言机  $\mathcal{O}_{\text{inv}}$ , 而 LTDF 的安全试验中并没有提供类似的谕言机访问接口. 因此, 构造的思路是通过引入精巧的结构完成解密谕言机  $\mathcal{O}_{\text{inv}}$  的模拟. 总体的思路如下:

- 令 ATDF 的像  $y$  形如  $(y_0, y_1)$ , 确保  $y_1$  由  $y_0$  唯一确定, 可行的设计是计算原像  $x$  的 LTDF 值作为  $y_0$ , 再以  $y_0$  为分支编号计算  $x$  的 ABO-TDF 值作为  $y_1$ .

$$y_0 \leftarrow f(\text{ek}_{\text{ldf}}, x), y_1 \leftarrow g(\text{ek}_{\text{abo}}, y_0, x)$$

- 上述设计利用 ABO-TDF 的相对分支标签的“全除一”求逆陷门嵌入了相对于像的“全除一”可逆结构.

**构造 4.6 (基于 LTDF 和 ABO-TDF 的 ATDF 构造)**

构造所需的组件是:

- $(n, \tau_1)$ -LTDF  $\mathcal{F} : X \rightarrow Y_1$ ;
- $(n, \tau_2)$ -ABO-TDF  $\mathcal{G} : X \rightarrow Y_2$  w.r.t.  $Y_1$  作为分支集合;

其中  $\log_2 |X| = n, \log_2 |Y_1| = m_1, \log_2 |Y_2| = m_2$ .

构造 ATDF  $: X \rightarrow Y_1 \times Y_2$  如下:

- **Setup**( $1^\kappa$ ): 以安全参数  $\kappa$  为输入, 计算  $pp_{\text{ldf}} \leftarrow \mathcal{F}.\text{Setup}(1^\kappa)$ ,  $pp_{\text{abo}} \leftarrow \mathcal{G}.\text{Setup}(1^\kappa)$ , 输出  $pp = (pp_{\text{ldf}}, pp_{\text{abo}})$ .
- **Gen**( $pp$ ): 以公开参数  $pp = (pp_{\text{ldf}}, pp_{\text{abo}})$  为输入, 计算  $(\text{ek}_{\text{ldf}}, \text{td}_{\text{ldf}}) \leftarrow \mathcal{F}.\text{GenInjective}(pp_{\text{ldf}})$ ,  $(\text{ek}_{\text{abo}}, \text{td}_{\text{abo}}) \leftarrow \mathcal{G}.\text{Gen}(pp_{\text{abo}}, 0^{m_1})$ , 输出求值公钥  $\text{ek} = (\text{ek}_{\text{ldf}}, \text{ek}_{\text{abo}})$  和陷门  $\text{td} = (\text{td}_{\text{ldf}}, \text{td}_{\text{abo}})$ .
- **Eval**( $\text{ek}, x$ ): 以求值公钥  $\text{ek} = (\text{ek}_{\text{ldf}}, \text{ek}_{\text{abo}})$  和  $x \in \{0, 1\}^n$  为输入, 计算  $y_1 \leftarrow f_{\text{ek}_{\text{ldf}}}(x)$ ,  $y_2 \leftarrow g_{\text{ek}_{\text{abo}}}(y_1, x)$ , 输出  $y = (y_1, y_2)$ .
- **TdInv**( $\text{td}, y$ ): 以陷门  $\text{td} = (\text{td}_{\text{ldf}}, \text{td}_{\text{abo}})$  和  $y = (y_1, y_2)$  为输入, 计算  $x \leftarrow \mathcal{F}.\text{TdInv}(\text{td}_{\text{ldf}}, y_1)$ , 验证  $y_2 \stackrel{?}{=} g_{\text{ek}_{\text{abo}}}(y_1, x)$ : 如果是输出  $x$ , 否则输出  $\perp$ .



上述构造的正确性显然成立. 安全性由如下定理保证:

**定理 4.8**

基于 LTDF 和 ABO-TDF 的安全性, 上述构造在  $n - \tau_1 - \tau_2 \geq \omega(\log \kappa)$  构成一族 ATDF.



**证明** 令  $(x^*, y^* = (y_1^*, y_2^*))$  为单向挑战实例, 其中  $x^*$  是原像,  $y^*$  是像. 证明的思路将像  $y^* = (y_1^*, y_2^*)$  的计算方式从单射无损模式逐步切换到有损模式, 最终在信息论意义下论证单向性.

**Game<sub>0</sub>**: 真实的 ATDF 单向性试验.  $\mathcal{CH}$  与  $\mathcal{A}$  交互如下:

- 初始化:  $\mathcal{CH}$  进行如下操作
  1. 运行  $pp_{\text{ldf}} \leftarrow \mathcal{F}.\text{Setup}(1^\kappa)$ ,  $pp_{\text{abo}} \leftarrow \mathcal{G}.\text{Setup}(1^\kappa)$ ;
  2. 计算  $(\text{ek}_{\text{ldf}}, \text{td}_{\text{ldf}}) \leftarrow \mathcal{F}.\text{GenInjective}(pp_{\text{ldf}})$ ,  $(\text{ek}_{\text{abo}}, \text{td}_{\text{abo}}) \leftarrow \mathcal{G}.\text{Gen}(pp_{\text{abo}}, 0^{m_1})$ ;
  3. 发送  $pp = (pp_{\text{ldf}}, pp_{\text{abo}})$  和  $\text{ek} = (\text{ek}_{\text{ldf}}, \text{ek}_{\text{abo}})$  给  $\mathcal{A}$ .

- 挑战:  $\mathcal{CH}$  随机选取  $x^* \leftarrow^R X$ , 计算  $y_1^* \leftarrow f_{ek_{\text{ldf}}}(x^*)$ ,  $y_2^* \leftarrow g_{ek_{\text{abo}}}(y_1^*, x^*)$ , 发送  $y^* = (y_1^*, y_2^*)$  给  $\mathcal{A}$ .
- 求逆询问: 当  $\mathcal{A}$  向  $\mathcal{O}_{\text{inv}}$  询问  $y = (y_1, y_2)$  的原像时,  $\mathcal{CH}$  分情况应答如下:
  - $y_1 = y_1^*$ : 直接返回  $\perp$ .
  - $y_1 \neq y_1^*$ : 首先计算  $x \leftarrow \mathcal{F}.\text{TdInv}(td_{\text{ldf}}, y_1)$ , 如果  $y_2 = g_{ek_{\text{abo}}}(y_1, x)$  则返回  $x$ , 否则返回  $\perp$ .

根据 ATDF 像的生成方式可知, 第一部分完全确定了第二部分, 当  $y_1 = y_1^*$  时, 如  $y_2 = y_2^*$  则  $\mathcal{A}$  的询问为禁讯点, 如  $y_2 \neq y_2^*$  则像的格式不正确. 基于以上分析,  $\mathcal{CH}$  在应答形如  $(y_1^*, y_2)$  的求逆询问时, 无须进一步检查第二部分  $y_2$ , 直接返回  $\perp$  即可保证应答的正确性.

**Game<sub>1</sub>:** 在 **Game<sub>0</sub>** 中  $\mathcal{CH}$  使用  $\mathcal{F}$  的陷门进行求逆, 因此  $\mathcal{F}$  必须工作在单射可逆模式. 为了将  $y_1^*$  的计算模式切换到有损模式, 需要利用  $\mathcal{G}$  的陷门进行求逆. 注意到在 **Game<sub>0</sub>** 中  $\mathcal{G}$  的“全除一”陷门根据预先设定的有损分支  $0^{m_1}$  生成, 因此必须先激活再使用, 因此 **Game<sub>1</sub>** 的设计目的是为激活做准备:

- $\mathcal{CH}$  在初始化阶段即随机采样  $x^* \leftarrow^R X$ , 并计算  $y_1^* \leftarrow f_{ek_{\text{ldf}}}(x^*)$ .

与 **Game<sub>0</sub>** 相比, **Game<sub>1</sub>** 仅将上述操作从挑战阶段提前至初始化阶段, 敌手的视图没有发生任何变化, 因此有:

$$\text{Game}_0 \equiv \text{Game}_1$$

**Game<sub>2</sub>:** 上一游戏已经做好激活  $\mathcal{G}$  陷门的准备, 因此在 **Game<sub>2</sub>** 中将预设的有损分支值由  $0^{m_1}$  替换为  $y_1^*$  完成激活:

- $(ek_{\text{abo}}, td_{\text{abo}}) \leftarrow \mathcal{G}.\text{Gen}(pp_{\text{abo}}, y_1^*)$

由 ABO-TDF 的有损分支隐藏性质, 可以得到:

$$\text{Game}_1 \approx_c \text{Game}_2$$

**Game<sub>3</sub>:** 使用  $\mathcal{G}$  的陷门  $td_{\text{abo}}$  应答求逆询问, 当  $\mathcal{A}$  发起询问  $y = (y_1, y_2)$  时,  $\mathcal{CH}$  分情形应答如下:

- $y_1 = y_1^*$ : 直接返回  $\perp$ .
- $y_1 \neq y_1^*$ : 计算  $x \leftarrow \mathcal{G}.\text{TdInv}(td_{\text{abo}}, y_1, y_2)$ , 如果  $y_1 = f_{ek_{\text{ldf}}}(x)$  则返回  $x$ , 否则返回  $\perp$ .

像的生成方式和  $\mathcal{G}$  求逆算法的正确性和保证了  $\mathcal{O}_{\text{inv}}$  应答的正确性, 因此有:

$$\text{Game}_2 \equiv \text{Game}_3$$

**Game<sub>4</sub>:** 将  $y_1^*$  的生成方式切换到有损模式

- $\mathcal{CH}$  在初始化阶段计算  $(ek_{\text{ldf}}, \perp) \leftarrow \mathcal{F}.\text{GenLossy}(pp_{\text{ldf}})$

LTDF 的单射/有损模式的计算不可区分性保证了

$$\text{Game}_3 \approx_c \text{Game}_4$$

#### 断言 4.5

对任意的敌手  $\mathcal{A}$  (即使拥有无穷的计算能力) 均有  $\text{Adv}_{\mathcal{A}}(\text{Game}_4) = \text{negl}(\kappa)$ .



**证明** 在 **Game<sub>4</sub>** 中, 函数  $f_{ek_{\text{ldf}}}(\cdot)$  有损且像集大小至多为  $2^{\tau_1}$ , 函数  $g_{ek_{\text{abo}}}(y_1^*, \cdot)$  有损且像集大小至多为  $2^{\tau_2}$ . 因此  $y_1^*$  和  $y_2^*$  均在信息论意义下损失了原像  $x^*$  的信息, 在敌手  $\mathcal{A}$  的视图中,  $x^*$  的平均最小熵为  $\tilde{H}_{\infty}(x^* | (y_1^*, y_2^*)) \geq H_{\infty}(x^*) - \tau_1 - \tau_2 = n - \tau_1 - \tau_2 \geq \omega(\log \kappa)$ . 从而对于任意敌手  $\mathcal{A}$  均有:

$$\text{Adv}_{\mathcal{A}}(\text{Game}_4) = \text{negl}(\kappa)$$

断言得证! □

综上, 定理得证! □

## 注记 4.9

上述构造的设计思想值得读者反复拆解, 体会其精妙之处. 上述 ATDF 构造在形式上与 Naor-Yung 的双钥加密有异曲同工之处: 分别使用  $f_{ek_{\text{tdf}}}(\cdot)$  和  $g_{ek_{\text{abo}}}(\cdot, \cdot)$  两个函数计算原像的函数值作为像. 一个自然的想法是上述构造显得冗余, 是否仅用 ABO-TDF 即可呢? 答案是否定的, 如果仅依赖 ABO-TDF 构造 ATDF, 需要满足以下四点:

- 求值分支可由输入公开确定计算得出, 以确保 ATDF 是公开可计算函数.
- 像所对应的求值分支可由像中计算得出, 以确保 ATDF 的求逆算法可以基于 ABO-TDF 的求逆算法设计.
- 在安全归约中势必需要将 ATDF 的单向性建立在 ABO-TDF 的信息有损性上, 也即  $y^*$  是  $x^*$  在有损分支的求值.

上述三点潜在要求 ATDF 的像包含两个部分, 一部分是原像对应的分支值, 一部分是 ABO-TDF 在该分支值下的像, 这使得在安全证明时存在如下两个障碍:

1. 分支值泄漏原像的多少信息难以确定
2. 敌手可以从挑战的像中计算出有损分支值, 从而可以发起关于有损分支的求逆询问, 而归约算法无法应答

通过上述的拆解分析, 便可看出 ATDF 设计的必然性. 引入 LTDF 并将分支值设定为原像的 LTDF 值有三重作用:

- LTDF 的陷门确保了 ATDF 构造存在功能完备的陷门.
- 可将分支值泄漏的关于原像信息量控制在指定范围.
- 分支值完全确定了像, 从而使得 ABO-TDF 的陷门在归约证明中可用于模拟求逆预言机  $\mathcal{O}_{\text{inv}}$ .

LTDF+ABO-TDF  $\Rightarrow$  ATDF 的设计思路有如二级运载火箭, 第一级运载火箭 (LTDF) 在完成推动后从单射切换到有损模式, 同时激活第二级运载火箭 (ABO-TDF).

我们再展示如何基于 CP-TDF 构造 ATDF. 构造的难点是在归约证明中归约算法如何在不掌握全部 CP-TDF 实例陷门的情况下正确模拟  $\mathcal{O}_{\text{inv}}$ . 大体的设计思路和以上基于有损陷门函数构造 LTDF 相似, 通过多重求值引入冗余结构, 从而使得归约算法在掌握部分 CP-TDF 实例陷门时能够正确应答求逆询问.

- 设计像  $y$  形如  $(y_0, y_1, \dots, y_n)$ , 确保  $y_0$  能够惟一确定  $(y_1, \dots, y_n)$ 
  - 令  $y_0$  是原像的 CP-TDF 函数值, 目的是确保  $y_0$  不会破坏最终 ATDF 函数的单向性
  - 令  $(y_1, \dots, y_n)$  是关于原像  $x$  的  $|y_0| = n$  重冗余函数求值
- 嵌入“全除一”求逆结构
  - 对  $y_0^*$  进行比特分解: 归约算法使用 Dolev-Dwork-Naor(DDN) 类技术逐比特嵌入对应的陷门, 使得对于点  $y = (y_0, y_1, \dots, y_n)$  处的求逆询问:
    1.  $y_0 = y_0^*$ : 归约算法可根据  $\mathcal{O}_{\text{inv}}$  的定义直接拒绝, 返回  $\perp$
    2.  $y_0 \neq y_0^*$ :  $\mathcal{R}$  可至少寻找到一个可用陷门用于应答  $\mathcal{O}_{\text{inv}}$ .

## 构造 4.7 (基于 CP-TDF 的 ATDF)

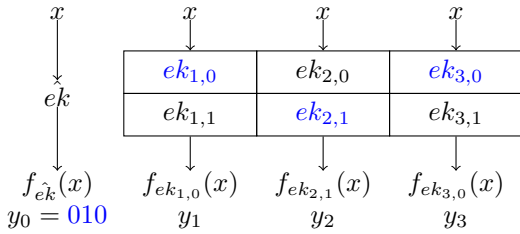
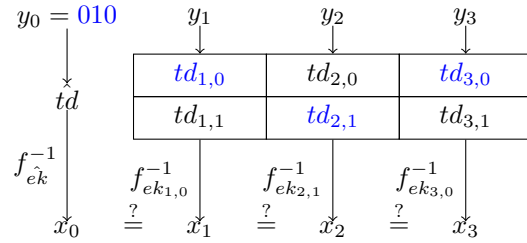
构造所需组件: 单射 CP-TDF  $\mathcal{F}: X \rightarrow \{0, 1\}^n$

构造 ATDF:  $X \rightarrow \{0, 1\}^{n(n+1)}$  如下:

- $\text{Setup}(1^\kappa)$ : 运行  $pp \leftarrow \mathcal{F}.\text{Setup}(1^\kappa)$ , 输出  $pp$  作为公开参数.
- $\text{KeyGen}(pp)$ : 以公开参数  $pp$  为输入
  1. 计算  $(\hat{ek}, \hat{td}) \leftarrow \mathcal{F}.\text{KeyGen}(\lambda)$ ;
  2. 对于  $b \in \{0, 1\}$  和  $i \in [n]$ , 计算  $(ek_{i,b}, td_{i,b}) \leftarrow \mathcal{F}.\text{KeyGen}(\lambda)$ ;
  3. 输出  $(\hat{ek}, (ek_{i,0}, ek_{i,1}), \dots, (ek_{n,0}, ek_{n,1}))$  作为求值公钥, 输出  $(\hat{td}, (td_{i,0}, td_{i,1}), \dots, (td_{n,0}, td_{n,1}))$  作为求逆陷门.

- $\text{Eval}(ek, x)$ : 以求值公钥  $ek = \hat{ek} \parallel (ek_{1,0}, ek_{1,1}) \dots (ek_{n,0}, ek_{n,1})$  和原像  $x$  为输入, 计算
  1. 计算  $y_0 \leftarrow f_{\hat{ek}}(x)$ ;
  2. 令  $b_i \leftarrow y_0[i]$ , 对  $i \in [n]$  计算  $y_i \leftarrow f_{ek_{i,b_i}}(x)$ ;
  3. 输出  $y = y_0 \parallel y_1 \parallel \dots \parallel y_n$ .
- $\text{TdInv}(td, y)$ : 以陷门  $td = (\hat{td}, \{td_{i,0}, td_{i,1}\}_{i \in [n]})$  和像  $y = y_0 \parallel y_1 \parallel \dots \parallel y_n$  为输入:
  1. 计算  $x_0 \leftarrow \mathcal{F}.\text{TdInv}(\hat{td}, y_0)$ ;
  2. 令  $b_i \leftarrow y_0[i]$ , 对所有  $i \in [n]$  计算  $x_i \leftarrow \mathcal{F}.\text{TdInv}(td_{i,b_i}, y_i)$ ;
  3. 检查  $x_i = x_0$  是否对于  $i \in [n]$  均成立, 若是则输出  $x_0$ , 否则输出  $\perp$ .



$$\begin{matrix} \hat{ek} \\ \hat{td} \end{matrix} \begin{array}{|c|c|c|} \hline ek_{1,0} & ek_{2,0} & ek_{3,0} \\ \hline td_{1,0} & td_{2,0} & td_{3,0} \\ \hline ek_{1,1} & ek_{2,1} & ek_{3,1} \\ \hline td_{1,1} & td_{2,1} & td_{3,1} \\ \hline \end{array}$$
图 4.2:  $n = 3$  时的求值公钥和求逆陷门图示图 4.3:  $n = 3, y = 010$  时的求值图示图 4.4:  $n = 3, y = 010$  时的求逆图示

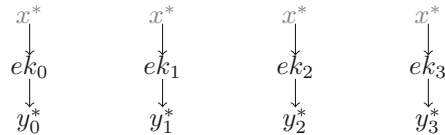
上述 ATDF 构造的正确性显然. 构造中, 函数的像  $y = y_0 \parallel y_1 \parallel \dots \parallel y_n$  是对原像的  $n+1$  重求值, 其中  $y_0$  确定了使用哪些求值公钥  $ek_{i,b}$  计算  $y_i$ , 因此当底层的 CP-TDF 是单射函数时,  $y_0$  可惟一确定  $y_1, \dots, y_n$ . 下面的定理就是利用上述结构特性模拟求逆预言机  $\mathcal{O}_{\text{inv}}$ .

#### 定理 4.9

如果  $\mathcal{F}$  是一族相对于  $\mathcal{U}_t$  安全的 CP-TDF, 那么上述构造一族 A 自适应单向陷门函数.

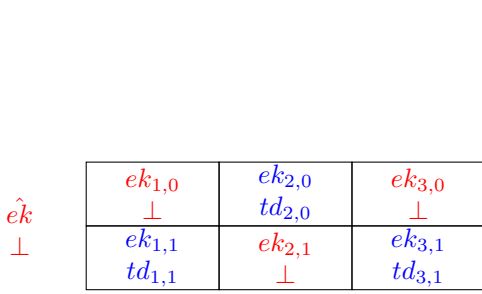
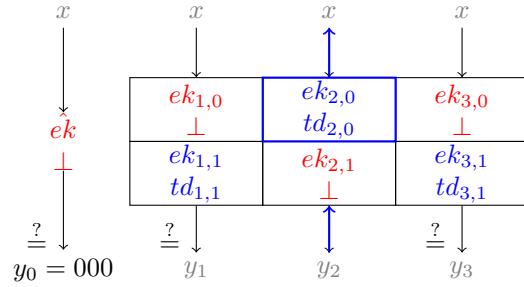


**证明** 使用反证法通过单一游戏完成归约证明. 假设存在 PPT 的敌手  $\mathcal{A}$  能以不可忽略的优势打破 ATDF 的自适应单向性, 那么可以黑盒调用  $\mathcal{A}$  的能力构造 PPT 的  $\mathcal{B}$  打破 CP-TDF 相对于  $\mathcal{U}_{n+1}$  的单向性.  $\mathcal{B}$  的 CP-TDF 挑战是公开参数  $pp$ 、求值公钥  $(ek_0, ek_1, \dots, ek_n)$  和像  $y^* = (y_0^*, y_1^*, \dots, y_n^*)$ , 其中  $y_i^* \leftarrow f_{ek_i}(x^*)$ ,  $x^* \xleftarrow{\mathcal{R}} X$ .  $\mathcal{B}$  并不知晓  $x^*$ , 其攻击目标是求解  $x^*$ .

图 4.5:  $n = 3$  时  $\mathcal{B}$  的 CP-TDF 挑战实例

令  $b_i^*$  是  $y_0^*$  的第  $i$  比特,  $\mathcal{B}$  (扮演挑战者) 与  $\mathcal{A}$  在 ATDF 的自适应单向性游戏中交互如下:

- 初始化:  $\mathcal{B}$  将 CP-TDF 的  $pp$  设为 ATDF 的公开参数, 设定  $\hat{ek} := ek_0$ , 对  $i \in [n]$  设定  $ek_{i,b_i^*} := ek_i$ , 计算  $(ek_{i,1-b_i^*}, td_{i,1-b_i^*}) \leftarrow \mathcal{F}.\text{KeyGen}(\kappa)$ .
- 挑战阶段:  $\mathcal{B}$  发送  $(y_0^*, y_1^*, \dots, y_n^*)$  给  $\mathcal{A}$  作为挑战.
- 求逆询问:  $\mathcal{A}$  向  $\mathcal{B}$  发起求逆询问  $y = (y_0, y_1, \dots, y_n)$ ,  $\mathcal{B}$  分情况应答如下:
  1.  $y_0 = y_0^*$ : 直接返回  $\perp$ , 应答的正确性由以下两种细分情况保证:

图 4.6:  $y_0^* = 010$  时生成求值公钥和求逆陷门的过程图示图 4.7:  $y_0 = 000$  时的求逆过程图示

- 对于所有的  $i \in [n]$  均有  $y_i = y_i^*$ : 询问为禁询点, 因此根据  $\mathcal{O}_{\text{inv}}$  的定义需返回  $\perp$ .
  - 对于某个  $i \in [n]$  使得  $y_i \neq y_i^*$ :  $\mathcal{F}$  的单射性质和像的生成方式保证了像的首项  $y_0$  确定了其余  $n$  项  $y_1, \dots, y_n$ .
2.  $y_0 \neq y_0^*$ : 必然存在  $\exists j \in [n]$  s.t.  $b_j \neq b_j^*$  且  $y_j = f_{ek_j, b_j}(x)$ , 其中  $x$  是未知原像. 此时,  $\mathcal{B}$  拥有关于  $ek_{j, b_j}$  的求逆陷门  $td_{j, b_j}$ ,  $\mathcal{B}$  可计算  $x \leftarrow f_{ek_j, b_j}^{-1}(y_j)$
- 如果  $y_0 = f_{ek_0}$  且  $y_i = f_{ek_i, b_i}(x)$  对其余所有  $i \neq j$  也均成立, 那么返回  $x$ , 否则返回  $\perp$ .
- 求解:  $\mathcal{A}$  输出  $x$  作为 ATDF 的挑战应答,  $\mathcal{B}$  将  $x$  转发给 CP-TDF 的挑战者.

容易验证,  $\mathcal{B}$  的优势与  $\mathcal{A}$  的优势相同. 定理得证! □

#### 注记 4.10 (优化)

以上 ATDF 构造的像  $(y_0, y_1, \dots, y_n)$  包含了对原像的  $(n+1)$  重 CP-TDF 求值:

- $y_0$  构造中起到的作用求值的公钥选择向量, 在归约证明中起到的作用是“全除一”求逆陷门的激活扳机 (trigger), 当  $y_0 \neq y_0^*$  时即可激活求逆陷门.

$y_0$  的编码长度决定了像的冗余重数. 能否缩减  $|y_0|$  以提高效率呢? 答案是肯定的, 可以使用密码组件进行值域扩张 (domain extension) 的通用技术, 使用  $y_0$  的抗碰撞哈希值代替  $y_0$ . 在上述构造中, 我们贴合 ATDF 的安全定义进行更为精细的处理, 使用 TCRHF (target collision resistant hash function) 代替 CRHF. 具体的, 令  $\text{TCR} : \{0, 1\}^n \rightarrow \{0, 1\}^m$ , 使用  $\text{TCR}(y_0)$  代替  $y_0$  作为公钥选择向量和陷门激活扳机. 从而利用 TCR 压缩的性质将像的重数从  $1+n$  缩减到  $1+m$ . 安全论证仍然成立, 这是因为 TCR 的抗碰撞性质保证了在计算意义下:

$$y_0 \neq y_0^* \iff \text{TCR}(y_0) \neq \text{TCR}(y_0^*)$$

类似的优化技术同样可以用于  $\text{LTDF} + \text{ABO-TDF} \Rightarrow \text{ATDF}$  的构造中: 可以使用  $y_0$  的 TCR 哈希值代替  $y_0$  作为分支值. 这样处理的好处是增加分支集合选择的灵活性. 💧

📌 **笔记**  $\text{LTDF} + \text{ABO-TDF} \Rightarrow \text{ATDF}$  与  $\text{CP-TDF} \Rightarrow \text{ATDF}$  的构造分别与 Naor-Yung 范式 [NY90] 和 Dolev-Dwork-Naor 范式 [DDN91] 在思想上极为相似, 总体思路都是通过冗余的结构来保证求逆谕言机的完美模拟.

## 自适应单向陷门关系

将 ATDF 中的确定性函数泛化为可公开高效验证的二元关系可得到自适应单向陷门关系 (ATDR, adaptive trapdoor relation).

- 确定性函数  $\leadsto$  概率关系
- 可高效计算  $\leadsto$  可高效采样

#### 定义 4.8 (单向陷门关系)

一族单向陷门关系包含以下算法:

- $\text{Setup}(\kappa)$ : 以安全参数  $\kappa$  为输入, 输出公开参数  $pp = (X, Y, EK, TD, R)$ , 其中  $R = \{R_{ek} : X \times$



$Y\}_{ek \in EK}$  是定义在  $X \times Y$  上由  $ek$  索引的一族二元单向关系.

- **KeyGen**( $pp$ ): 以公开参数  $pp$  为输入, 输出公钥  $ek$  和陷门  $td$ .
- **Sample**( $ek$ ): 输出二元关系的一个随机采样  $(x, y) \xleftarrow{R} R_{ek}$ .
- **TdInv**( $td, y$ ): 以  $td$  和  $y \in Y$  为输入, 输出  $x \in X \cup \perp$ .



**正确性:**  $\forall (ek, td) \leftarrow \text{KeyGen}(pp), \forall (x, y) \leftarrow \text{Sample}(ek)$ , 总有  $(\text{TdInv}(td, y), y) \in R_{ek}$ .

我们可以将函数的单射性质平行推广至二元关系的场景下: 如果  $\forall (x_1, y_1), (x_2, y_2) \in R_{ek}$  均有  $x_1 \neq x_2 \Rightarrow y_1 \neq y_2$ , 即  $y$  惟一确定了  $x$ , 那么则称二元关系满足单射性.



**笔记** **Sample** 是概率算法, 因此当  $y_1 \neq y_2$  时, 存在  $x_1 = x_2$  的可能.

#### 定义 4.9 (自适应单向性)

令  $R$  是一族二元关系, 定义敌手  $\mathcal{A}$  的优势如下:

$$\Pr \left[ \begin{array}{l} (x', y^*) \in R_{ek} : \\ \begin{array}{l} pp \leftarrow \text{Setup}(\kappa); \\ (ek, td) \leftarrow \text{KeyGen}(pp); \\ (x^*, y^*) \leftarrow \text{Sample}(ek); \\ x' \leftarrow \mathcal{A}^{\mathcal{O}_{\text{inv}}}(ek, y^*); \end{array} \end{array} \right]$$

其中  $\mathcal{O}_{\text{inv}}$  是求逆预言机,  $\forall x \neq x^*, \mathcal{O}_{\text{inv}}(y) = \text{TdInv}(td, y)$ . 如果任意 PPT 敌手  $\mathcal{A}$  在上述安全试验中的优势均为  $\text{negl}(\kappa)$ , 那么则称  $R$  是自适应单向的.



ATDR 是 ATDF 的弱化, 弱化允许我们可以给出更加高效灵活的设计, 同时不严重降低可用性. 在给出 ATDR 的构造之前, 我们首先回顾基于 CP-TDF 的 ATDF 构造. 构造的关键之处是将像  $y$  设计为  $y_0$  和  $(y_1, \dots, y_n)$  两部分, 其中  $y_0$  设定为  $f_{ek}(x)$ , 通过单射性完美绑定了  $(y_1, \dots, y_n)$ , 同时在归约证明中起到了“全除一”陷门触发器的作用: 当目标不再是构造确定性单向函数而是概率二元关系时, 我们有着更加灵活的选择: 使用一次性签名 (OTS, one-time signature) 的验证公钥作为  $(y_1, \dots, y_n)$  的求值选择器和求逆陷门触发器.

#### 构造 4.8 (基于 CP-TDF 和 OTS 的 ATDR 构造)

**构造组件:** 单射 CP-TDF  $\mathcal{F}: X \rightarrow Y$  和 strong OTS (令  $|vk| = \{0, 1\}^n$ , 签名空间为  $\Sigma$ );

**构造目标:** ATDR  $X \rightarrow VK \times Y^n \times \Sigma$

- **Setup**( $1^\kappa$ ): 运行  $pp_{\text{cptdf}} \leftarrow \mathcal{F}.\text{Setup}(1^\kappa)$ ,  $pp_{\text{ots}} \leftarrow \text{OTS}.\text{Setup}(1^\kappa)$ , 输出  $pp = (pp_{\text{cptdf}}, pp_{\text{ots}})$ .
- **KeyGen**( $pp$ ): 以  $pp = (pp_{\text{cptdf}}, pp_{\text{ots}})$  为输入, 对  $b \in \{0, 1\}$  和  $i \in [n]$  运行  $(ek_{i,b}, td_{i,b}) \leftarrow \mathcal{F}.\text{KeyGen}(pp_{\text{cptdf}})$  输出  $ek = ((ek_{1,0}, ek_{1,1}), \dots, (ek_{n,0}, ek_{n,1}))$ ,  $td = ((td_{1,0}, td_{1,1}), \dots, (td_{n,0}, td_{n,1}))$ .
- **Sample**( $ek$ ): 以  $ek = (ek_{1,0}, ek_{1,1}) \dots (ek_{n,0}, ek_{n,1})$  为输入, 采样如下:
  1. 生成  $(vk, sk) \leftarrow \text{OTS}.\text{KeyGen}(pp_{\text{ots}})$ ;
  2. 随机选择  $x \in X$ , 对  $i \in [n]$  计算  $y_i \leftarrow f_{ek_{i,b_i}}(x)$ , 其中  $b_i \leftarrow vk[i]$ ;
  3. 计算  $\sigma \leftarrow \text{OTS}.\text{Sign}(sk, y_1 || \dots || y_n)$ ;
 输出  $y = (vk, y_1 || \dots || y_n, \sigma)$ .
- **TdInv**( $td, y$ ): 以  $td = (\{td_{i,0}, td_{i,1}\}_{i \in [n]})$  和  $y = (vk, y_1 || \dots || y_n, \sigma)$  为输入, 求逆如下:
  1. 检查  $\text{OTS}.\text{Verify}(vk, y_1 || \dots || y_n, \sigma) \stackrel{?}{=} 1$ , 如果签名无效则返回  $\perp$ ;
  2. 对所有  $i \in [n]$  计算  $x_i \leftarrow \mathcal{F}.\text{TdInv}(td_{i,b_i}, y_i)$ , 其中  $b_i = vk[i]$ .
  3. 如果对所有  $i \in [n]$  均有  $x_i = x_1$  则返回  $x_1$ , 否则返回  $\perp$ .

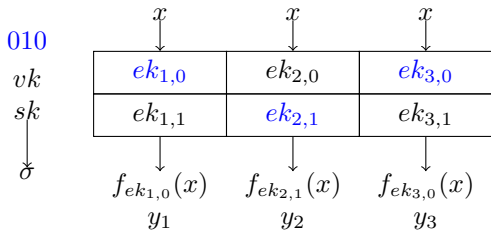
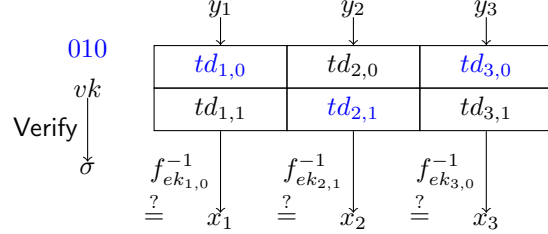


构造的正确性显然, 构造的以下三个特性使得归约算法能够成功模拟  $\mathcal{O}_{\text{inv}}$ .

- $R_{ek}$  是单射的并且  $y_1 || \dots || y_n$  是对原像  $x$  的  $n$  重冗余求值.

$$|vk| = 3$$

$ek_{1,0}$	$ek_{2,0}$	$ek_{3,0}$
$td_{1,0}$	$td_{2,0}$	$td_{3,0}$
$ek_{1,1}$	$ek_{2,1}$	$ek_{3,1}$
$td_{1,1}$	$td_{2,1}$	$td_{3,1}$

图 4.8:  $|vk| = 3$  时的求值公钥和求逆陷门生成图示图 4.9:  $vk = 010$  时的采样过程图 4.10:  $vk = 010$  时的求逆过程

- $vk$  是求值公钥的选择比特向量.
- 利用 OTS 的 sEUF-CMA 安全性,  $vk$  在计算意义下绑定了  $(y_1, \dots, y_n)$ .

**定理 4.10**

如果 OTS 是 sEUF-CMA 安全的, 并且  $\mathcal{F}$  是  $\mathcal{U}_n$  相关积单向的, 那么上述二元关系的构造满足自适应单向性.

**证明** 证明通过以下游戏序列完成.

**Game<sub>0</sub>**: 对应真实的 ATDR 自适应单向性安全试验. 令  $y^* = (vk^*, y_1^* || \dots || y_n^*, \sigma^*)$  是挑战的像.

**Game<sub>1</sub>**: 与 **Game<sub>0</sub>** 相同, 唯一的区别是挑战者对于求逆询问  $y = (vk^*, y_1 || \dots || y_n, \sigma)$  直接返回  $\perp$ . 应答的合理性分情况解释如下:

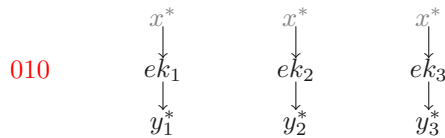
1.  $(y_1 || \dots || y_n, \sigma) = (y_1^* || \dots || y_n^*, \sigma^*)$ : 禁询点
2.  $(y_1 || \dots || y_n, \sigma) \neq (y_1^* || \dots || y_n^*, \sigma^*)$ : 构成 OTS 的存在性伪造

记敌手发起第二种类型求逆询问的事件为  $F$ , 那么利用 **Difference Lemma** 可以证明  $|\Pr[S_1] - \Pr[S_0]| \leq \Pr[F]$ , 而基于 OTS 的 sEUF-CMA 安全性, 可以推出  $\Pr[F] \leq \text{negl}(\kappa)$ , 从而  $|\Pr[S_1] - \Pr[S_0]| \leq \text{negl}(\kappa)$ .

**断言 4.6**

如果  $\mathcal{F}$  是  $\mathcal{U}_t$  相关积安全的, 那么对于任意的 PPT 敌手均有  $\Pr[S_1] = \text{negl}(\kappa)$ .

**证明** 论证通过单一归约完成. 假设存在 PPT 的敌手  $\mathcal{A}$  在 **Game<sub>1</sub>** 中的优势不可忽略, 那么尝试构造 PPT 算法  $\mathcal{B}$ , 通过黑盒调用  $\mathcal{A}$  的能力打破 CP-TDF 相对  $\mathcal{U}_n$  的相关积单向性.  $\mathcal{B}$  的 CP-TDF 挑战是公开参数  $pp_{\text{cpdf}}$ , 求值公钥  $(ek_1, \dots, ek_n)$  和像  $(y_1^*, \dots, y_n^*)$ , 其中  $y_i^* \leftarrow f_{ek_i}(x^*)$ ,  $x^* \xleftarrow{R} X$ .  $\mathcal{B}$  并不知晓  $x^*$ , 其攻击目标是求解  $x^*$ .

图 4.11:  $n = 3$  时  $\mathcal{B}$  的 CP-TDF 挑战实例

$\mathcal{B}$ (扮演挑战者) 与  $\mathcal{A}$  在 **Game<sub>1</sub>** 中交互如下:

- 初始化:  $\mathcal{B}$  运行  $pp_{\text{ots}} \leftarrow \text{OTS.Setup}(1^\kappa)$ , 生成  $(vk^*, sk^*) \leftarrow \text{OTS.KeyGen}(pp_{\text{ots}})$ . 令  $b_i^*$  是  $vk^*$  的第  $i$  比特,  $\mathcal{B}$  进行如下操作:
  1. 对  $i \in [n]$  设定  $ek_{i,b_i^*} := ek_i$ .



2. 对  $i \in [v]$  计算  $(ek_{i,1-b_i^*}, td_{i,1-b_i^*}) \leftarrow \mathcal{F}.\text{KeyGen}(pp_{\text{cptdf}})$ .  
 $\mathcal{B}$  发送  $pp = (pp_{\text{cptdf}}, pp_{\text{ots}})$  和  $ek = (ek_{1,0}, ek_{1,1}, \dots, ek_{n,0}, ek_{n,1})$  给  $\mathcal{A}$ .

$vk^*$ $sk^*$	$ek_{1,0}$ $\perp$	$ek_{2,0}$ $td_{2,0}$	$ek_{3,0}$ $\perp$
	$ek_{1,1}$ $td_{1,1}$	$ek_{2,1}$ $\perp$	$ek_{3,1}$ $td_{3,1}$

图 4.12:  $|vk| = 010$  时归约算法设定求值公钥和求逆陷门的过程图示

- 挑战:  $\mathcal{B}$  计算  $\sigma^* \leftarrow \text{OTS}.\text{Sign}(sk^*, (y_1^*, \dots, y_n^*))$ , 发送  $(vk^*, y_1^*, \dots, y_n^*, \sigma^*)$  给  $\mathcal{A}$  作为挑战.
  - 求逆询问: 对于求逆询问  $y = (vk, y_1 || \dots || y_v, \sigma)$ ,  $\mathcal{B}$  应答如下:
    - $vk = vk^*$ : 直接返回  $\perp$ .
    - $vk \neq vk^*$ : 必然存在  $\exists j \in [n]$  s.t.  $b_j \neq b_j^*$  且  $y_j = f_{ek_{j,b_j}}(x)$ , 其中  $x$  是未知原像. 此时,  $\mathcal{B}$  拥有关于  $ek_{j,b_j}$  的求逆陷门  $td_{j,b_j}$ ,  $\mathcal{B}$  可计算  $x \leftarrow f_{ek_{j,b_j}}^{-1}(y_j)$ 
      - 如果  $y_i = f_{ek_{i,b_i}}(x)$  对所有的  $i \neq j$  也均成立, 那么返回  $x$ , 否则返回  $\perp$ .
- 由  $\mathcal{F}$  的单射性可知,  $\mathcal{B}$  完美的模拟了  $\text{Game}_1$  中的  $\mathcal{O}_{\text{inv}}$  应答.

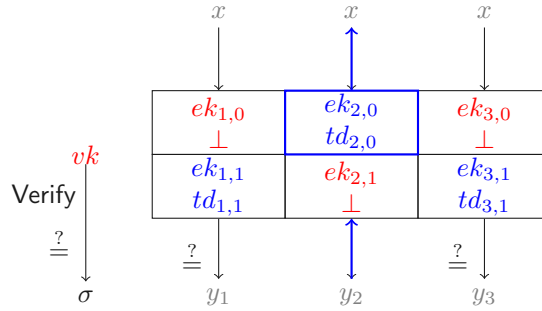


图 4.13:  $vk = 010$  时归约算法求逆过程图示

- 求解:  $\mathcal{A}$  输出  $x$  作为  $\text{Game}_1$  中 ATDF 的挑战应答,  $\mathcal{B}$  将  $x$  转发给 CP-TDF 的挑战者.
- 容易验证,  $\mathcal{B}$  的优势与  $\mathcal{A}$  的优势相同. 断言得证. □
- 综上, 定理得证! □

## 小结

Rosen 和 Regev [RS09] 证明了 CP-TDF 与 LTDF 之间存在黑盒分离, Kiltz、Mohassel 和 O'Neill [KMO10] 证明了 ATDF 与 CP-TDF 之间也存在黑盒分离. 因此, 在黑盒的意义下, ATDF 和 ATDR 是目前单向函数类中构造 CCA-KEM 所需的最弱组件.

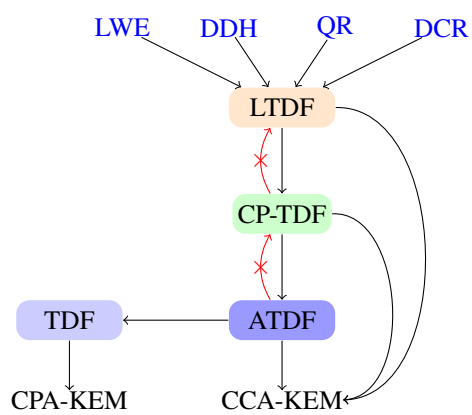


图 4.14: 各类单向函数之间的蕴含关系

## 4.2 哈希证明系统类

一阴一阳之谓道，继之者善也，成之者性也。

— 《易经·系辞上》

1998 年, Cramer 和 Shoup [CS98] 基于判定性 Diffie-Hellman 问题构造出首个标准模型下高效的公钥加密方案, 成为 CS98-PKE. 2002 年, Cramer 和 Shoup [CS02] 再度合作, 提出了哈希证明系统 (HPS, hash proof system) 的概念, 给出了标准模型下构造 CCA-secure PKE 的全新范式, 完美的阐释了 CS98-PKE 的设计原理. 在同一篇论文中, 作者还正式提出了 KEM+DEM 的公钥加密工作模式, 相比朴素混合加密更加现代、模块化. 以下首先介绍 HPS 的定义和相关性质.

### 定义 4.10 (哈希证明系统)

HPS 包含以下多项式时间算法:

- $\text{Setup}(1^\kappa)$ : 以安全参数  $\kappa$  为输入, 输出公开参数  $pp = (H, SK, PK, X, L, W, \Pi, \alpha)$ , 其中  $H : SK \times X \rightarrow \Pi$  是由私钥集合  $SK$  索引的一族带密钥哈希函数 (keyed hash function),  $L$  是定义在  $X$  上的  $\mathcal{NP}$  语言,  $W$  是对应的证据集合,  $\alpha$  是从私钥集合  $SK$  到公钥集合  $PK$  的投射函数.
- $\text{KeyGen}(pp)$ : 以公开参数  $pp$  为输入, 随机采样  $sk \xleftarrow{R} SK$ , 计算  $pk \leftarrow \alpha(sk)$ , 输出  $(pk, sk)$ .
- $\text{PrivEval}(sk, x)$ : 以私钥  $sk$  和  $x \in X$  为输入, 输出  $\pi = H_{sk}(x)$ .
- $\text{PubEval}(pk, x, w)$ : 以公钥  $pk$ 、 $x \in L$  以及相应的  $w$  为输入, 输出  $\pi = H_{sk}(x)$ , 其中  $\alpha(sk) = pk$ .

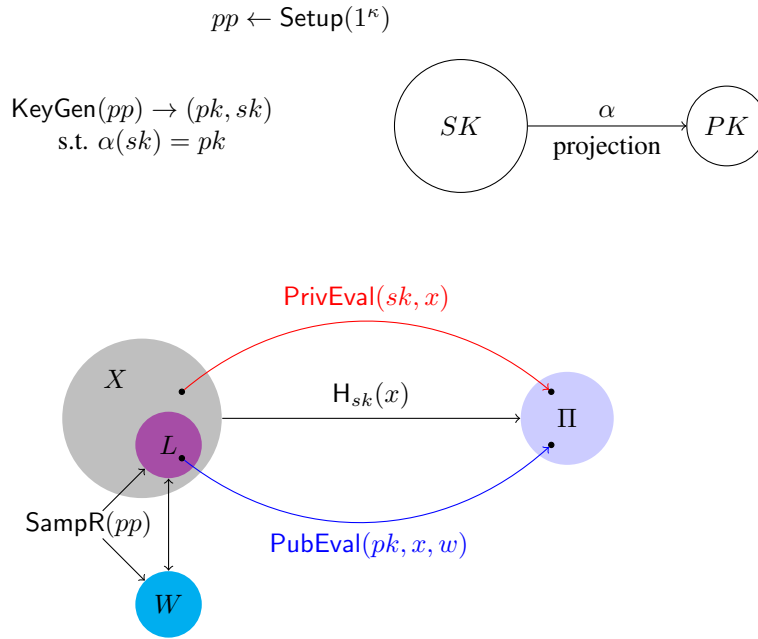


图 4.15: HPS 示意图

HPS 的定义围绕  $L \subset X$  展开, 引入了  $\text{KeyGen}$ ,  $\text{PrivEval}$  和  $\text{PubEval}$  这三个核心算法. 以下性质刻画了哈希函数在输入  $x \in L$  上的行为, 用于保证上层密码方案的功能性.

**投射性 (Projectivity):**  $\forall x \in L$ , 函数值  $H_{sk}(x)$  由  $x$  和私钥的投射  $pk \leftarrow \alpha(sk)$  完全确定.

以下性质由弱到强刻画了哈希函数在输入  $x \in X \setminus L$  上的行为, 用于保证上层密码方案的安全性.

**平滑性 (Smooth):**  $H_{sk}(\cdot)$  在输入  $x \xleftarrow{R} X \setminus L$  时的输出与  $\Pi$  上的均匀分布统计接近, 即:

$$(pk, H_{sk}(x)) \approx_s (pk, \pi)$$

其中  $(pk, sk) \leftarrow \text{KeyGen}(pp)$ ,  $\pi \xleftarrow{R} \Pi$ .

**1-一致性 (1-Universal):**  $H_{sk}(\cdot)$  在任意输入的输出与  $\Pi$  上的均匀分布统计接近, 即  $\forall x \in X \setminus L$ , 有:


$$(pk, H_{sk}(x)) \approx_s (pk, \pi)$$

其中  $(pk, sk) \leftarrow \text{KeyGen}(pp)$ ,  $\pi \xleftarrow{R} \Pi$ .

**2-一致性 (2-Universal):** 在给定某点  $x^* \in X \setminus L$  哈希函数值的情形下,  $H_{sk}(\cdot)$  在任意输入的输出仍与  $\Pi$  上的均匀分布统计接近, 即  $\forall x, x^* \in X \setminus L$  且  $x \neq x^*$ , 有:

$$(pk, H_{sk}(x^*), H_{sk}(x)) \approx_s (pk, H_{sk}(x^*), \pi)$$

其中  $(pk, sk) \leftarrow \text{KeyGen}(pp)$ ,  $\pi \xleftarrow{R} \Pi$ .

 **笔记** 以上三条性质由弱到强. smooth 性质同时建立在  $sk \xleftarrow{R} SK$  和  $x \xleftarrow{R} X \setminus L$  两根随机带上, 1-universal 性质仅建立在  $sk \xleftarrow{R} SK$  一根随机带上, 而 2-universal 性质则可解读为要求 1-universal 性质在随机带  $sk \xleftarrow{R} SK$  有偏时 (将  $H_{sk}(x^*)$  理解为关于  $sk$  的泄漏) 仍然成立. 特别注意, 三条性质均刻画的是输入在语言外时哈希函数的行为.

### 4.2.1 起源释疑

相信很多读者在阅读 HPS 早期的文献时, 都会对这个范式的命名和引入动机感到疑惑. 事实上, HPS 是一类指定验证者的非交互式零知识证明系统 (designated verifier NIZK), 引入的动机来自以下的思考: Naor-Yung 双重加密范式使用标准的 NIZK 来证明密文的合法性 (well-formedness), 然而密文的合法性并非一定是可公开验证的 (public verifiable), 解密私钥  $sk$  的持有者可验证即可. 指定可验证弱于公开可验证, 因此 DV-NIZK 的效率通常高于 NIZK. 想必 Cramer 和 Shoup 正是基于以上的思考, 引入了 HPS, 目的是在标准模型下构造高效的 CCA-secure PKE.

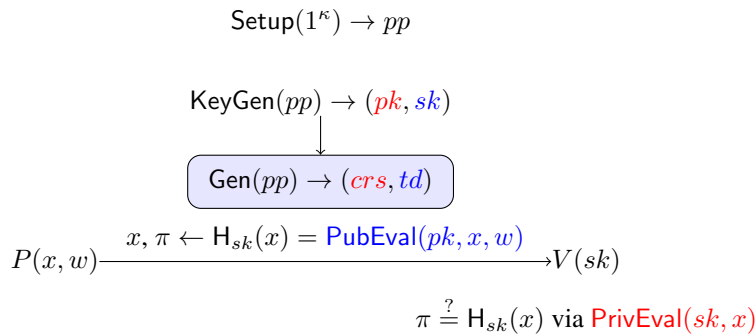



图 4.16: 从 DV-NIZK 的视角解析 HPS

 **笔记** 上图解释了 HPS 的命名渊源, 其本质上是指定验证者零知识证明, 证明的形式是实例的哈希值, 故名哈希证明系统.

- DV-NIZK 的完备性由  $H_{sk}(\cdot)$  的投射性保证:

$$\forall x \in L, H_{sk}(x) = \text{PubEval}(pk, x, w)$$

- DV-NIZK 的合理性由 1-universal 性质保证  $\forall x \notin L$ ,  $H_{sk}(x)$  随机分布, 即使拥有无限计算能力的证明者  $P^*$  也无法预测, 因此通过验证的概率可忽略. 2-universal 性质则保证了更强的合理性, 即敌手在看到 No 实例的有效证明后, 也无法为一个新的 No 实例生成有效证明.
- DV-NIZK 的零知识性是显然且平凡的: 指定验证者拥有私钥, 因此可以对任意的  $x \in L$  (甚至是  $x \in X$ ) 生成正确的证明

此外, 证明系统是有效的, 即证明者在拥有证据时可以高效计算出实例的证明, 这对于基于 HPS 密码方案的功能性至关重要.

### 4.2.2 HPS 的构造

我们首先以针对  $L_{\text{DDH}}$  语言的 HPS 构造为例, 获得对 HPS 设计方式的直观认识. 令  $(\mathbb{G}, p, g)$  是算法  $\text{GroupGen}(1^\kappa)$  的输出, 其中  $\mathbb{G}$  是阶为素数  $p$  的群,  $g$  是生成元. 随机选取  $\mathbb{G}$  中的两个生成元  $g_1, g_2$ . 令  $pp = (\mathbb{G}, p, g_1, g_2)$  是公开参数, 定义由  $pp$  索引的  $\mathcal{NP}$  语言如下:

$$L_{\text{DDH}} = \{(x_1, x_2) \in X : \exists w \in W \text{ s.t. } x_1 = g_1^w, x_2 = g_2^w\}$$

其中  $X = \mathbb{G} \times \mathbb{G}$ ,  $W = \mathbb{Z}_p$ .

容易验证, 语言中的元素是 DH 对, 语言外的元素是非 DH 对,  $(x_1, x_2) \xleftarrow{R} L_{\text{DDH}}$ . DDH 假设蕴含  $L \subset X$  上的 SMP 困难问题成立, 即:

- $U_L \approx_c U_X$ : 随机 DH 对与  $X$  中的随机二元组计算不可区分
- 由于  $|L|/|X| = 1/p = \text{negl}(\kappa)$ ,  $L$  在  $X$  中稀疏, 所以可以进一步得到  $U_L \approx_c U_{X \setminus L}$ : 随机 DH 对与随机非 DH 对计算不可区分

#### 构造 4.9 ( $L_{\text{DDH}}$ 语言的 HPS 构造)

$L_{\text{DDH}}$  的 HPS 构造如下, 如图 4.17 所示:

- $\text{Setup}(1^\kappa)$ : 以安全参数  $\kappa$  为输入, 输出公开参数  $pp = (\mathbb{G}, p, g_1, g_2)$ .  $pp$  还包括了对  $SK = \mathbb{Z}_p \times \mathbb{Z}_p$ ,  $PK = \mathbb{G}$ ,  $L_{\text{DDH}}$ ,  $X = \mathbb{G} \times \mathbb{G}$  和  $W = \mathbb{Z}_p$  的描述.
- $\text{KeyGen}(pp)$ : 以公开参数  $pp$  为输入, 随机采样  $sk \xleftarrow{R} \mathbb{Z}_p^2$ , 计算  $pk \leftarrow \alpha(sk) = g_1^{sk_1} g_2^{sk_2}$ , 输出  $(pk, sk)$ .
- $\text{PrivEval}(sk, x)$ : 以私钥  $sk$  和  $x \in X$  为输入, 输出  $\pi = H_{sk}(x) = x_1^{sk_1} x_2^{sk_2}$ .
- $\text{PubEval}(pk, x, w)$ : 以公钥  $pk$ 、 $x \in L_{\text{DDH}}$  以及相应的  $w$  为输入, 输出  $\pi = pk^w$ , 其中  $\alpha(sk) = pk$ . 以下等式说明了公开求值算法的正确性:

$$pk^w = (g_1^{sk_1} g_2^{sk_2})^w = x_1^{sk_1} x_2^{sk_2} = H_{sk}(x)$$



#### 引理 4.2

以上关于  $L_{\text{DDH}}$  的 HPS 满足 1-universal 性质.



**证明** 证明的目标是

$$\forall x \in X \setminus L, (pk, H_{sk}(x)) \approx_s (pk, \pi)$$

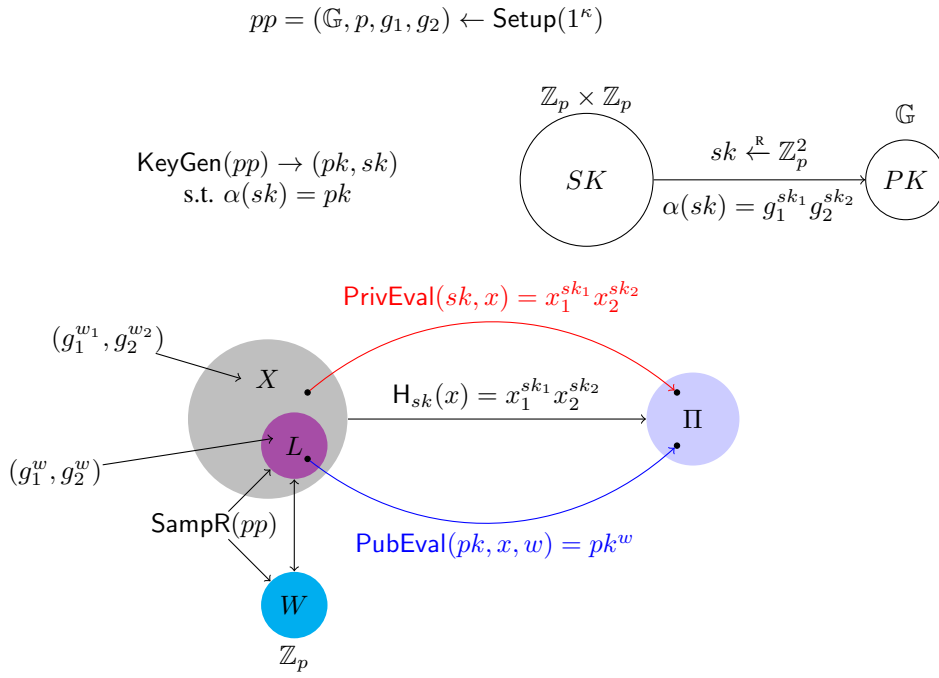
其中  $(pk, sk) \leftarrow \text{KeyGen}(pp)$ ,  $\pi \xleftarrow{R} \Pi$ .

首先固定  $x = (x_1 = g_1^{w_1}, x_2 = g_2^{w_2}) \in X \setminus L$ , 其中  $w_1 \neq w_2$ . 将左式表示为关于  $sk$  函数的形式:

$$(pk, H_{sk}(x)) = f_{g_1, g_2, x_1, x_2}(sk_1, sk_2) := (g_1^{sk_1} g_2^{sk_2}, x_1^{sk_1} x_2^{sk_2})$$

用矩阵的形式描述函数作用过程:

$$\begin{pmatrix} g_1 & g_2 \\ g_1^{w_1} & g_2^{w_2} \end{pmatrix} \begin{pmatrix} sk_1 \\ sk_2 \end{pmatrix} = \begin{pmatrix} pk \\ H_{sk}(x) \end{pmatrix}$$

图 4.17:  $L_{\text{DDH}}$  的 HPS

令  $g_2 = g_1^\beta$ , 其中  $\beta \in \mathbb{Z}_p$ , 将最左边矩阵进行等价变形:

$$\begin{pmatrix} g_1 & g_2 \\ g_1^{w_1} & g_2^{w_2} \end{pmatrix} = \begin{pmatrix} g_1 & g_1^\beta \\ g_1^{w_1} & g_1^{w_2\beta} \end{pmatrix} = g_1 \underbrace{\begin{pmatrix} 1 & \beta \\ w_1 & w_2\beta \end{pmatrix}}_M$$

$\det(M) = \beta(w_1 - w_2) \Rightarrow M$  满秩  $\Rightarrow f$  单射. 又由于函数的定义域和值域大小相等, 最终得出:

$$\underbrace{\begin{pmatrix} g_1 & g_2 \\ g_1^{w_1} & g_2^{w_2} \end{pmatrix}}_{\text{full rank } 2 \times 2} \underbrace{\begin{pmatrix} sk_1 \\ sk_2 \end{pmatrix}}_{\text{uniform over } \mathbb{Z}_p^2} = \underbrace{\begin{pmatrix} pk \\ H_{sk}(x) \end{pmatrix}}_{\text{uniform over } \mathbb{G}^2}$$

从而 1-universal 性质得证! □

#### 注记 4.11

HPS 并不一定要求  $L \subseteq X$  之上一定存在 SMP 问题, 但只有当  $L \subseteq X$  之上存在 SMP 问题时, 相应的 HPS 有密码学意义. 这是因为 HPS 中所有关于哈希函数的性质均是针对输入在语言外时定义的, 只有当 SMP 问题存在时, 才可以间接刻画出哈希函数在输入在语言内时的行为.

HPS 存在两个局限:

- 证明只支持私密验证, 不满足公开验证性
- 证明的表达能力有限, 目前仅能对证明群中的子群成员归属问题, 尚未知能否延伸到任意的 NP 语言.

在很多具体的零知识证明应用场合, 公开验证性和强大的表达能力均不是必须, 因此用标准的零知识证明系统有大材小用之嫌, 哈希证明系统可以做的更快更好, 其中效率的优势恰恰源自局限. 以下展示如何基于 HPS 设计 IND-CPA 和 IND-CCA 的 KEM 方案.

### 4.2.3 基于 HPS 构造 IND-CPA KEM

作为暖场应用,我们首先介绍如何基于 HPS 构造 CPA 安全的 KEM. 设计的思路如下:

- 发送方扮演 HPS 中的证明者, 选择  $L$  中的随机实例  $x$  作为密文  $c$ , 利用公钥  $pk$  和相应的证据  $w$  计算其哈希证明  $\pi$  作为会话密钥  $k$ .
- 接收方扮演 HPS 中的验证者, 使用私钥  $sk$  计算  $x$  的哈希证明以恢复会话密钥  $k$ .

#### 构造 4.10 (基于 HPS 的 CPA 安全 KEM)

从 smooth 的 HPS 出发, 构造 CPA 安全的 KEM 如下:

- **Setup( $\kappa$ ):** 运行  $pp \leftarrow \text{HPS.Setup}(1^\kappa)$ , 输出  $pp = (H, SK, PK, X, L, W, \Pi, \alpha)$  作为公开参数, 其中  $X$  作为密文空间,  $\Pi$  作为会话密钥空间.
- **KeyGen( $pp$ ):** 运行  $(pk, sk) \leftarrow \text{HPS.KeyGen}(pp)$ , 输出公钥  $pk$  和私钥  $sk$ .
- **Encaps( $pk; r$ ):** 以公钥  $pk$  和随机数  $r$  为输入, 执行如下步骤:
  1. 运行  $(x, w) \leftarrow \text{SampR}(r)$  生成随机实例和相应的证据;
  2. 通过  $\text{HPS.PubEval}(pk, x, w)$  计算实例  $x$  的哈希证明  $\pi \leftarrow H_{sk}(x)$ ;
  3. 输出实例  $x$  作为密文  $c$ , 输出哈希证明  $\pi$  作为会话密钥  $k$ .
- **Decaps( $sk, c$ ):** 以私钥  $sk$  和密文  $c$  为输入, 通过  $\text{HPS.PrivEval}(sk, c)$  计算  $c$  的哈希证明  $\pi \leftarrow H_{sk}(x)$  以恢复会话密钥  $k$ .

KEM 方案的正确性有 HPS 的完备性保证. 安全性由如下定理保证.

#### 定理 4.11

如果  $L \subseteq X$  上的 SMP 困难问题成立, 那么构造 4.10 中的 KEM 是 IND-CPA 安全的.

**证明** 我们将通过游戏序列组织证明. 游戏序列的编排次序由如下证明思路指引:

- 将诚实生成的密文分布  $x \xleftarrow{R} L$  切换为  $x \xleftarrow{R} X \setminus L$
- 论证当  $x \xleftarrow{R} X \setminus L$  时,  $(pk, \pi = H_{sk}(x))$  的分布与  $(pk, \pi \xleftarrow{R} \Pi)$  统计接近.

**Game<sub>0</sub>:** 对应真实的游戏, 其中挑战密文  $x^* \xleftarrow{R} L$ , 计算会话密钥的方式是对  $H_{sk}(x^*)$  进行公开求值

- 初始化:  $\mathcal{CH}$  计算  $pp \leftarrow \text{HPS.Setup}(1^\kappa)$ ,  $(pk, sk) \leftarrow \text{HPS.KeyGen}(pp)$ , 将  $pp$  和  $pk$  发送给  $\mathcal{A}$ .
- 挑战:  $\mathcal{CH}$  按照以下步骤生成挑战
  1. 随机采样  $(x^*, w^*) \leftarrow \text{SampR}(r^*)$ ;
  2. 通过  $\text{HPS.PubEval}(pk, x^*, r^*)$  公开计算  $\pi^* \leftarrow H_{sk}(x^*)$ ;
  3. 令  $c^* = x^*$ ,  $k_0^* = \pi^*$ , 随机采样  $k_1^* \xleftarrow{R} \Pi$ ;
  4. 随机选取  $\beta \xleftarrow{R} \{0, 1\}$ , 将  $(c^*, k_\beta^*)$  发送给  $\mathcal{A}$  作为挑战.

敌手  $\mathcal{A}$  在游戏中的视图包括  $(pp, pk, x^*, k_\beta^*)$ .

- 应答:  $\mathcal{A}$  输出对  $\beta$  的猜测  $\beta'$ ,  $\mathcal{A}$  成功当且仅当  $\beta' = \beta$ .

为了准备将挑战密文的分布从  $x^* \in L$  切换到  $x^* \in X \setminus L$ , 我们首先需要引入以下的游戏作为过渡, 这是因为分布切换后  $x^*$  已经不在语言  $L$  内,  $\mathcal{CH}$  无法再以公开求值的方式计算哈希证明, 所以需要提前改变  $\mathcal{CH}$  的求值方式.

**Game<sub>1</sub>:** 与 **Game<sub>0</sub>** 相比唯一的区别在于挑战阶段的步骤 2,  $\mathcal{CH}$  通过  $\text{HPS.PrivEval}(sk, x^*)$  秘密计算  $\pi^* \leftarrow H_{sk}(x^*)$ .  $H_{sk}(\cdot)$  的投射性质保证了当  $x^* \in L$  时,  $\text{PubEval}(pk, x^*, w^*) = H_{sk}(x^*) = \text{PrivEval}(sk, x^*)$  因此在敌手的视角中,  $\mathcal{CH}$  所作出的改变完全不可察觉, 我们有:

$$\text{Game}_0 \equiv \text{Game}_1$$

经过 **Game<sub>1</sub>** 的铺垫, 我们可以顺利过渡到以下的 **Game<sub>2</sub>**.

**Game<sub>2</sub>:** 与 **Game<sub>1</sub>** 唯一的区别是调用  $\text{SampNo}(r^*)$  采样  $x^* \xleftarrow{R} X \setminus L$ . SMP 问题的困难性保证了敌手在相邻游戏中的视图计算不可区分:

$$\text{Game}_1 \approx_c \text{Game}_2$$



Game<sub>3</sub>: 与 Game<sub>2</sub> 的惟一不同是在挑战阶段随机采样  $\pi^* \xleftarrow{R} \Pi$  替代  $\pi^* \leftarrow H_{sk}(x^*)$ . 由  $H_{sk}(\cdot)$  的平滑性保证:

$$\text{Game}_2 \approx_s \text{Game}_3$$

在 Game<sub>3</sub> 中,  $k_0^*$  和  $k_1^*$  均是  $\Pi$  上的均匀分布, 因此即使对于拥有无穷计算能力的敌手  $\mathcal{A}$ , 其优势也为 0. 综合以上, 定理得证!  $\square$

#### 4.2.4 基于 HPS 构造 IND-CCA KEM

我们首先以自问自答的方式分析基于 HPS 构造 CCA-secure KEM 的难点.

构造 4.10 中的 KEM 方案是 IND-CCA 安全的么?

- 从归约证明的角度粗略分析似乎并没有技术困难, 因为归约算法  $\mathcal{R}$  始终掌握私钥  $sk$ , 可以回答任意的解封装询问. 然而细致分析后发现并非如此. 与 IND-CPA 安全游戏相比, 在 IND-CCA 安全游戏中敌手的视图中额外包括了对解封装询问的应答. 当解封装询问  $c = x$  的密文  $x \notin L$  时, 应答会泄漏更多关于  $sk$  的信息 (公钥  $pk$  可以看做关于  $sk$  的部分泄漏). 因此我们无法再使用平滑性得出  $\text{Game}_2 \approx_s \text{Game}_3$  的结论.

接上问, 既然当  $x \in X \setminus L$  时的解封装询问会泄漏  $sk$  的信息, 那拒绝此类询问是否可以达到 IND-CCA 安全性呢?

- 不可以. 这是因为 SMP 问题的困难性使得 PPT 的解密者无法判定是否  $x \in L$ . 善于思考的读者很能发现解密者还拥有解密私钥  $sk$ , 然而解密者 (对应诚实用户) 仅拥有一个解密私钥, 依然无法判定是否  $x \in L$ . 那是否有巧妙的方案设计使得解密者拥有多个解密私钥, 从而解密者可以通过检测多个私钥求值的一致性来判定  $x \in L$ ? 答案依然是否定的, 因为 SMP 的困难性否定了此类方案设设计算法的存在性. 反过来, 如果解密者拥有了对应 SMP 问题公开参数对应的秘密参数, 那么确实可以设计方案使得解密者拥有多个解密私钥, 比如考虑  $L_{DDH}$  语言的 HPS 4.9, 如果解密者知晓  $\alpha$  使得  $g_1^\alpha = g_2$ , 那么任取  $\Delta \in \mathbb{Z}_p$ , 均有:

$$(sk_1, sk_2) \sim (sk'_1 = sk_1 + \alpha\Delta, sk'_2 = sk_2 - \Delta) \Leftrightarrow g_1^{sk_1} g_2^{sk_2} = g_1^{sk'_1} g_2^{sk'_2}$$

上述设计方案已经暗含了 SMP 问题的困难性对解密者不复存在, 这使得安全归约将会在游戏  $\text{Game}_1 \approx_c \text{Game}_2$  的步骤失败, 原因是归约算法 (针对 SMP 问题的敌手) 不掌握  $\alpha$ , 从而无法模拟解密者的行为.

通过以上的分析, 不难得出基于 HPS 构造 CCA-secure KEM 的一种思路是杜绝“危险”的解密询问:

- $x \in L$  属于安全的解密询问, 这是因为应答  $\pi = \text{HPS.PubEval}(pk, x, w)$  没有额外泄漏关于  $sk$  的信息, 因此不会破坏平滑性.
- $x \notin L$  属于危险的解密询问, 杜绝的思路在密文中嵌入某种**私密认证结构**, 使得 PPT 的敌手无法生成有效的 (valid) 危险密文, 同时解密者能够判定密文是否有效. 具体的设计思路是将哈希证明作为信息论意义下的一次性消息验证码 (information-theoretic one-time MAC), 此处需要满足 2-universal 性质的 HPS.

#### 构造 4.11 (基于 HPS 的 CCA-secure KEM)

构造的组件是:

- 满足 smooth 性质的 HPS<sub>1</sub>
- 满足 2-universal 性质的 HPS<sub>2</sub>
- Setup( $1^\kappa$ ):
  1. 运行  $pp_1 \leftarrow \text{HPS}_1.\text{Setup}(1^\kappa)$ , 其中  $pp_1 = (H_1, SK_1, PK_1, X, L, W, \Pi_1, \alpha_1)$ ;
  2. 运行  $pp_2 \leftarrow \text{HPS}_2.\text{Setup}(1^\kappa)$ , 其中  $pp_2 = (H_2, SK_2, PK_2, X, L, W, \Pi_2, \alpha_2)$ ;
  3. 输出公开参数  $pp = (pp_1, pp_2)$ . 公钥空间  $PK = PK_1 \times PK_2$ , 私钥空间  $SK = SK_1 \times SK_2$ , 密文空间  $C = X \times \Pi_2$ , 会话密钥空间  $K = \Pi_1$ .
- KeyGen( $pp$ ): 解析  $pp = (pp_1, pp_2)$ , 执行以下步骤:
  1. 计算  $(pk_1, sk_1) \leftarrow \text{HPS}_1.\text{KeyGen}(pp_1)$ ;

2. 计算  $(pk_2, sk_2) \leftarrow \text{HPS}_2.\text{KeyGen}(pp_2)$ ;
3. 输出公钥  $pk = (pk_1, pk_2)$  和私钥  $sk = (sk_1, sk_2)$ .
- $\text{Encaps}(pk; r)$ : 以公钥  $pk = (pk_1, pk_2)$  和随机数  $r$  为输入, 执行以下步骤:
  1. 运行  $(x, w) \leftarrow \text{SampR}(r)$  随机采样语言  $L_1$  中的实例和相应证据;
  2. 通过  $\text{HPS}_1.\text{PubEval}(pk_1, x, w)$  计算实例  $x$  在  $\text{HPS}_1$  中的哈希证明  $\pi_1 \leftarrow H_1(sk_1, x)$ ;
  3. 通过  $\text{HPS}_2.\text{PubEval}(pk_2, x, w)$  计算实例  $x$  在  $\text{HPS}_2$  中的哈希证明  $\pi_2 \leftarrow H_2(sk_2, x)$ ;
  4. 输出实例  $x$  和  $\pi_2$  作为密文  $c$ , 其中  $\pi_2$  可以看做  $x$  的 MAC 值; 输出哈希证明  $\pi_1$  作为会话密钥  $k$ .
- $\text{Decap}(sk, c)$ : 以私钥  $sk = (sk_1, sk_2)$  和密文  $c = (x, \pi_2)$  为输入, 通过  $\text{HPS}_2.\text{PrivEval}(sk_2, x)$  计算  $x$  的哈希证明  $\pi'_2 \leftarrow H_2(sk_2, x)$ ; 如果  $\pi_2 \neq \pi'_2$  则输出  $\perp$ , 否则通过  $\text{HPS}_1.\text{PrivEval}(sk_1, x)$  计算  $x$  的哈希证明  $\pi_1 \leftarrow H_1(sk_1, x)$  以恢复会话密钥  $k$ .

构造 4.11 的正确性由  $\text{HPS}_1$  和  $\text{HPS}_2$  的完备性保证, 安全性由以下定理保证.

#### 定理 4.12

如果  $L \subseteq X$  上的 SMP 问题成立, 那么构造 4.11 中的 KEM 是 IND-CCA 安全的.

**证明** 为了便于安全分析, 首先对密文  $c = (x, \pi_2)$  做如下的分类:

- 良生成的 (well-formed)  $\iff x \in L$
- 有效的 (valid)  $\iff H_{sk_2}^2(x) = \pi_2$

根据以上定义, 良生成的密文有可能是无效的, 有效的密文也可能是非良生成的. 在基于 HPS 构造的 KEM 中, 非良生成的密文是“危险的”, 因为解封装询问的结果会泄漏关于私钥的信息.

以下通过游戏序列完成定理证明:

**Game<sub>0</sub>**: 对应真实的游戏

- 初始化:  $\mathcal{CH}$  生成  $pp_1 \leftarrow \text{HPS}_1.\text{Setup}(1^\kappa)$ ,  $pp_2 \leftarrow \text{HPS}_2.\text{Setup}(1^\kappa)$ , 计算  $(pk_1, sk_1) \leftarrow \text{HPS}_1.\text{KeyGen}(pp_1)$ ,  $(pk_2, sk_2) \leftarrow \text{HPS}_2.\text{KeyGen}(pp_2)$ , 发送  $pp = (pp_1, pp_2)$  和  $pk = (pk_1, pk_2)$  给敌手  $\mathcal{A}$ .
- 挑战:  $\mathcal{CH}$  执行以下操作生成挑战
  1. 运行  $(x^*, w^*) \leftarrow \text{SampR}(r^*)$  随机采样  $L$  中的实例和证据;
  2. 通过  $\text{HPS}_1.\text{PubEval}(pk_1, x^*, r^*)$  计算哈希证明  $\pi_1^* \leftarrow H_1(sk_1, x^*)$ ;
  3. 通过  $\text{HPS}_2.\text{PubEval}(pk_2, x^*, w^*)$  计算哈希证明  $\pi_2^* \leftarrow H_2(sk_2, x^*)$ ;
  4. 令  $c^* = (x^*, \pi_2^*)$ ,  $k_0^* = \pi_1^*$ ,  $k_1^* \xleftarrow{R} \Pi$ ;
  5. 选择随机比特  $\beta \xleftarrow{R} \{0, 1\}$ , 发送  $(c^*, k_\beta^*)$  给  $\mathcal{A}$  作为挑战.
- 解封装询问: 当敌手发起解封装询问  $c = (x, \pi_2)$  时,  $\mathcal{CH}$  分情况应答如下
  - $c = c^*$ : 返回  $\perp$ ;
  - $c \neq c^*$ : 如果  $\pi_2 = \text{HPS}_2.\text{PrivEval}(sk_2, x)$  返回  $\text{HPS}_1.\text{PrivEval}(sk_1, x)$ ; 否则返回  $\perp$ .

**Game<sub>1</sub>**: 与 CPA 构造情形类似, 该游戏的引入是为了将密文  $c^*$  由语言  $L$  内切换到语言外. 在挑战阶段,  $\mathcal{CH}$  通过  $\text{HPS}_1.\text{PrivEval}(sk_1, x^*)$  计算  $\pi_1^* \leftarrow H_1(sk_1, x^*)$ , 通过  $\text{HPS}_2.\text{PrivEval}(sk_2, x^*)$  计算  $\pi_2^* \leftarrow H_2(sk_2, x^*)$ . HPS 的投射性保证了  $\text{Game}_0 \equiv \text{Game}_1$ .

**Game<sub>2</sub>**: 将随机采样  $L$  中的实例和证据  $(x^*, w^*) \xleftarrow{R} \text{SampR}(r^*)$  切换为随机采样  $X \setminus L$  中的实例  $x^* \leftarrow \text{SampNo}(r^*)$ . SMP 问题的困难性保证了敌手在相邻游戏中的视图计算不可区分:

$$\text{Game}_1 \approx_c \text{Game}_2$$

在游戏序列演进过程中, 仅在论证  $\text{Game}_1 \approx_c \text{Game}_2$  时依赖计算困难假设; 其余的分析均在信息论意义下 (information-theoretic) 完成, 从此刻起挑战者  $\mathcal{CH}$  拥有无穷计算能力.

**Game<sub>3</sub>**: 微调解密规则, 将直接拒绝非良生成但有效的 (ill-formed but valid) 密文. 对于解封装询问  $c = (x, \pi_2)$ , 只要  $x \notin L$ , 那么即使  $\pi_2 = H_{sk_2}^2(x)$  也直接返回  $\perp$  表示拒绝. 改变规则的目的是拒绝所有危险密文, 从而确保解封装询问的应答不泄漏关于私钥的信息.

#### 断言 4.7

$$|\Pr[S_3] - \Pr[S_2]| \leq \text{negl}(\kappa).$$

**证明** 注意到正常的解封装算法会对此类密文返回解封装结果, 并不是直接返回  $\perp$  拒绝. 为了分析规则改变引发的差异, 引入如下事件  $E$ :

- $\mathcal{A}$  发起非良生成但有效的解封装询问, 即:  $x \notin L \wedge \pi_2 = H_{sk_2}^2(x)$

显然如果事件  $E$  不发生, 那么 **Game<sub>2</sub>** 与 **Game<sub>3</sub>** 完全相同. 令  $Q$  表示  $\mathcal{A}$  发起解封装询问的最大次数, **HPS<sub>2</sub>** 的 2-universal 保证了 (存疑):

$$\Pr[E] \leq Q/|\Pi_2| = \text{negl}(\kappa)$$

利用差异引理, 断言得证.  $\square$

**Game<sub>4</sub>**: 对所有良生成的解封装询问  $c = (x, \pi_2)$  也即  $x \in L$ ,  $\mathcal{CH}$  使用公钥  $pk = (pk_1, pk_2)$  和相应的证据  $w$  应答. 注意到  $\mathcal{CH}$  拥有无穷计算能力, 因此能够计算出  $x \in L$  的证据  $w$ . 该规则变化仅是为了说明对良生成密文的解封装不会额外泄漏关于私钥的信息, 不会引发敌手视图的任何改变, 因此 **Game<sub>3</sub>**  $\equiv$  **Game<sub>4</sub>**.

**Game<sub>5</sub>**: 随机采样  $\pi_1^* \xleftarrow{R} \Pi_1$  代替  $\pi^* \leftarrow H_1(sk_1, x^*)$ .

#### 断言 4.8

敌手  $\mathcal{A}$  在 **Game<sub>4</sub>** 和 **Game<sub>5</sub>** 中的视图统计不可区分.

**证明** 敌手  $\mathcal{A}$  在 **Game<sub>4</sub>** 和 **Game<sub>5</sub>** 中的视图均由以下部分组成:

- 公开参数:  $pp = (pp_1, pp_2)$ ;
- 公钥:  $pk = (pk_1, pk_2)$ ;
- 挑战: 密文  $c^* = (x^*, \pi_2^*)$  和会话密钥  $k_\beta^*$ ;
- 解封装询问: 由公钥  $pk$  和敌手  $\mathcal{A}$  的询问确定.

接下来, 我们通过递增分布项的方式证明断言:

1. 首先由 **HPS<sub>1</sub>** 的平滑性可知, 当  $x^* \xleftarrow{R} X \setminus L$  是有:

$$(pk_1, x^*, \boxed{H_1(sk_1, x^*)}) \approx_s (pk_1, x^*, \boxed{U_{\Pi_1}})$$

2. 将  $(pk_2, \pi_2^*)$  表示为  $g_{sk_2}(x^*)$ , 其中  $g_{sk_2}(x) := (\alpha_2(sk_2), H_2(sk_2, x))$ . 复合引理 (composition lemma) 可推出  $X \approx_s Y \Rightarrow f(X) \approx_s f(Y)$ , 其中  $f$  可以是任意 (概率) 函数. 将上面公式左右两边的分布分别看成  $X$  和  $Y$ , 令  $f(pk_2, x^*, \pi_2) = (g_{sk_2}(x^*), pk_2, x^*, \pi_2)$ , 应用复合引理即可得:

$$(pk_2, \pi_2^*, \underline{pk_1, x^*, H_{sk_1}^1(x^*)}) \approx_s (pk_2, \pi_2^*, \underline{pk_1, x^*, U_{\Pi_1}})$$

令  $view' = (pk, x^*, \pi_2^*, k_\beta^*)$ , 上面公式可以简写为  $view'_4 \approx_s view'_5$ .

3. 在左右两边添加解封装结果.  $\mathcal{CH}$  对解封装询问的应答总可以表示为  $f_{\text{decaps}}(view')$ ,  $f_{\text{decaps}}$  编码了敌手  $\mathcal{A}$  选择密文  $\{c_i\}$  的策略和解封装算法, 易知  $f_{\text{decaps}}$  是一个 PPT 算法. 再次应用复合引理, 可以得到:

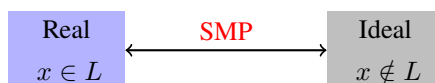
$$(f_{\text{decaps}}(view'_4), view'_4) \approx_s (f_{\text{decaps}}(view'_5), view'_5)$$

根据敌手视图的定义, 可以得到 **Game<sub>4</sub>**  $\approx_s$  **Game<sub>5</sub>**, 断言得证.  $\square$

在 **Game<sub>5</sub>** 中,  $k_0^*$  和  $k_1^*$  均从  $\Pi_1$  中随机采样. 因此对于任意敌手均有  $\Pr[S_5] = 0$ . 综合以上, 定理得证!  $\square$

## 小结

HPS 给出了基于 SMP 类型判定性问题构造公钥加密的范式, 在论证安全性时遵循如下的三步走 (三板斧) 套路:



1. 真实游戏中挑战密文为语言中的随机实例  $x \in L$ ;
2. 理想游戏中挑战密文为语言外的随机实例  $x \notin L$ , 在信息论意义下证明敌手优势可忽略;
3. 利用 SMP 完成语言内外的切换, 论证 PPT 敌手在真实游戏和理想游戏中的优势差可忽略.

论证过程与中国道家的“阴阳相生”思想暗合.

在很多情形下, 公钥加密的私钥嵌入于底层困难问题, 因此设计高等级安全公钥加密的一个常见难点是归约证明过程中, 归约算法  $\mathcal{R}$  需要在未知私钥的情形下模拟与私钥相关的谕言机. 一个具体的例子就是难以证明 ElGamal PKE 具备私钥抗泄漏安全性, 因为私钥嵌入在底层 DDH 困难问题中. Cramer 和 Shoup 另辟蹊径, 绕过了该难点, 关窍是在基于 HPS 的公钥加密设计中, 公钥加密的密文嵌入于底层困难问题, 归约算法  $\mathcal{R}$  始终掌握私钥, 从而可以完美模拟任意与私钥相关的谕言机. 正是该特性使得 HPS 的用途极为广泛, 远远超越了最初的 CCA 安全的公钥加密, 如 HPS 在基于口令的密钥交换 (PAKE: Password authenticated key exchange)、不经意传输 (Oblivious transfer) 的构造中均有重要应用, 更是达成密钥泄漏安全、消息依赖密钥安全等高等级安全的主流技术工具.

## 4.3 可提取哈希证明系统类

事要知其所以然。

— 《朱子语类·卷九·论行知》

1991 年, Rackoff 和 Simon [RS91] 提出了构造 CCA 安全 PKE 的另一条技术路线:

1. 发送方随机选择会话密钥  $k$  并使用接收方的公钥对其加密得到密文  $c$ , 同时生成关于  $k$  的非交互零知识知识证明  $\pi$ , 将  $c$  和  $\pi$  一起发送给接收方;
2. 接收方先验证  $\pi$  的正确性, 若验证通过则利用私钥解密恢复会话密钥;

该条技术路线被称为 Rackoff-Simon 范式, 与 Naor-Yung 范式/Sahai 范式的不同之处是前者需要使用非交互零知识知识的证明 (NIZKPoK), 而后者使用的势非交互零知识证明 (NIZK).

Cramer 和 Shoup 于 2002 年正式提出的哈希证明系统是 NIZK 的弱化: 公开可验证弱化为指定验证者, 表达能力由任意  $\mathcal{NP}$  语言限制为群论语言, 证明的形式特化为哈希值. 2010 年, Wee [Wee10] 提出了可提取哈希证明系统 (EHPS, extractable hash proof system), 并展示了如何基于 EHPS 以一种简洁、模块化的方式构造 CCA 安全的 PKE. 该构造范式统一了几乎所有已知的基于计算性假设的 CCA 安全 PKE 方案. 相对 HPS 是 NIZK 的弱化, EHPS 是 NIZKPoK 的弱化. 以下首先介绍 EHPS 的定义和相关性质.

### 定义 4.11 (可提取哈希证明系统)

EHPS 包含以下多项式时间算法:

- $\text{Setup}(1^\kappa)$ : 以安全参数  $\kappa$  为输入, 输出公开参数  $pp = (H, PK, SK, L, W, \Pi)$ , 其中  $L$  是由困难关系  $R_L$  定义的平凡  $\mathcal{NP}$  语言,  $H: PK \times L \rightarrow \Pi$  是由公钥集合  $PK$  索引的一族带密钥哈希函数. 关系  $R_L$  支持随机采样, 即存在 PPT 算法  $\text{SampRel}$  以随机数  $r$  为输入, 输出随机的“实例-证据”元组  $(x, w) \in R_L$ . 为了方便后续的应用,  $\text{SampRel}$  可以进一步分解为  $\text{SampIns}$  和  $\text{SampWit}$ , 前者随机采样语言中的实例, 后者随机采样证据, 对于任意随机数  $r \in R$ , 我们有  $(\text{SampIns}(r), \text{SampWit}(r)) \in R_L$ .
- $\text{KeyGen}(pp)$ : 以公开参数  $pp$  为输入, 输出公钥  $pk$  和私钥  $sk$ .
- $\text{PubEval}(pk, x, r)$ : 以公钥  $pk$ 、 $x \in L$  和随机数  $r$  为输入, 输出证明  $\pi \in \Pi$ . 正确性要求是当  $r$  是采样  $x$  的随机数时 (即  $(x, w) \leftarrow \text{SampRel}(r)$ ), 算法正确计算出哈希证明:  $\pi = H_{pk}(x)$ . 注意, 当给定采样随机数  $r$  时, 可以运行算法  $\text{SampRel}$  恢复  $x$ , 因此算法的第 2 项输入  $x$  可以省去.
- $\text{Ext}(sk, x, \pi)$ : 以私钥  $sk$ 、 $x \in L$  和证明  $\pi \in \Pi$  为输入, 输出证据  $w \in W \cup \perp$ . 正确性要求是:

$$\pi = H_{pk}(x) \iff (x, \text{Ext}(sk, x, \pi)) \in R_L$$

- $\text{KeyGen}'(pp)$ : 以公开参数  $pp$  为输入, 输出公钥  $pk$  和私钥  $sk'$ .
- $\text{PrivEval}(sk', x)$ : 以私钥  $sk'$  和  $x \in L$  为输入, 输出证明  $\pi \in \Pi$ . 正确性要求是  $\text{PrivEval}$  正确计算出哈希证明:  $\pi = H_{pk}(x)$ .

以上算法中,  $\text{KeyGen}$ 、 $\text{PubEval}$  和  $\text{Ext}$  工作在真实模式,  $\text{KeyGen}'$  和  $\text{PrivEval}$  工作在模拟模式, 两种模式共享同一个  $\text{Setup}$  算法生成公开参数. 两种模式之间的关联是公钥的分布统计不可区分, 即:

$$\text{KeyGen}(pp)[1] \approx_s \text{KeyGen}'(pp)[1]$$



### 4.3.1 起源释疑



**笔记** 上图解释了 EHPS 的命名渊源, 其本质上是指定验证者零知识知识的证明, 证明的形式是实例的哈希值, 故名可提取哈希证明系统.

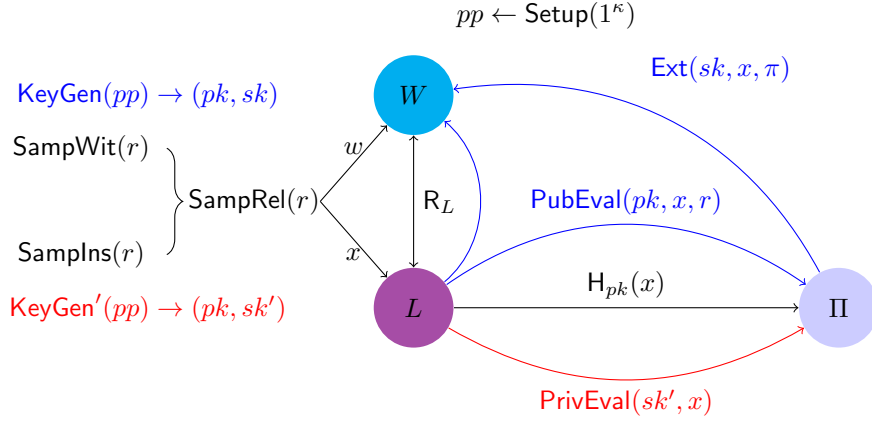


图 4.18: EHPS 的示意图

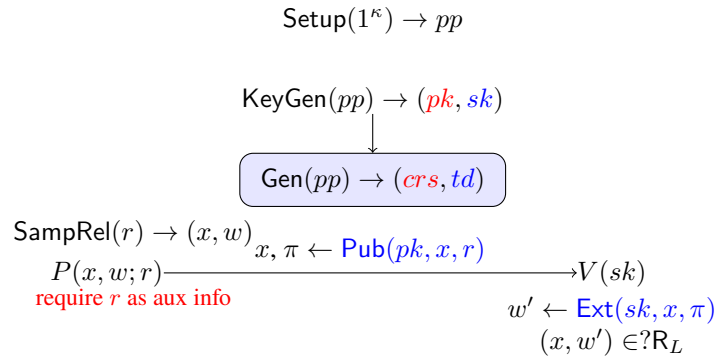


图 4.19: 从 DV-NIZKPoK 的视角解析 EHPS

- DV-NIZKPoK 的完备性和可提取性由  $\text{Ext}$  的正确性保证, 即在正常模式下,

$$\pi = H_{pk}(x) \iff (x, \text{Ext}(sk, x, \pi)) \in R_L$$

其中  $\text{KeyGen}(pp) \rightarrow (pk, sk)$ .

- DV-NIZKPoK 的零知识性论证如下, 令  $\text{KeyGen}(pp) \rightarrow (pk, sk')$ ,  $\text{KeyGen}'(pp) \rightarrow (pk, sk')$ , 对于  $\forall x \in L$ , 我们有:

$$pk \approx_s pk \Rightarrow (pk, H_{pk}(x)) \approx_s (pk, \text{PrivEval}(sk', x))$$

再由秘密求值算法的正确性  $H_{pk}(x) = \text{PrivEval}(sk', x)$  可以得到:

$$(pk, H_{pk}(x)) \approx_s (pk, \text{PrivEval}(sk', x))$$

### 4.3.2 EHPS 的构造

我们以针对  $L_{\text{CDH}}$  语言的 EHPS 构造为例, 获得对 EHPS 设计方式的直观认识. 令  $(\mathbb{G}, p, g)$  是算法  $\text{GroupGen}(1^\kappa)$  的输出, 其中  $\mathbb{G}$  是阶为素数  $p$  的群,  $g$  是生成元. 随机选取  $\mathbb{G}$  中的另一生成元  $g^\alpha$ , 其中  $\alpha \xleftarrow{\mathbb{R}} \mathbb{Z}_p$ . 令  $pp = (\mathbb{G}, p, g, g^\alpha)$  是公开参数, 定义由  $pp$  索引的平凡  $\mathcal{NP}$  语言如下:

$$L_{\text{CDH}} = \{x \in X : \exists w \in W \text{ s.t. } w = x^\alpha\}$$

其中  $L = X = \mathbb{G}$ ,  $W = \mathbb{G}$ . 定义  $L_{\text{CDH}}$  的二元关系为  $R_{\text{CDH}}$ ,  $(x, w) \in R_{\text{CDH}} \iff w = x^\alpha$ . 容易验证:

- $R_{\text{CDH}}$  基于 CDH 假设是困难的.



- $R_{CDH}$  是高效可采样的: 存在 PPT 采样算法  $\text{SampRel}$  随机选取  $r \xleftarrow{R} \mathbb{Z}_p$ , 输出  $(g^r, (g^\alpha)^r) \in R_{CDH}$ .
- 如果  $\mathbb{G}$  是双线性映射群, 则  $R_{CDH}$  是公开可验证的.

#### 构造 4.12 ( $L_{CDH}$ 语言的 EHPS 构造)

$L_{CDH}$  的 EHPS 构造如下, 如图所示:

- $\text{Setup}(1^\kappa)$ : 以安全参数  $\kappa$  为输入, 输出公开参数  $pp = (\mathbb{G}, p, g, g^\alpha)$ . 其中  $pp$  还包括了对  $SK = \mathbb{Z}_p$ ,  $PK = \mathbb{G}$ ,  $L_{CDH} = X = \mathbb{G}$  和  $W = \mathbb{G}$  的描述.
- $\text{KeyGen}(pp)$ : 以公开参数  $pp$  为输入, 随机采样  $sk \xleftarrow{R} \mathbb{Z}_p$ , 计算  $pk = g^{sk} \in \mathbb{G}$ , 输出  $(pk, sk)$ .
- $\text{PubEval}(pk, x, r)$ : 以公钥  $pk$ 、实例  $x \in L_{CDH}$  和  $r \in \mathbb{Z}_p$  为输入, 输出  $\pi \leftarrow (g^\alpha \cdot pk)^r$ .
- $\text{Ext}(sk, x, \pi)$ : 以私钥  $sk$ 、实例  $x \in L_{CDH}$  和  $\pi$  为输入, 计算  $w \leftarrow \pi / x^{sk}$ , 如果  $(x, w) \in R_L$  则返回  $w$ , 否则返回  $\perp$ . 正确性由以下公式保证:

$$\pi / x^{sk} = (g^\alpha \cdot pk)^r / x^{sk} = (g^\alpha \cdot g^{sk})^r / g^{r \cdot sk} = (g^\alpha)^r = w$$

- $\text{KeyGen}'(pp)$ : 以公开参数  $pp$  为输入, 随机采样  $sk' \xleftarrow{R} \mathbb{Z}_p$ , 计算  $pk \leftarrow g^{sk'} / g^\alpha$ .
- $\text{PrivEval}(sk', x)$ : 以私钥  $sk'$  和实例  $x \in L_{CDH}$  为输入, 输出  $w \leftarrow x^{sk'}$ . 正确性由以下公式保证:

$$H_{pk}(x) = (g^\alpha \cdot pk)^r = (g^\alpha \cdot g^{sk'} / g^\alpha)^r = (g^{sk'})^r = x^{sk'}$$

容易验证, 两种模式下生成的  $pk$  服从同样的分布—— $\mathbb{G}$  上的均匀分布.

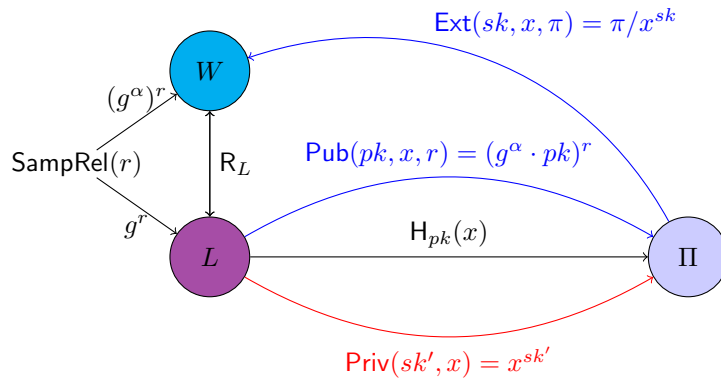


图 4.20:  $L_{CDH}$  的 EHPS

### 4.3.3 基于 EHPS 构造 IND-CPA KEM

作为暖场应用, 我们首先介绍如何基于 EHPS 构造 CPA 安全的 KEM. 设计的思路源自 Rackoff-Simon 范式. 令  $R_L$  为定义在  $X \times W$  上的单向关系,  $hc: W \rightarrow K$  为相应的 hardcore function.

- 发送方扮演 EHPS 中的证明者, 运行  $\text{SampRel}(r)$  算法随机采样  $(x, w) \in R_L$ , 利用公钥  $pk$  和随机数  $r$  计算  $x$  的哈希证明  $\pi$ , 生成密文  $c = (x, \pi)$ , 计算证据  $w$  的 hardcore function 值作为会话密钥  $k$ .
- 接收方扮演 EHPS 中的验证者: 使用私钥  $sk$  从密文  $(x, \pi)$  中恢复  $w$ , 进而恢复会话密钥.

#### 构造 4.13 (基于 EHPS 的 CPA 安全 KEM)

从语言  $L$  的 EHPS 出发, 构造 CPA 安全的 KEM 如下:

- $\text{Setup}(1^\kappa)$ : 运行  $pp \leftarrow \text{EHPS.Setup}(1^\kappa)$ , 输出  $pp = (H, SK, PK, X, L, W, \Pi)$  作为公开参数, 其中  $X \times \Pi$  作为密文空间,  $R_L$  hardcore function 的值域  $K$  作为会话密钥空间.



- $\text{KeyGen}(pp)$ : 运行  $(pk, sk) \leftarrow \text{EHPS.KeyGen}(\lambda)$ , 输出公钥  $pk$  和私钥  $sk$ .
- $\text{Encaps}(pk; r)$ : 以公钥  $pk$  和随机数  $r$  为输入, 执行如下步骤:
  1. 运行  $(x, w) \leftarrow \text{SampRel}(r)$  生成随机实例和相应证据;
  2. 通过  $\text{EHPS.Pub}(pk, x, r)$  计算实例  $x$  的哈希证明  $\pi \leftarrow H_{pk}(x)$ ;
  3. 输出  $(x, \pi)$  作为密文, 计算  $k \leftarrow \text{hc}(w)$  作为会话密钥.
- $\text{Decaps}(sk, c)$ : 以私钥  $sk$  和密文  $c = (x, \pi)$  为输入, 计算  $w \leftarrow \text{EHPS.Ext}(sk, x, \pi)$ , 如果  $(x, w) \notin R_L$  则输出  $\perp$ , 否则输出  $k \leftarrow \text{hc}(w)$ .



KEM 的正确性由 EHPS 的完备性和  $R_L$  的单射性保证, 安全性由如下定理保证.

#### 定理 4.13

如果  $R_L$  是单向的, 那么构造 4.13 中的 KEM 是 IND-CPA 安全的.



**证明** 证明的目标是论证会话密钥  $\text{hc}(w^*)$  在敌手  $\mathcal{A}$  的视图是伪随机的, 其中  $\mathcal{A}$  的视图包括:

- 公开参数  $pp$ ;
- 公钥  $pk$ : 与  $w^*$  无关;
- 密文  $c^* = (x^*, \pi^*)$ :  $R_L$  的单向性保证了  $x^*$  隐藏了  $w^*$ , EHPS 的零知识性进一步保证了  $\pi^*$  (相对于  $x^*$ ) 不会额外泄漏关于  $w^*$  的信息.

我们通过以下的游戏序列组织证明.

$\text{Game}_0$ : 对应真实的游戏.  $\mathcal{CH}$  在真实模式下运行 EHPS 与敌手  $\mathcal{A}$  交互.

- 初始化:  $\mathcal{CH}$  计算  $pp \leftarrow \text{EHPS.Setup}(1^\kappa)$ ,  $(pk, sk) \leftarrow \text{EHPS.KeyGen}(pp)$ , 将  $pp$  和  $pk$  发送给  $\mathcal{A}$ .
- 挑战:  $\mathcal{CH}$  按照以下步骤生成挑战
  1. 随机采样  $(x^*, w^*) \leftarrow \text{SampRel}(r^*)$ ;
  2. 通过  $\text{EHPS.PubEval}(pk, x^*, r^*)$  公开计算  $\pi^* \leftarrow H_{pk}(x^*)$ ;
  3. 计算  $k_0^* \leftarrow \text{hc}(w^*)$ , 随机采样  $k_1^* \xleftarrow{R} K$ ;
  4. 随机选取  $\beta \xleftarrow{R} \{0, 1\}$ , 将  $(c^* = (x^*, \pi^*), k_\beta^*)$  发送给  $\mathcal{A}$  作为挑战.
- 应答:  $\mathcal{A}$  输出对  $\beta$  的猜测  $\beta'$ ,  $\mathcal{A}$  成功当且仅当  $\beta' = \beta$ .

为了利用 EHPS 的零知识性论证  $\pi^*$  不额外泄漏关于  $w^*$  的信息, 需要将 EHPS 由真实模式切换到模拟模式.

$\text{Game}_1$ :  $\mathcal{CH}$  在模拟模式下运行 EHPS 与敌手  $\mathcal{A}$  交互.

- 初始化:  $\mathcal{CH}$  计算  $(pk, sk') \leftarrow \text{EHPS.KeyGen}'(pp)$ .
- 挑战:  $\mathcal{CH}$  在第二步通过  $\text{EHPS.Priv}(sk', x^*)$  计算  $\pi^* \leftarrow H_{pk}(x^*)$ .

敌手  $\mathcal{A}$  在游戏中的视图为  $(pp, pk, x^*, k_\beta^*)$ . 容易验证, EHPS 的零知识性保证了  $\text{Game}_0 \approx_s \text{Game}_1$ :

#### 断言 4.9

如果  $R_L$  是单向的,  $\text{Adv}_{\mathcal{A}}^{\text{Game}_1} = \text{negl}(\kappa)$ .



**证明** 证明思路是如果存在  $\mathcal{A}$  以不可忽略的优势赢得  $\text{Game}_1$ , 那么可以构造出  $\mathcal{B}$  以不可忽略的优势打破  $\text{hc}$  的伪随机性, 从而与单向性假设冲突. 给定关于  $\text{hc}$  的伪随机性挑战  $pp$  和  $(x^*, k_\beta^*)$ , 其中  $(x^*, w^*) \leftarrow \text{SampRel}(r^*)$ ,  $\mathcal{B}$  模拟  $\text{Game}_1$  中的挑战者  $\mathcal{CH}$  与  $\mathcal{A}$  交互, 目标是猜测  $\beta$ .

- $\mathcal{B}$  运行 EHPS 的模拟模式与  $\mathcal{A}$  在  $\text{Game}_1$  进行交互, 在初始化阶段不再采样  $x^*$  而是直接嵌入接收到的  $x^*$ , 在挑战阶段将  $R_L$  的挑战  $(x^*, k_\beta^*)$  作为  $\mathcal{A}$  的 KEM 挑战. 最终,  $\mathcal{B}$  输出  $\mathcal{A}$  的猜测  $\beta'$ .

容易验证,  $\mathcal{B}$  在  $\text{Game}_1$  中的模拟是完美的. 因此  $\mathcal{B}$  打破  $R_L$  伪随机性的优势与  $\text{Adv}_{\mathcal{A}}^{\text{Game}_1}$  相同. 断言得证!  $\square$

综上, 定理得证!  $\square$

### 4.3.4 基于 EHPS 构造 IND-CCA KEM

我们首先从安全归约的角度分析基于 EHPS 构造 IND-CCA KEM 的难点. EHPS 模拟模式下的  $sk'$  可以在不知晓采样实例  $x$  随机数的情况下正确计算出相应的哈希证明, 但无法提取出证据, 因此归约算法无法应答解密询问. 因此, 为了构造 IND-CCA 的 KEM, 需要赋予 EHPS 更丰富的功能.

PKE/KEM 的选择密文安全游戏是“全除一”(ABO, all-but-one) 式的一  $\mathcal{A}$  可以发起除挑战密文  $x^*$  以外的任意解密/解封装询问. Wee [Wee10] 引入了量身定制的 ABO-EHPS.

#### 定义 4.12 (ABO-EHPS)

ABO-EHPS 与 EHPS 的定义差别集中在模拟模式, 真实模式下完全相同. 与 EHPS 相比, ABO-EHPS 在模拟模式下的功能更加丰富.

- $\text{KeyGen}'(pp, x^*)$ : 以公开参数  $pp$  和  $x^* \in L$  为输入, 输出  $(pk, sk')$ .
- $\text{PrivEval}(sk', x^*)$ : 以私钥  $sk'$  和  $x^*$  为输入, 输出证明  $\pi^* = H_{pk}(x^*)$ .
- $\text{Ext}'(sk', x, \pi)$ : 以私钥  $sk'$ 、 $x \neq x^*$  和  $\pi \in \Pi$  为输入, 输出证据  $w \in W$ . 正确性的要求是:

$$\pi = H_{pk}(x) \iff (x, \text{Ext}'(sk', x, \pi)) \in R_L$$

$\text{KeyGen}'$  算法以预先嵌入的点  $x^*$  为输入, 输出相应的密钥对  $(pk, sk')$ . ABO 的含义是模拟模式中的  $sk'$  具备以下功能:

- “一除全”哈希求值 (one-out-all evaluation):  $sk'$  可以计算  $x^*$  的哈希值  $H_{pk}(x^*)$ .
- “全除一”证据抽取 (all-but-one extraction):  $sk'$  可以从除  $x^*$  以外的点  $x$  和相应的证明中正确抽取证据  $\text{Ext}'(sk', x, \pi)$ .



#### 注记 4.12

模拟模式下  $sk'$  的功能在 CCA 安全归约中起到如下作用:

- “一除全”哈希求值允许归约算法  $\mathcal{R}$  生成挑战密文  $c^* = (x^*, \pi^*)$ .
- “全除一”证据抽取允许归约算法  $\mathcal{R}$  回答所有合法的解封装询问  $c \neq c^*$ .



Wee [Wee10] 展示了如何基于 EHPS 构造 ABO-EHPS.

#### 构造 4.14 (基于 EHPS 的 ABO-EHPS 构造)

起点: 二元关系  $R_L$  的 EHPS

设计目标: 二元关系  $R_L$  的 ABO-EHPS

设计思路: 不妨设  $L$  中每个实例均可编码为  $n$  长的比特串, 利用 DDN 结构 [DDN91] 实现 ABO 功能. 具体构造如下:

- $\text{Setup}(1^\kappa)$ : 运行  $pp \leftarrow \text{EHPS.Setup}(1^\kappa)$ ;
- $\text{KeyGen}(pp)$ : 独立运行  $\text{EHPS.KeyGen}(pp)$  算法  $2n$  次, 生成  $\{(pk_{i,b}, sk_{i,b})\}_{i \in [n], b \in \{0,1\}}$ , 输出公钥  $pk = \{pk_{i,0}, pk_{i,1}\}_{i \in [n]}$  和私钥  $sk = \{sk_{i,0}, sk_{i,1}\}_{i \in [n]}$ .
- $\text{PubEval}(pk, x, r)$ : 对所有的  $i \in [n]$  计算  $\pi_i \leftarrow \text{EHPS.PubEval}(pk_{i,x_i}, x, r)$ , 输出  $\pi = (\pi_1, \dots, \pi_n)$ .
- $\text{Ext}(sk, x, \pi)$ : 对所有的  $i \in [n]$  计算  $w_i \leftarrow \text{EHPS.Ext}(sk_{i,x_i}, x, \pi_i)$ , 如果所有结果一致则输出, 否则返回  $\perp$ .
- $\text{KeyGen}'(pp, x^*)$ : 独立运行  $\text{EHPS.KeyGen}'(pp)$  算法  $n$  次生成  $\{(pk_{i,x_i^*}, sk_{i,x_i^*})\}_{i \in [n]}$ , 独立运行  $\text{EHPS.KeyGen}(pp)$  算法  $n$  次生成  $\{(pk_{i,1-x_i^*}, sk_{i,1-x_i^*})\}_{i \in [n]}$ , 输出  $pk = \{pk_{i,0}, pk_{i,1}\}_{i \in [n]}$  and  $sk' = \{sk_{i,0}, sk_{i,1}\}_{i \in [n]}$ .
- $\text{PrivEval}(sk', x^*)$ : 对所有的  $i \in [n]$  计算  $\pi_i \leftarrow \text{EHPS.PrivEval}(sk_{i,x_i^*}, x^*)$ , 输出  $\pi = (\pi_1, \dots, \pi_n)$ .
- $\text{Ext}'(sk', x, \pi)$ : 对所有满足  $x_i^* = x_i$  的索引  $i \in [n]$  验证  $\pi_i = \text{EHPS.PrivEval}(sk_{i,x_i}, x_i)$  是否成立, 如

果否则输出  $\perp$ , 如果是则继续对所有满足  $x_i^* \neq x_i$  的索引  $i \in [n]$  计算  $\text{EHPS.Ext}(sk_{i,x_i}, x, \pi_i)$ , 如果提取结果一致则输出, 否则输出  $\perp$ .



ABO-EHPS 真实模式下算法的正确性由 EHPS 对应算法保证.

$$n = 3$$

$pk_{1,0}$ $sk_{1,0}$	$pk_{2,0}$ $sk_{2,0}$	$pk_{3,0}$ $sk_{3,0}$
$pk_{1,1}$ $sk_{1,1}$	$pk_{2,1}$ $sk_{2,1}$	$pk_{3,1}$ $sk_{3,1}$

图 4.21: 真实模式下  $n = 3$  时密钥结构图示

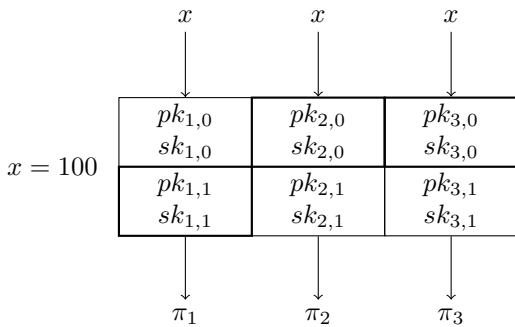


图 4.22: 真实模式下  $x = 010$  时哈希证明计算图示

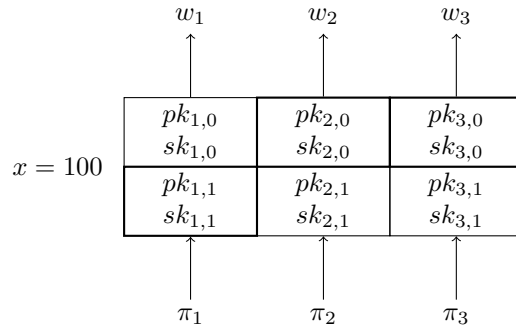


图 4.23: 真实模式下  $x = 010$  时证据提取图示

ABO-EHPS 模拟模式下算法的正确性由 DDN 结构和 EHPS 对应算法保证. ABO-EHPS 两种模式下公钥分布的统计不可区分性由 EHPS 两种模式下公钥分布的统计不可区分性与各公钥分量生成的独立性保证.

$$x^* = 010$$

$pk_{1,0}$ $sk'_{1,0}$	$pk_{2,0}$ $sk_{2,0}$	$pk_{3,0}$ $sk'_{3,0}$
$pk_{1,1}$ $sk_{1,1}$	$pk_{2,1}$ $sk'_{2,1}$	$pk_{3,1}$ $sk_{3,1}$

图 4.24: 模拟模式下  $n = 3, x^* = 010$  时的密钥生成

基于 ABO-EHPS 设计 IND-CCA KEM 的方式与构造 4.13 完全相同. KEM 构造的正确性由 ABO-EHPS 的正确性和  $R_L$  的射影性保证, 安全性由以下定理保证.

#### 定理 4.14

如果  $R_L$  是单向的, 那么 KEM 是 IND-CCA 安全的.



**证明** 证明的思路仍然是首先由真实模式切换到模拟模式, 再在模拟模式下利用零知识性证明安全性. 证明的要点是保证两种模式下对解密谕言机  $\mathcal{O}_{\text{decap}}$  回复的一致性. 以下通过游戏序列完成定理证明:

**Game<sub>0</sub>**: 对应真实的游戏.  $\mathcal{CH}$  在真实模式下运行 ABO-EHPS 与敌手  $\mathcal{A}$  交互.

- 初始化:  $\mathcal{CH}$  计算  $pp \leftarrow \text{Setup}(1^\kappa)$ ,  $(pk, sk) \leftarrow \text{KeyGen}(\lambda)$ , 将  $pp$  和  $pk$  发送  $\mathcal{A}$ .
- 挑战:  $\mathcal{CH}$  按照以下步骤生成挑战
  1. 随机采样  $(x^*, w^*) \leftarrow \text{SampRel}(r^*)$ ;
  2. 通过  $\text{PubEval}(pk, x^*, r^*)$  公开计算  $\pi^* \leftarrow H_{pk}(x^*)$ ;
  3. 计算  $k_0^* \leftarrow \text{hc}(w^*)$ , 随机采样  $k_1^* \xleftarrow{R} K$ ;
  4. 随机选取  $\beta \xleftarrow{R} \{0, 1\}$ , 将  $(c^* = (x^*, \pi^*), k_\beta^*)$  发送给  $\mathcal{A}$  作为挑战.

不知晓随机数, 使用  $\text{PrivEval}(sk', x^*)$  计算

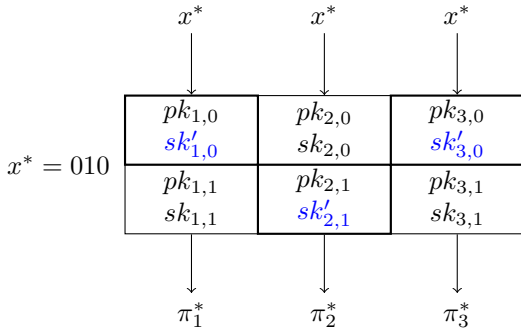


图 4.25: 模拟模式下  $x^* = 010$  时哈希证明计算

知晓随机数, 使用  $\text{PubEval}(pk, x, r)$  计算

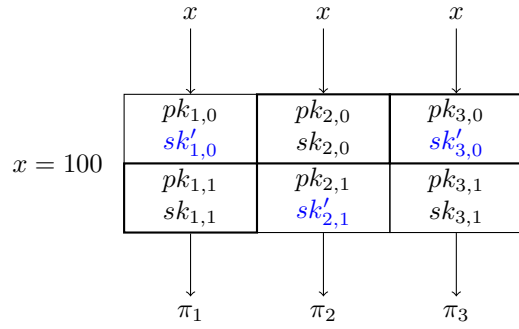
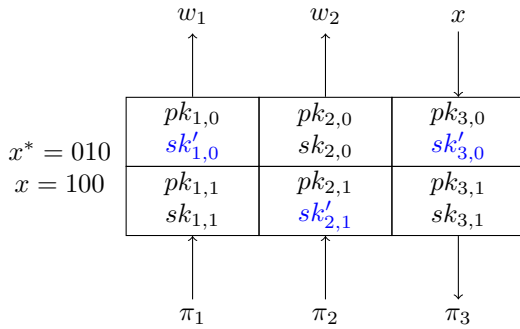


图 4.26: 模拟模式下  $x = 100$  时哈希证明计算



$$\begin{aligned} w_1 &\leftarrow \text{EHPS.Ext}(sk_{1,1}, \pi_1) \\ w_2 &\leftarrow \text{EHPS.Ext}(sk_{2,0}, \pi_2) \\ \text{check } w_1 &\stackrel{?}{=} w_2 \\ \text{check } \pi_3 &\stackrel{?}{=} \text{EHPS.PrivEval}(sk'_{3,0}, x) \end{aligned}$$

图 4.27: 模拟模式下  $x = 100$  时证据提取过程

- 解封装询问  $c = (x, \pi) \neq c^*$ : 计算  $w \leftarrow \text{Ext}(sk, x, \pi)$ , 如果  $(x, w) \in R_L$  则输出  $\text{hc}(w)$ , 否则输出  $\perp$ .
- 应答:  $\mathcal{A}$  输出对  $\beta$  的猜测  $\beta'$ ,  $\mathcal{A}$  成功当且仅当  $\beta' = \beta$ .

为了利用 ABO-EHPS 的零知识性论证  $\pi^*$  和解封装询问不额外泄漏关于  $w^*$  的信息, 需要将 ABO-EHPS 由真实模式切换到模拟模式. 为此, 先引入以下游戏作为过渡.

**Game<sub>1</sub>**: 与 **Game<sub>0</sub>** 完全相同, 惟一的区别是  $\mathcal{CH}$  将  $(x^*, w^*) \leftarrow \text{SampRel}(r^*)$  由挑战阶段提前至初始化阶段. 显然, 该变化不会对敌手的视图有任何改变. 因此有:

$$\text{Game}_0 \equiv \text{Game}_1$$

**Game<sub>2</sub>**: 本游戏对解封装应答方式稍加改动, 以便于后续游戏将密文的 ABO 解封装询问转化为 ABO-EHPS 相对于  $x^*$  的 ABO 证据抽取. 对于解封装询问  $c = (x, \pi) \neq c^*$ ,  $\mathcal{CH}$  应答如下:

- $x = x^* \wedge \pi \neq \pi^*$ : 直接返回  $\perp$ .
- $x \neq x^*$ : 计算  $w \leftarrow \text{Ext}(sk, x, \pi)$ , 如果  $(x, w) \in R_L$  则返回  $\text{hc}(w)$ , 否则返回  $\perp$ .

由于  $H_{pk}$  是确定性算法, 因此 **Game<sub>2</sub>** 与 **Game<sub>1</sub>** 中的解封装应答完全相同.

**Game<sub>3</sub>**:  $\mathcal{CH}$  在模拟模式下运行 ABO-EHPS 与敌手  $\mathcal{A}$  交互.

- 初始化:  $\mathcal{CH}$  与上一游戏的区别在于通过  $(pk, sk') \leftarrow \text{KeyGen}'(pp, x^*)$  生成密钥对.
- 挑战:  $\mathcal{CH}$  与上一游戏的区别在于通过  $\text{PrivEval}(sk', x^*)$  计算  $\pi^* \leftarrow H_{pk}(x^*)$ .
- 解封装询问  $c = (x, \pi) \neq c^*$ :  $\mathcal{CH}$  应答如下
  - $x = x^* \wedge \pi \neq \pi^*$ : 直接返回  $\perp$ .
  - $x \neq x^*$ : 计算  $w \leftarrow \text{Ext}'(sk', x, \pi)$ , 如果  $(x, w) \in R_L$  则返回  $\text{hc}(w)$ , 否则返回  $\perp$ .

基于以下事实, 我们有:  $\text{Game}_2 \approx_s \text{Game}_3$

- $\text{KeyGen}(pp)[1] \approx_s \text{KeyGen}'(pp, x^*)[1]$
- $\text{PubEval}(pk, x^*, r^*) = H_{pk}(x^*) = \text{PrivEval}(sk', x^*)$

- 对于解封装询问  $c = (x, \pi)$ : 当  $x = x^*$  时, 均返回  $\perp$ ; 当  $x \neq x^*$  时, ABO-EHPS 真实模式和模拟模式的正确性以及解封装算法“提取-检验”的设计保证了应答一致.

**断言 4.10**

如果  $R_L$  是单向的, 那么  $\text{Adv}_{\mathcal{A}}^{\text{Game}_3} = \text{negl}(\kappa)$ .



**证明** 证明思路是如果存在  $\mathcal{A}$  以不可忽略的优势赢得  $\text{Game}_3$ , 那么可以构造出  $\mathcal{B}$  以不可忽略的优势打破 hc 的伪随机性, 从而与  $R_L$  的单向性假设冲突. 给定关于 hc 的伪随机性挑战  $pp$  和  $(x^*, k_\beta^*)$ , 其中  $(x^*, w^*) \leftarrow \text{SampRel}(r^*)$ ,  $\mathcal{B}$  模拟  $\text{Game}_3$  中的挑战者  $\mathcal{CH}$  与  $\mathcal{A}$  交互, 目标是猜测  $\beta$ .

- $\mathcal{B}$  运行 ABO-EHPS 的模拟模式与  $\mathcal{A}$  进行交互, 其在初始化阶段不再采样  $x^*$  而是直接嵌入接收到的  $x^*$ , 在挑战阶段将 hc 的挑战  $(x^*, k_\beta^*)$  作为  $\mathcal{A}$  的 KEM 挑战. 最终,  $\mathcal{B}$  输出  $\mathcal{A}$  的猜测  $\beta'$ .

容易验证,  $\mathcal{B}$  在  $\text{Game}_3$  中的模拟是完美的. 因此  $\mathcal{B}$  打破 hc 伪随机性的优势与  $\text{Adv}_{\mathcal{A}}^{\text{Game}_3}$  相同. 断言得证!

综上, 定理得证!

Wee [Wee10] 展示了 ABO-EHPS 蕴含 ATDR.

**构造 4.15 (基于 ABO-EHPS 的 ATDR 构造)**

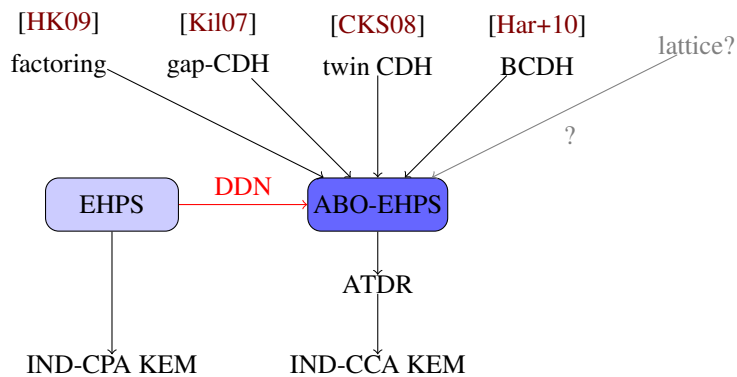
- $\text{Setup}(1^\kappa)$ : 运行  $pp \leftarrow \text{ABO-EHPS.Setup}(1^\kappa)$ .
- $\text{KeyGen}(pp)$ : 运行  $(pk, sk) \leftarrow \text{ABO-EHPS.KeyGen}(pp)$ , 令  $pk$  为求值公钥  $ek$ ,  $sk$  为求逆陷门  $td$ .
- $\text{Sample}(pk; r)$ : 运行  $(x, w) \leftarrow \text{SampRel}(r)$ , 通过  $\text{ABO-EHPS.PubEval}(pk, x, r)$  计算  $\pi \leftarrow H_{pk}(x)$ , 输出  $(w, (x, \pi))$ .
- $\text{TdInv}(td, (x, \pi))$ : 计算  $w \leftarrow \text{ABO-EHPS.Ext}(sk, (x, \pi))$ , 如果  $(x, w) \in R$  则返回  $w$ , 否则返回  $\perp$ .



上述 ATDR 构造的自适应单向性由 ABO-EHPS 的性质  $R_L$  的单向性保证. 该构造也在更抽象的层面解释了基于 ABO-EHPS 设计 CCA 安全 KEM 的实质是在构造 ATDR.

**4.3.5 小结**

EHPS 的重要理论意义在于它阐释统一了一大类标准模型下的基于计算性假设的 IND-CCA PKE 方案 [Kil07; CKS08; HK09; Har+10]. 绝大多数标准模型下的 PKE 构造都可纳入 EHPS 和 HPS 的设计范式, 这也从公钥加密的角度再次证实了零知识证明的强大威力. 目前尚未知晓如何基于 lattice 上的困难问题构造 EHPS.

**4.3.5.1 HPS 与 EHPS 的分析比较**

相同点

- 均可看成指定验证者的零知识证明系统 (DV-NIZK).
- 证明的形式是哈希值.

不同点

- HPS 是标准的证明系统, 而 EHPS 是知识的证明系统.

- HPS 中哈希函数族  $H_{sk}$  由私钥索引, EHPS 中哈希函数族  $H_{pk}$  有公钥索引.
- 在基于 HPS 的 PKE 构造中, 密文  $c$  是实例  $x$ , 会话密钥  $k$  是证明  $\pi$ .
  - HPS 的正确性保证了 PKE 的正确性
  - HPS 的合理性 (哈希函数的平滑性、一致性) 与 SMP 问题的困难性保证了 PKE 的安全性, 在证明过程中, 挑战实例需要从语言  $L$  上切换到语言外  $X \setminus L$ .
- 在基于 EHPS 的 PKE 构造中, 密文  $c$  由实例  $x$  和证明  $\pi$  组成, 会话密钥  $k$  是证据  $w$ .
  - EHPS 的知识提取性质保证了 PKE 的正确性
  - EHPS 的零知识性和二元关系的单向性保证了 PKE 的安全性, 在证明过程中, EHPS 需要由真实模式切换为模拟模式.

## 4.4 程序混淆类

只要代码足够乱, 就没人发现我写错。

— 网上·佚名

### 4.4.1 背景知识

程序混淆 (program obfuscation) 是一种编译的方法技术, 它将容易理解的源程序转化成难以理解的形式, 同时保持原有功能性不变. 程序混淆概念起源于上世界 70 年代的代码混淆领域, 在软件保护领域 (如软件水印、防逆向工程) 有着广泛的应用, 然而一直缺乏严格的安全定义。

```

319 int KDF(Zz2 x, char *s)
320 { // Hash an fp2 to an n-byte string
321     sha256 sh;
322     Big a,b;
323     int m;
324
325     sha256_init(&sh);
326     x.get(a,b);
327
328     while (a>0)
329     {
330         m=a%256;
331         sha256_process(&sh,m);
332         a/=256;
333     }
334     while (b>0)
335     {
336         m=b%256;
337         sha256_process(&sh,m);
338         b/=256;
339     }
340     sha256_hash(&sh,s);

```

```

#include<stdio.h> #include<string.h> main(){
  +=acgo\1777\isp -.\08^8)N362K40*42M(*0ID5783
  fgets(1+45,964,stdin)){*1=0[ strlen(0)[0-1]=0
  while(*0)switch((+1&&issalnum(*0))-+1){case-
  strapa(0,1+12)+1)-2,0=34;while(*1&3&&(0=(0-1
  putchar(0&35*1&61)|1/ 1=switch( 1 , 0 , 44
  break;case 1: ;}*1=(+0&31)[1-15+(+0&61)*32];
  (*1+=1+32>>1)>35;case 0:putchar(++0,32);};

```

图 4.28: 程序混淆

Barak 等 [Bar+01] 首次将程序混淆引入密码学领域, 将程序从狭义的代码泛化为广义的算法, 同时提出了虚拟黑盒 (VBB, virtual black-box) 混淆的严格定义。

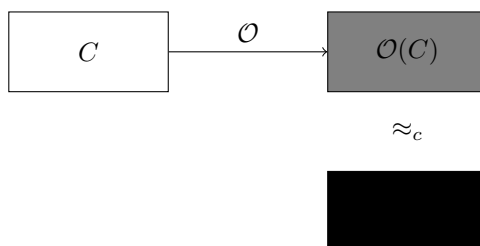


图 4.29: 虚拟黑盒混淆

我们回顾不可区分混淆 (indistinguishability obfuscator,  $iO$ ) [Gar+13] 如下。

#### 定义 4.13 (虚拟黑盒混淆)

我们称一个 uniform PPT machine  $iO$  是电路簇  $\{C_\kappa\}$  的不可区分混淆器当且仅当其满足以下两个条件:

- 功能保持: 对于任意安全参数  $\kappa \in \mathbb{N}$ 、任意的  $C \in \mathcal{C}_\kappa$  和所有输入  $x \in \{0,1\}^*$  有:

$$\Pr[C'(x) = C(x) : C' \leftarrow iO(\kappa, C)] = 1$$

- 虚拟黑盒混淆: 存在 PPT 的模拟器  $S$ , 对于任意  $C \in \{C_\kappa\}$ , 对于任意 PPT 敌手  $\mathcal{A}$ , 我们有:

$$\mathcal{A}(O(C)) \approx_c S^C$$

其中公式左边表示  $\mathcal{A}$  的视图, 公式右边表示  $S$  在通过对  $C$  进行黑盒访问所输出的视图。



**笔记** 虚拟黑盒混淆的定义是基于模拟的游戏, 刻画的是 PPT 敌手从混淆程序  $O(C)$  中获取的任何信息不会比黑盒访问  $C$  获得的信息更多. 换言之, 掌握  $O(C)$  的敌手视图可以由模拟器通过黑盒访问  $C$  模拟得出. VBB 试图隐



藏程序  $C$  的所有细节. 比如  $C$  以平方差公式计算  $x^2 - 1$ , 即  $C(x) = (x+1)(x-1)$ . 那么敌手在获得  $\mathcal{O}(C)$  后, 掌握的所有信息等价输入/输出元组  $(x, x^2 - 1)$ , i.e.,  $(1, 0), (2, 3), (3, 8), \dots$

VBB 的混淆定义强到极致, 因此在密码学中应用起来颇为简单直观. 事实上, 在 1976 年 Diffie 和 Hellman 的划时代论文 [DH76] 中, 就已经提出了利用混淆器将对称加密方案编译为公钥加密方案的想法 (如图 4.30):

1. 将对 SKE 的加密算法  $\text{Enc}(sk, m, r)$  中的第一个输入 **hardwired** 进电路, 得到  $\text{Enc}_{sk}(m, r)$ ;
2. 利用混淆器编译  $\text{Enc}_{sk}(\cdot, \cdot)$ , 将得到的混淆程序作为公钥  $pk$ .

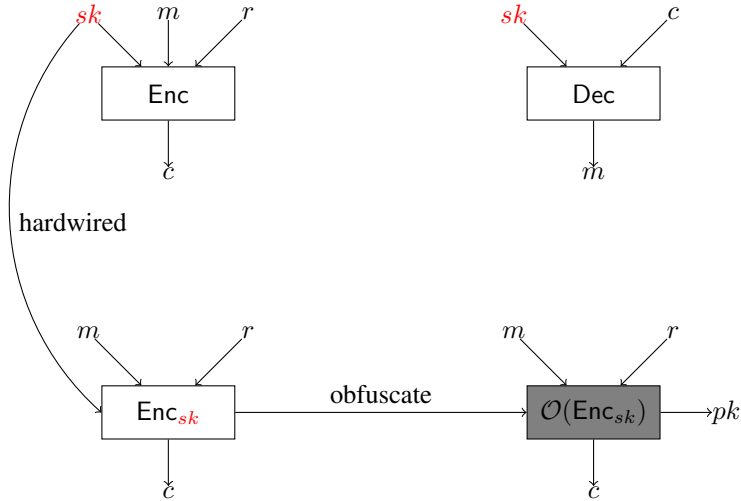


图 4.30: SKE  $\Rightarrow$  PKE via obfuscation

VBB 的安全定义至强, Barak 等 [Bar+01] 指出 VBB “too good to be true!”—不存在针对任意电路 (通用, general-purpose) 的 VBB. VBB 因为安全太强以至于不存在, Garg 等 [Gar+13] 降低了安全性要求, 引入了不可区分混淆 (indistinguishability obfuscator,  $i\mathcal{O}$ ).

#### 定义 4.14 (不可区分混淆 ( $i\mathcal{O}$ ))

我们称一个 uniform PPT machine  $i\mathcal{O}$  是电路簇  $\{C_\kappa\}$  的不可区分混淆器当且仅当其满足以下两个条件:

- 功能保持: 对于任意安全参数  $\kappa \in \mathbb{N}$ 、任意的  $C \in \mathcal{C}_\kappa$  和所有输入  $x \in \{0, 1\}^*$  有:

$$\Pr[C'(x) = C(x) : C' \leftarrow i\mathcal{O}(\kappa, C)] = 1$$

- 不可区分混淆: 对于任意 PPT 敌手  $(\mathcal{S}, \mathcal{D})$ , 存在关于安全参数的可忽略函数  $\alpha$  使得: 如果  $\Pr[\forall x, C_0(x) = C_1(x) : (C_0, C_1, aux) \leftarrow \mathcal{S}(\kappa)] \geq 1 - \alpha(\kappa)$ , 那么我们有:

$$|\Pr[\mathcal{D}(aux, i\mathcal{O}(\kappa, C_0)) = 1] - \Pr[\mathcal{D}(aux, i\mathcal{O}(\lambda, C_1)) = 1]| \leq \alpha(\kappa)$$

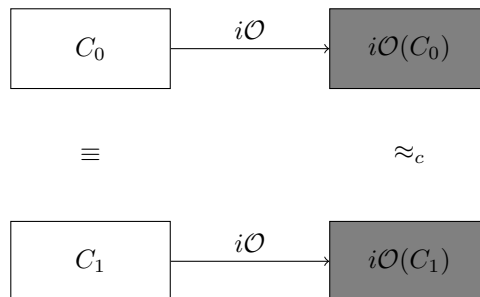


图 4.31: 不可区分混淆示意图

### 注记 4.13

- 不可区分混淆的定义类似加密方案的不可区分性, 对于任意功能相同的电路  $C_0$  和  $C_1$ , 均有  $i\mathcal{O}(C_0) \approx_c i\mathcal{O}(C_1)$ . 这里, 我们可以把电路  $C$  类比为消息,  $i\mathcal{O}$  类比为加密算法. 与 VBB 试图隐藏电路的所有信息不同,  $i\mathcal{O}$  只试图隐藏电路的部分信息: 比如  $C_0(x) = (x+1)(x-1)$ ,  $C_1(x) = (x+2)(x-2) + 3$ , 那么如果混淆后的程序均是  $x^2 - 1$  即可满足不可区分安全性. 非严格的说,  $i\mathcal{O}$  试图在以统一的方式完成同质的计算.
- 在上述定义中, 条件  $\Pr[\forall x, C_0(x) = C_1(x) : (C_0, C_1, aux) \leftarrow \mathcal{S}(\kappa)] \geq 1 - \alpha(\kappa)$  并不意味着  $C_0$  和  $C_1$  存在差异输入 (differing-inputs), 而指的是  $C_0$  和  $C_1$  以极高的概率功能性完全相同, 这一点体现在概率空间定义在  $\mathcal{S}$  的随机带而与  $x$  无关.
- $aux$  表示  $\mathcal{S}$  在采样  $C_0, C_1$  过程中得到的任意信息, 用于辅助  $\mathcal{D}$  区分  $i\mathcal{O}(C_0)$  和  $i\mathcal{O}(C_1)$ .



**差异输入混淆.** 如果在上述  $i\mathcal{O}$  定义中, 将  $\mathcal{S}$  所采样两个电路的要求由功能性完全相同放宽为允许存在差异输入, 则得到的是更强的混淆器, 称为差异输入混淆 ( $di\mathcal{O}$ , differing-input obfuscation). [BCP14] 中给出了正面结果: 证明了  $i\mathcal{O}$  蕴含多项式级别差异输入规模的  $di\mathcal{O}$ . [BSW16] 中给出了负面结果: 证明了亚指数安全 (sub-exponentially secure) 的单向函数存在, 则针对无界输入 Turing machine (TMs with unbounded inputs) 亚指数安全的  $di\mathcal{O}$  不存在.

我们再把注意力转回不可区分混淆. 如上所述, VBB 易用但对于通用电路并不存在,  $i\mathcal{O}$  弱化了安全要求, 从而有了基于合理困难性假设的构造. 安全性弱化后  $i\mathcal{O}$  是否还有着强大的威力? 如何去应用呢? 直观上: 混淆后的程序既可以保持功能性, 又能够在某种程度上隐藏常量. 常量皆程序. 在密码学中, 公钥和私钥均可以看做一段程序, 其中硬编码 (hardwired) 原本的公钥和私钥作为常量: 比如加密就是以明文和随机数为输入, 运行“公钥程序”, 输出密文; 解密就是以密文为输入, 运行“解密程序”, 输出明文. 混淆在密码学中的一类强大应用就是完成从 Minicrypt 到 Cryptomania 的穿越, 因为借助混淆, 可以在不泄漏秘密的情况下以公开的方式执行某个任务.

- 保持功能性  $\Rightarrow$  确保密码方案的功能性
- 在某种程度上隐藏常量 (对应需要保护的秘密)  $\Rightarrow$  确保密码方案的安全性

### 注记 4.14

混淆的威力强大如魔法, 其力量的来源在于对底层密码组件的调用方式是非黑盒的 (non-black-box), 因此可以绕过黑盒意义下的不可能结果 (black-box impossibilities).



在  $i\mathcal{O}$  提出后最初的一段时间, 应用只局限于属性加密 (ABE). 原因是应用  $i\mathcal{O}$  设计密码方案并非易事, 需要解决的技术难题是精准的隐藏“部分信息”. 2014 年, Sahai 和 Waters [SW14] 创造性的发展了可穿孔编程技术 (puncture program technique), 以此给出了应用  $i\mathcal{O}$  的范式, 展示了  $i\mathcal{O}$  的巨大威力—结合单向函数和  $i\mathcal{O}$  重构了几乎所有的密码组件, 包括公钥加密/密钥封装、可否认加密、数字签名、单向陷门函数、非交互式零知识证明、不经意传输等.

## 4.4.2 受限伪随机函数

以下首先介绍可穿孔编程技术的关键密码组件—受限伪随机函数 (constrained PRF). 对于标准的伪随机函数  $F : K \times X \rightarrow Y$ , 密钥  $k$  不支持代理, 因此求值方式是“完全或无”(all-or-nothing):

- 知晓密钥  $k$  时, 可以对定义域中的所有点  $x \in X$  进行函数求值, 得到  $F_k(x)$ .
- 不知晓密钥  $k$  时,  $F_k(\cdot)$  是伪随机的.

三组科学家 [BW13; Kia+13; BGI14] 独立并行的提出了受限伪随机函数 (constrained PRF). 在受限伪随机函数中,  $k$  可以进一步代理得到受限密钥, 受限密钥可以对定义域中的部分输入进行求值. 正式的定义如下:

**定义 4.15 (受限伪随机函数 (constrained PRF))**

受限伪随机函数包含以下多项式时间算法:

- $\text{Setup}(1^\kappa)$ : 以安全参数为输入, 输出公开参数  $pp$ , 其中包含了函数  $F: K \times X \rightarrow Y$  的描述以及电路族  $\mathcal{C} = \{f: X \rightarrow \{0, 1\}\}$  的描述.
- $\text{KeyGen}(pp)$ : 随机采样密钥  $k \xleftarrow{R} K$ .
- $\text{Constrain}(k, f)$ : 以密钥  $k$  和  $c \in \mathcal{C}$  为输入, 输出受限密钥  $k_f$ .
- $\text{Eval}(k/k_f, x) = F_k(x)$ : 以密钥  $k$  或受限密钥  $k_c$  和  $x \in X$  为输入, 当第一输入为  $k$  时输出  $F_k$ , 当第一输入为  $k_c$  时, 如果  $f(x) = 1$  则输出  $F_k(x)$ , 否则输出  $\perp$ .



**安全性.** 受限伪随机函数要求对于没有被受限密钥和求值询问覆盖的输入, 其输出仍然是伪随机的. 定义敌手  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$  的优势函数如下:

$$\Pr \left[ \begin{array}{l} pp \leftarrow \text{Setup}(1^\lambda); \\ k \leftarrow \text{KeyGen}(pp); \\ \beta = \beta': (x^*, \text{state}) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\text{eval}}, \mathcal{O}_{\text{constrain}}}(pp); \\ \beta \xleftarrow{R} \{0, 1\}, y_0^* = F_k(x^*), y_1^* \xleftarrow{R} Y; \\ \beta' \leftarrow \mathcal{A}_2^{\mathcal{O}_{\text{eval}}, \mathcal{O}_{\text{constrain}}}(\text{state}, y_\beta^*); \end{array} \right] - \frac{1}{2}.$$

其中  $\mathcal{O}_{\text{eval}}$  表示求值预言机, 以  $x \in X$  为输入, 返回  $F_k(x)$ ;  $\mathcal{O}_{\text{constrain}}$  表示受限密钥预言机, 以  $c \in \mathcal{C}$  为输入, 输出  $k_c$ . 在安全试验过程中, 挑战者隐式的维护集合  $H$  以记录被已发起的受限密钥询问和求值询问覆盖的定义域中元素. 为了避免定义无意义,  $\mathcal{A}_1$  在第一阶段被禁止选择  $H$  中的元素作为挑战点,  $\mathcal{A}_2$  在第二阶段被禁止发起能够覆盖挑战点的受限密钥询问和求值询问. 如果任意的 PPT 敌手  $\mathcal{A}$  在上述游戏中的优势函数均为可忽略函数, 则称受限伪随机函数是安全的.

**注记 4.15**

对任意  $\mathcal{C}$ , 均存在平凡的受限伪随机函数构造: 受限密钥生成函数通过枚举的方式生成受限密钥, 即  $k_f = \{y = F_k(x)\}_{f(x)=1}$ . 为了排除此类平凡的构造, 我们要求受限密钥是紧致的, 即对于  $\forall f \in \mathcal{C}$ , 均有  $|k_f| = \kappa^{O(1)}$ .



Sahai 和 Waters [SW14] 引入了受限伪随机函数的特例——可穿孔伪随机函数 (puncturable PRF), 正式定义如下:

**定义 4.16 (可穿孔伪随机函数 (puncturable PRF))**

可穿孔伪随机函数包含以下多项式时间算法:

- $\text{Setup}(1^\kappa)$ : 以安全参数为输入, 输出公开参数  $pp$ , 其中包含了函数  $F: K \times X \rightarrow Y$  的描述和电路族  $\mathcal{C} = \{f_{x^*}: X \rightarrow \{0, 1\}\}_{x^* \in X}$  的描述.  $f_{x^*}(\cdot)$  的具体定义是  $f_{x^*}(x) = \neg x^* \stackrel{?}{=} x$ . 为了表述简洁, 以下在不引起混淆的情况下使用  $x^*$  表征  $f_{x^*}$ .
- $\text{KeyGen}(pp)$ : 随机采样密钥  $k \xleftarrow{R} K$ .
- $\text{Puncture}(k, x^*)$ : 以密钥  $k$  和  $x^* \in X$  为输入, 输出受限密钥  $k_{x^*}$ .
- $\text{Eval}(k/k_{x^*}, x)$ : 以密钥  $k$  或受限密钥  $k_{x^*}$  和  $x \in X$  为输入, 当第一输入为  $k$  时输出  $F_k$ , 当第一输入为  $k_{x^*}$  时, 如果  $x \neq x^*$  时输出  $F_k(x)$ , 否则输出  $\perp$ .



可穿孔伪随机函数的安全定义直觉是对于没有被受限密钥覆盖的输入, 其输出仍然是伪随机的. 存在以下两种等价定义.

选择伪随机性. 定义敌手  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$  的优势函数如下:

$$\Pr \left[ \begin{array}{l} (x^*, state) \leftarrow \mathcal{A}_1(\kappa); \\ pp \leftarrow \text{Setup}(1^\lambda); \\ k \leftarrow \text{KeyGen}(pp); \\ k_{x^*} \leftarrow \text{Puncture}(k, x^*); \\ \beta \xleftarrow{\mathcal{R}} \{0, 1\}, y_0^* = F_k(x^*), y_1^* \xleftarrow{\mathcal{R}} Y; \\ \beta' \leftarrow \mathcal{A}_2(state, k_{x^*}, y_\beta^*); \end{array} \right] = \frac{1}{2}.$$

如果任意的 PPT 敌手  $\mathcal{A}$  在上述游戏中的优势函数均为可忽略函数, 则称可穿孔伪随机函数是选择伪随机的。

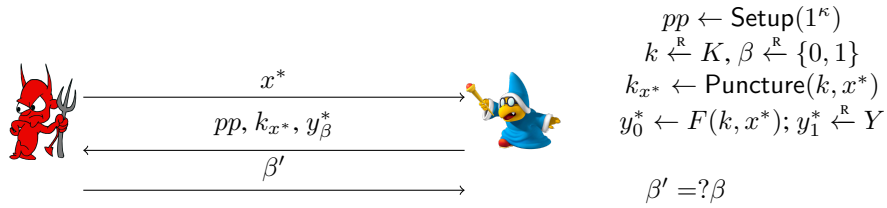


图 4.32: 可穿孔伪随机函数选择伪随机性示意图

弱伪随机性. 定义敌手  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$  的优势函数如下:

$$\Pr \left[ \begin{array}{l} pp \leftarrow \text{Setup}(1^\lambda); \\ k \leftarrow \text{KeyGen}(pp), x^* \xleftarrow{\mathcal{R}} X; \\ k_{x^*} \leftarrow \text{Puncture}(k, x^*); \\ \beta \xleftarrow{\mathcal{R}} \{0, 1\}, y_0^* = F_k(x^*), y_1^* \xleftarrow{\mathcal{R}} Y; \\ \beta' \leftarrow \mathcal{A}(pp, x^*, k_{x^*}, y_\beta^*); \end{array} \right] = \frac{1}{2}.$$

如果任意的 PPT 敌手  $\mathcal{A}$  在上述游戏中的优势函数均为可忽略函数, 则称可穿孔伪随机函数是弱伪随机的。

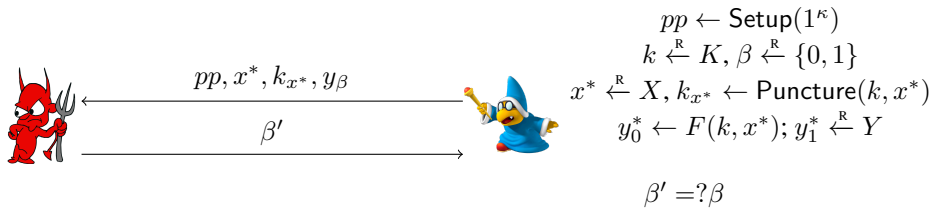


图 4.33: 可穿孔伪随机函数弱伪随机性示意图

文献 [CWZ18] 证明了可穿孔弱伪随机函数的两种安全定义等价。

#### 注记 4.16

在定义方面, 可穿孔伪随机函数中的受限密钥  $k_{x^*}$  求值功能是“全除一”的, 是 ABO 类型密钥. 在安全性方面, 可穿孔伪随机函数要求弱伪随机性, 即对于挑战者随机选择的点  $x^*$ , 即使拥有  $k_{x^*}$ ,  $F(x^*)$  在敌手的视图中依然伪随机, 即:

$$(pp, x^*, k_{x^*}, F_k(x^*)) \approx_c (pp, x^*, k_{x^*}, y^*), \text{ 其中 } y^* \xleftarrow{\mathcal{R}} Y$$

由于可穿孔伪随机函数的受限类型特殊, 因此安全定义相比一般的受限伪随机函数在形式上更加简单。

可穿孔伪随机函数可以通过 GGM 树形伪随机函数自然得出 [BW13; Kia+13; BGI14], 因此仍属于 Minicrypt.

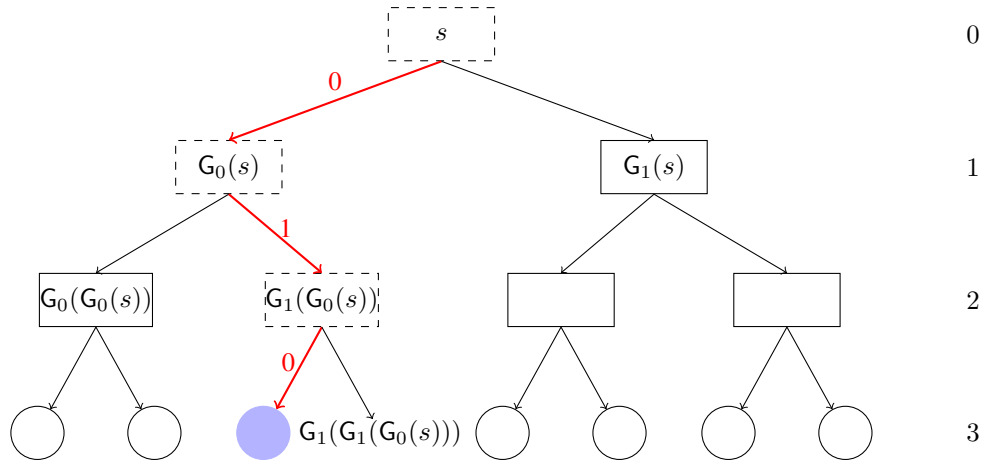
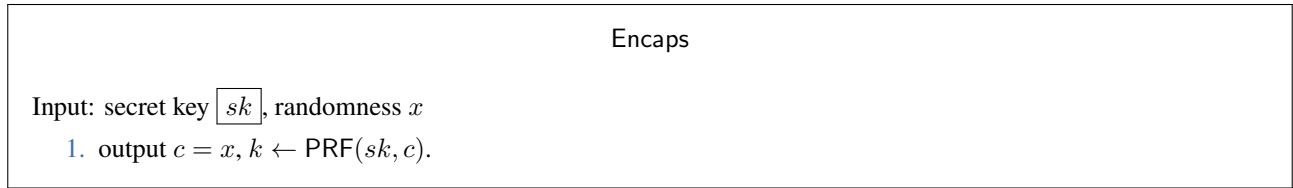


图 4.34: GGM construction:  $k_{010} = \{G_0(G_0(s)), G_1(G_0(G_0(s))), G_1(s)\}$

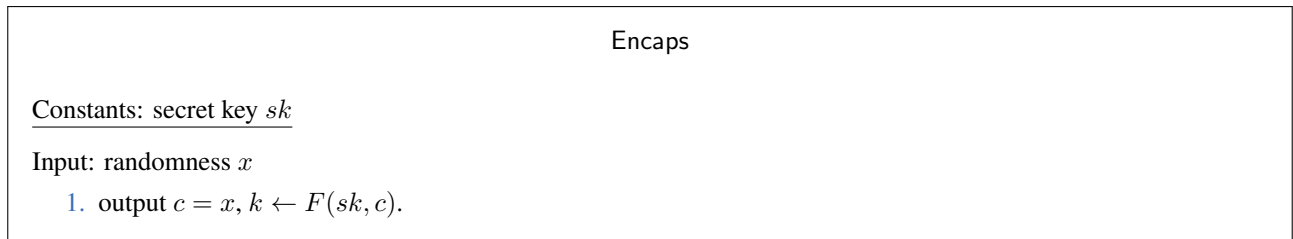
### 4.4.3 基于不可区分混淆的 KEM 构造

本章将逐步的展示如何基于  $i\mathcal{O}$  构造 KEM, 实现 Diffie-Hellman 当年的梦想.

**起点方案.** 首先将对称场景下基于 PRF 的 KEM 封装算法表达为程序的形式, 如下图所示.



再对程序进行微调, 将  $sk$  由输入变为 **hardwired** 的常量, 如下图所示:



由于通用的 VBB 并不存在, 因此尝使用  $i\mathcal{O}$  对程序混淆, 将混淆后的结果作为公钥

$$pk \leftarrow i\mathcal{O}(\text{Encaps})$$

**技术困难 1.** 在将 KEM 的 IND-CCA 安全性归约到 PRF 的伪随机性时, 会遇到以下矛盾点:

- 在构造层面, 归约算法  $\mathcal{R}$  需要掌握  $sk$  以生成  $pk$
- 为了让归约有意义, 归约算法  $\mathcal{R}$  不能掌握私钥  $sk$

观察到 KEM 的 IND-CCA 安全仅要求随机挑战密文  $c^*$  封装的会话密钥是伪随机的, 因此消除矛盾点的核心想法是使用可穿孔伪随机函数替代标准伪随机函数, 在挑战密文  $c^*$  处穿孔:

- 生成  $sk_{c^*}$  得以对  $c^*$  外的所有点求值, 同时保持  $F_{sk}(c^*)$  的伪随机性.
- 利用  $sk_{c^*}$  替代  $sk$  构建程序并混淆生成公钥.

## Encaps

Constants: secret key  $sk$ Input: randomness  $x$ 

1. output  $c = x, k \leftarrow F(sk, c)$ .

方案构造:  $pk \leftarrow i\mathcal{O}(\text{Encaps})$ 

## Encaps\*

Constants: secret key  $sk_{c^*}, c^*$ Input: randomness  $x$ 

1. output  $c = x, k \leftarrow F(sk_{c^*}, c)$ .

归约证明:  $pk \leftarrow i\mathcal{O}(\text{Encaps}^*)$ 

**技术困难 2.** 我们首先来分析归约证明中将要遇到的困难. 在模拟游戏中, 归约算法  $\mathcal{R}$  仅需要使用  $sk_{c^*}$  即可构建程序  $\text{Encaps}$ , 因此会话密钥  $k^* \leftarrow F(sk, c^*)$  的伪随机性可以归约到可穿孔伪随机函数的安全性上. 我们仍需证明敌手在真实游戏与模拟游戏中的视图不可区分. 在此过程中, 遇到的第一个障碍是由于在  $x^* := c^*$  处穿孔, 敌手可以通过观察程序在  $x^*$  的输出从而轻易区分真实游戏与模拟游戏:

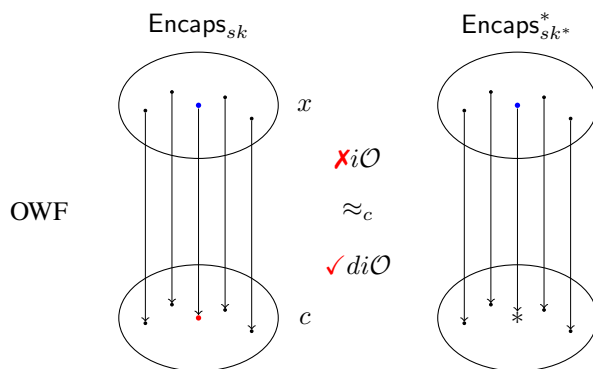
- 真实游戏:  $\text{Encaps}(x^*)$  返回  $k^*$  (已经不安全)
- 模拟游戏:  $\text{Encaps}^*(x^*)$  返回  $\perp$

以上设计不成立的根本原因是

- 密文的设定  $\boxed{c = x} \Rightarrow$  差异输入  $x^*$  将被挑战密文  $c^*$  直接暴露

为了隐藏差异输入, 初步的尝试为:

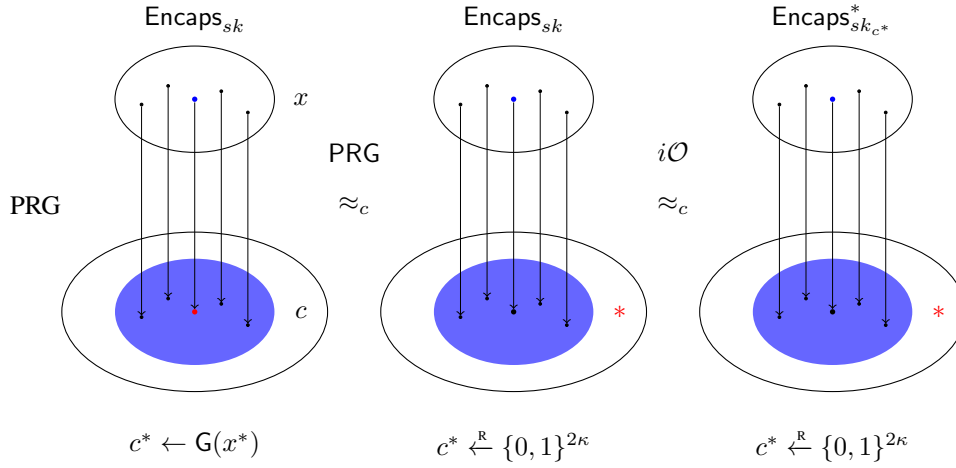
- $c = x \rightsquigarrow \boxed{c = f(x)}$ , 其中  $f$  是单向函数.



使用单向函数对挑战点进行隐藏并没有消除差异输入,  $\text{Encaps}_{sk}$  与  $\text{Encaps}_{sk_{c^*}}^*$  的输入输出行为存在不一致, 因此不满足  $i\mathcal{O}$  的应用条件, 需要使用更强的  $di\mathcal{O}$ .

消除差异输入的方法是将穿孔点  $c^*$  以敌手不可察觉的方式移到输入计算路径之外. 大致的技术路线是:

- 真实构造:  $c \leftarrow \text{OWF}(x) \rightsquigarrow \boxed{c \leftarrow G(x)}$ , 其中  $G$  是伪随机数发生器;
- 过渡游戏: 将  $c^* \leftarrow G(x^*)$  切换为  $c^* \xleftarrow{R} \{0, 1\}^{2\lambda}$ , 利用 PRG 的安全性保证切换不可察觉;
- 最终游戏: 利用  $sk_{c^*}$  替代  $sk$ , 利用  $i\mathcal{O}$  的安全性保证替代不可察觉.



综合以上, 最终的构造如下:

#### 构造 4.16 (基于不可区分混淆的构造 IND-CCA KEM)

构造所需的组件是:

- 不可区分混淆  $i\mathcal{O}$
- 伪随机数发生器 PRG  $G : \{0, 1\}^\kappa \rightarrow \{0, 1\}^{2\kappa}$
- 可穿孔伪随机函数 PPRF  $F : SK \times \{0, 1\}^{2\kappa} \rightarrow Y$

构造 KEM 如下:

- **Setup**( $1^\kappa$ ): 运行  $pp \leftarrow \text{PPRF.Setup}(1^\kappa)$  生成公开参数, 私钥空间  $SK$  为可穿孔伪随机函数的密钥空间  $K$ , 密文空间  $C = \{0, 1\}^{2\kappa}$ , 会话密钥空间  $K = Y$ .
- **KeyGen**( $pp$ ): 随机采样  $sk \xleftarrow{R} SK$ , 计算  $pk \leftarrow i\mathcal{O}(\text{Encaps})$ .
- **Encaps**( $pk; r$ ): 运行  $(c, k) \leftarrow pk(r)$ .
- **Decaps**( $sk, c$ ): 输出  $k \leftarrow F(sk, c)$ .



#### Encaps

Constants: PPRF key  $sk$

Input: randomness  $x \in \{0, 1\}^\lambda$

1. output  $c \leftarrow G(x), k \leftarrow F(sk, c)$

构造 4.16 的正确性显然, 安全性由以下定理保证.

#### 定理 4.15

如果  $F$  是安全的可穿孔伪随机函数、 $G$  是安全的伪随机数发生器、 $i\mathcal{O}$  是不可区分混淆, 则构造 4.16 满足 IND-CCA 安全性.



**证明** 以下通过游戏序列完成定理证明.

**Game<sub>0</sub>**: 对应真实游戏

- 初始化:  $\mathcal{CH}$  运行  $\text{PPRF.Setup}(1^\kappa)$  生成公开参数, 随机采样  $sk \xleftarrow{R} SK$ , 生成公钥  $pk \leftarrow i\mathcal{O}(\text{Encaps})$
- 挑战阶段:  $\mathcal{CH}$  随机采样  $x^* \xleftarrow{R} \{0, 1\}^\kappa$ , 计算  $c^* \leftarrow G(x^*)$ ,  $k_0^* \leftarrow F(sk, c^*)$ , 随机采样  $k_1^* \xleftarrow{R} K$ ,  $\beta \xleftarrow{R} \{0, 1\}$ , 将  $(c^*, k_\beta^*)$  发送给  $\mathcal{A}$  作为挑战.
- 解封装询问:  $\mathcal{A}$  发起询问  $c \in C$ ,  $\mathcal{CH}$  返回  $k \leftarrow F(sk, c)$ .



- 猜测:  $\mathcal{A}$  输出对  $\beta$  的猜测  $\beta'$ , 攻击成功当且仅当  $\beta = \beta'$ .

**Game<sub>1</sub>:** 与 **Game<sub>0</sub>** 的区别是  $\mathcal{CH}$  在挑战阶段随机采样  $c^* \xleftarrow{R} \{0,1\}^{2\kappa}$  而非计算  $c^* \leftarrow G(x^*)$ . PRG 的伪随机性保证了:

$$\text{Game}_0 \approx_c \text{Game}_1$$

**Game<sub>2</sub>:** 与 **Game<sub>0</sub>** 的区别是  $\mathcal{CH}$  将  $c^*$  的生成从挑战阶段提前到初始化阶段 (为后续使用可穿孔伪随机函数做准备). 该变化完全隐藏于敌手, 因此有:

$$\text{Game}_1 \equiv \text{Game}_2$$

**Game<sub>3</sub>:**  $\mathcal{CH}$  在初始化阶段计算  $pk \leftarrow i\mathcal{O}(\text{Encap}^*)$  而非之前的  $pk \leftarrow i\mathcal{O}(\text{Encap})$ ; 在应答解封装询问时, 使用  $sk_c^*$  计算并返回  $k \leftarrow F(sk_{c^*}, c)$ , 代替之前使用  $k$  计算并返回  $k \leftarrow F(sk, c)$ .

Encaps <sup>*</sup>	
Constants:	PPRF punctured key $sk_{c^*}, c^*$
Input:	randomness $x \in \{0,1\}^\lambda$
1. output	$c \leftarrow G(x), k \leftarrow F(sk_{c^*}, c)$ .

- 由于  $\Pr[c^* \in \text{Img}(G)] = 1/2^\kappa$ , 因此  $c^*$  落在  $G$  的像集中的概率可忽略, 故而穿孔导致程序输入输出行为差异的概率可忽略, 即  $\Pr[\text{Encaps}_{sk} \equiv \text{Encaps}_{sk_{c^*}}] = 1 - 1/2^\kappa$ .  $i\mathcal{O}$  的安全性保证了公钥的分布计算不可区分

$$i\mathcal{O}(\text{Encaps}) \approx_c i\mathcal{O}(\text{Encaps}^*)$$

- 对于所有合法的解密询问  $c \neq c^*$ , 可穿孔伪随机函数的正确性保证了  $F(sk, c) = F(sk_{c^*}, c)$ .
- 因此, 我们有

$$\text{Game}_2 \approx_c \text{Game}_3$$

**Game<sub>4</sub>:**  $\mathcal{CH}$  随机采样  $k_0^* \xleftarrow{R} K$  代替上一游戏的  $k_0^* \leftarrow F(sk, c^*)$ . 可穿孔伪随机函数的弱伪随机性保证了

$$\text{Game}_3 \approx_c \text{Game}_4$$

在 **Game<sub>4</sub>**,  $k_0^*$  和  $k_1^*$  均从  $K$  中均匀随机采样, 因此即使  $\mathcal{A}$  拥有无穷的计算能力, 其在 **Game<sub>4</sub>** 中的优势也是 0.

综合以上, 定理得证! □

#### 注记 4.17

构造 4.16 中的 KEM 也具备可穿孔性质. 该构造充分展示了  $i\mathcal{O}$  的魔力——使得在不暴露秘密的情况下可以公开执行“内嵌秘密值”的程序:

- 将私钥组件编译为公钥组件



## 4.5 可公开求值伪随机函数类

明修栈道,暗渡陈仓.

— 汉·司马迁《史记·高祖本纪》

前面的章节已经展示了若干种构造公钥加密的通用方法,包括单向陷门函数、哈希证明系统、可提取哈希证明系统以及不可区分混淆结合可穿孔伪随机函数. 这些通用构造阐释了绝大多数公钥加密方案,然而令人颇感以外的是,它们并无法阐释最经典的 ElGamal PKE [ELG85] 和 Goldwasser-Micali PKE [GM84] 另一方面,伪随机函数是密码学的核心基本组件之一,应用范围极其广泛,特别的,伪随机函数蕴含了简洁优雅的、也是目前唯一的 IND-CPA SKE 通用构造.

$$\text{Enc}(sk, m; r) \rightarrow (r, F(sk, x) \oplus m)$$

然而伪随机函数属于 Minicrypt, 因此在黑盒意义下无法蕴含 PKE.

以上的现象促使我们考虑如下的问题:

### 思考 4.1

是否存在新型的伪随机函数能够让 PRF-based SKE 延拓到公钥场景? 新型的伪随机函数是否能蕴含统一上述的不同构造,并阐释经典 PKE 方案的设计机理?

我们首先分析基于 PRF 构造 PKE 的技术难点:

- 密文必须可以公开计算: 显然, PRF-based SKE 的构造正是因为这个原因无法延拓到公钥加密场景中

$$(x, F(sk, x) \oplus m)$$

因为  $F$  的伪随机性意味着其不可能公开求值.

解决上述问题的关键在于探求伪随机性 (pseudorandomness) 和可公开求值性 (public evaluability) 是否能够共存. 标准的伪随机函数处处伪随机 (universal pseudorandom), 即对于定义域中任意  $x \in X$ , PPT 敌手  $\mathcal{A}$  都无法区分  $F_k(x)$  和随机值.

- 观察 1: 构造 IND-CPA KEM 仅需要弱伪随机性 (weak pseudorandomness), 即对于挑战者随机选择的挑战输入, 其 PRF 值是伪随机的
- 观察 2: 如果掌握输入  $x$  的某些辅助信息  $aux$  (比如采样  $x$  的随机数), 是有可能在不使用  $sk$  的情形下对  $F_{sk}(x)$  公开求值. 如果  $aux$  在平均意义下是难以抽取的, 则公开求值性与弱伪随机性不冲突.

综合以上, 在 KEM 中由发送方生成  $x$ , 因此其知晓  $aux$  信息, 从而以下两点成为可能:

- 功能性方面: 发送方可以借助  $aux$  对  $F_{sk}(x)$  公开求值从而生成密文.
- 安全性方面:  $F_{sk}(x)$  在  $\mathcal{A}$  的视图中仍然伪随机.

### 4.5.1 定义与安全性

正是基于上面的思考, 陈等 [CZ14] 提出了可公开求值伪随机函数 (PEPRF, Publicly Evaluable PRFs). PEPRF 考虑了定义域  $X$  包含  $\mathcal{NP}$  语言  $L$  的情形, 使用私钥可以对全域求值, 而使用公钥和证据可以对语言  $L$  内的元素求值. 在安全性上, PEPRF 要求函数在语言  $L$  上弱伪随机.

#### 定义 4.17 (可公开求值伪随机函数)

- $\text{Setup}(1^\kappa)$ : 以安全参数  $\kappa$  为输入, 输出公开参数  $pp = (F, PK, SK, X, L, W, Y)$ , 其中  $F : SK \times X \rightarrow Y \cup \perp$  是由  $SK$  索引的一族函数,  $L \subseteq X$  是由困难关系  $R_L$  定义的  $\mathcal{NP}$  语言, 其中  $W$  是相应的证据集合.  $R_L$  是高效可采样的, 存在 PPT 算法  $\text{SampR}$  以随机数  $r$  为输入, 输出实例证据元组  $(x, w) \in R_L$ .
- $\text{KeyGen}(pp)$ : 以公开参数  $pp$  为输入, 输出公钥  $pk$  和私钥  $sk$ .

- $\text{PrivEval}(sk, x)$ : 以私钥  $sk$  和元素  $x \in X$  为输入, 输出  $y \in Y \cup \perp$ .
- $\text{PubEval}(pk, x, w)$ : 以公钥  $pk$ 、实例  $x \in L$  以及相应的证据  $w \in W$  为输入, 输出  $y \in Y$ .

**注记 4.18**

在有些场景中有必要将单一语言  $L$  泛化为由  $PK$  索引的一族语言  $\{L_{pk}\}_{pk \in PK}$ . 相应的, 采样算法  $\text{SampRel}$  将以  $pk$  为额外输入, 随机采样  $(x, w) \in R_{L_{pk}}$ .

**正确性.** 对于任意  $pp \leftarrow \text{Setup}(1^\lambda)$  和  $(pk, sk) \leftarrow \text{KeyGen}(pp)$ , 我们有:

$$\begin{aligned} \forall x \in X : \quad & F_{sk}(x) = \text{PrivEval}(sk, x) \\ \forall x \in L \text{ 以及证据 } w : \quad & F_{sk}(x) = \text{PubEval}(pk, x, w) \end{aligned}$$

**(自适应) 弱伪随机性.** 定义敌手  $\mathcal{A}$  的优势函数如下:

$$\Pr \left[ \begin{array}{l} pp \leftarrow \text{Setup}(1^\kappa); \\ (pk, sk) \leftarrow \text{KeyGen}(pp); \\ r^* \xleftarrow{R}, (x^*, w^*) \leftarrow \text{SampR}(r^*); \\ y_0^* \leftarrow F_{sk}(x^*), y_1^* \leftarrow Y; \\ \beta \leftarrow \{0, 1\}; \\ \beta' \leftarrow \mathcal{A}^{\mathcal{O}_{\text{eval}}(\cdot)}(pp, pk, x^*, y_b^*); \end{array} \right] - \frac{1}{2}.$$

其中  $\mathcal{O}_{\text{eval}}$  表示求值谕言机, 以  $x \neq x^* \in X$  为输入, 返回  $F_{sk}(x)$ . 如果任意 PPT 敌手  $\mathcal{A}$  在上述游戏中的优势函数均为可忽略函数, 则称可公开求值伪随机函数是弱伪随机的. 如果敌手在上述游戏中可以访问  $\mathcal{O}_{\text{eval}}$  谕言机, 则称可公开求值伪随机函数是自适应弱伪随机的.

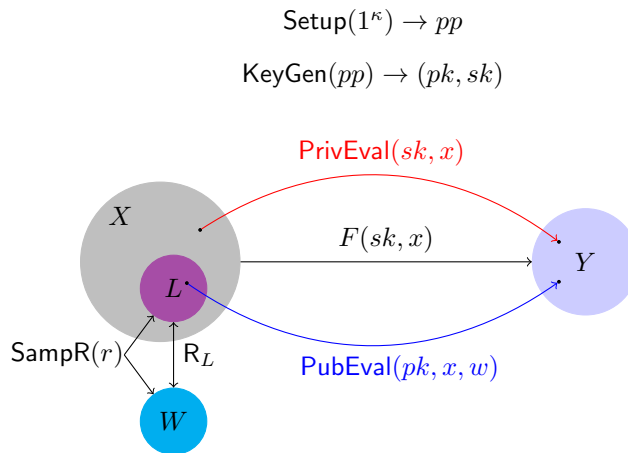


图 4.35: PEPRF 示意图

**注记 4.19**

在 PEPRF 中, 私钥用于秘密求值, 公钥则可在知晓相应证据时对语言内的元素进行公开求值. 密钥成对出现这一点对于 PEPRF 是自然的, 因为 PEPRF 是作为 PRF 在 Cryptomania 中的对应引入的. 另一方面, 标准的 PRF 也总是可以设置公钥用于发布与私钥相关联但可公开的信息, 例如在基于 DDH 假设的 Naor-Reingold PRF [NR04] 中,  $F_{\vec{a}}(x) = (g^{a_0})^{\prod_{i=1}^n a_i}$ , 其中  $\vec{a} = (a_0, a_1, \dots, a_n) \in \mathbb{Z}_p^n$  是私钥,  $\{g^{a_i}\}_{1 \leq i \leq n}$  则可发布为公钥. 如果没有信息可公开, 可设定  $pk = \{\perp\}$ . 如此可保持 PRF 与 PEPRF 的语法定义保持一致.

	PRF	PEPRF
带密钥函数	✓	✓
可公开求值	$\forall x \in X$ ✗	$x \in L$ ✓
安全性	$\forall x \in X$ pseudorandom	$x \xleftarrow{R} L$ weak pseudorandom

表 4.1: PRF 与 PEPRF 的比较

为什么 PEPRF 只定义了弱伪随机性呢？这是因为在公开求值算法 PubEval 存在的前提下，这是可达的最强安全性。

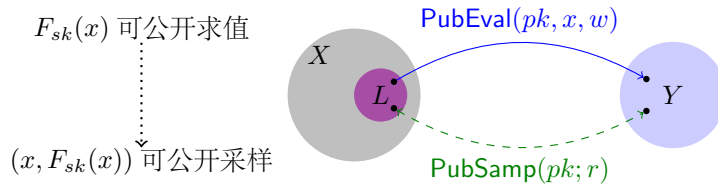
为了加深对概念的理解，表 4.1 对比分析 PRF 与 PEPRF 的异同。正是由于上述区别，我们可以基于 PEPRF 构造 KEM。

PEPRF 的定义可以进一步泛化以包容更多实例化构造。

#### 定义 4.18 (可公开采样伪随机函数 (PSPRF, Publicly Sampleable PRF))

PSPRF 将 PEPRF 的可公开求值功能可以放宽为可公开采样功能，即 PubEval 算法由以下的 PPT 随机采样算法替代：

- $\text{PubSamp}(pk; r) \rightarrow (x, y) \in L \times Y$  s.t.  $y = F_{sk}(x)$



显然，可以综合关系采样算法和函数公开求值算法构造公开采样算法，因此 PEPRF 蕴含 PSPRF：

- $\text{PubSamp}(pk; r)$ : 运行  $(x, w) \leftarrow \text{SampRel}(r)$ ，输出  $(x, \text{PEPRF.PubEval}(pk, x, w))$ 。

### 4.5.2 基于 PEPRF 的 KEM 构造

本章将展示如何基于 PEPRF 构造 KEM。

#### 构造 4.17 (基于 PEPRF 的 KEM 构造)

构造思路：随机采样语言中的元素作为密文，计算其函数值作为会话密钥  $k$ 。

起点：PEPRF  $F: SK \times X \rightarrow Y \cup \perp$ ，其中  $L \subseteq X$  是定义在  $X$  上的  $\mathcal{NP}$  语言。

构造如下：

- $\text{Setup}(1^\kappa)$ : 运行  $pp \leftarrow \text{PEPRF.Setup}(1^\kappa)$ ，其中密文空间  $C = X$ ，会话密钥空间  $K = Y$ 。
- $\text{KeyGen}(pp)$ : 运行  $(pk, sk) \leftarrow \text{PEPRF.KeyGen}(pp)$ 。
- $\text{Encaps}(pk; r)$ : 随机采样  $(x, w) \leftarrow \text{SampR}(r)$ ，输出  $c = x$  作为密文，通过  $\text{PEPRF.PubEval}(pk, x, w)$  公开计算  $k \leftarrow F_{sk}(x)$  作为会话密钥。
- $\text{Decaps}(sk, c)$ : 通过运行  $\text{PEPRF.PrivEval}(sk, c)$  秘密计算  $k \leftarrow F_{sk}(x)$  恢复会话密钥。

构造 4.17 的正确性由 PEPRF 的正确性保证，安全性由以下定理保证。

**定理 4.16**

如果 PEPRF 是弱伪随机的, 则构造 4.17 是 IND-CPA 安全的; 如果 PEPRF 是自适应弱伪随机的, 则构造 4.17 是 IND-CCA 安全的.

**证明** IND-CPA 安全性的归约是显然的, 建立 IND-CCA 安全性的关键是令归约算法利用  $\mathcal{O}_{\text{eval}}$  模拟  $\mathcal{O}_{\text{decaps}}$ .  $\square$

**注记 4.20**

在上述的 KEM 构造中, 可以将 PEPRF 弱化为 PSPRF.

**4.5.3 PEPRF 的构造**

天下同归而殊途, 一致而百虑.

— 《周易·系辞下》

本章节展示如何基于具体的困难假设和(半)通用的密码组件构造 PEPRF.

**4.5.3.1 基于 DDH 假设的 PEPRF**

图 4.36 展示了基于 DDH 假设的 PEPRF 构造, 其中可公开求值功能利用了 DH 函数的可交换性, 弱伪随机性建立在 DDH 假设之上. 将实例化代入构造 4.17 中, 得到的正是经典的 ElGamal PKE 方案 [EIG85].

**构造 4.18 (基于 DDH 假设的 PEPRF)**

- $\text{Setup}(1^\kappa)$ : 运行  $(\mathbb{G}, p, g) \leftarrow \text{GroupGen}(1^\kappa)$ , 生成公开参数  $pp = (F, PK, SK, X, L, W, Y)$ , 其中  $X = Y = PK = L = \mathbb{G}$ ,  $SK = W = \mathbb{Z}_p$ ,  $F : SK \times X \rightarrow Y$  定义为  $F_{sk}(x) = x^{sk}$ , 语言  $L = \{x : \exists w \in W \text{ s.t. } x = g^w\}$ , 相应的采样算法  $\text{SampRel}$  以随机数  $r$  为输入, 随机采样证据  $w \xleftarrow{R} \mathbb{Z}_p$ , 计算实例  $x = g^w$ .
- $\text{KeyGen}(pp)$ : 随机采样私钥  $sk \xleftarrow{R} \mathbb{Z}_p$ , 计算公钥  $pk = g^{sk}$ .
- $\text{PrivEval}(sk, x)$ : 输出  $x^{sk}$ .
- $\text{PubEval}(pk, x, w)$ : 输出  $pk^w$ .

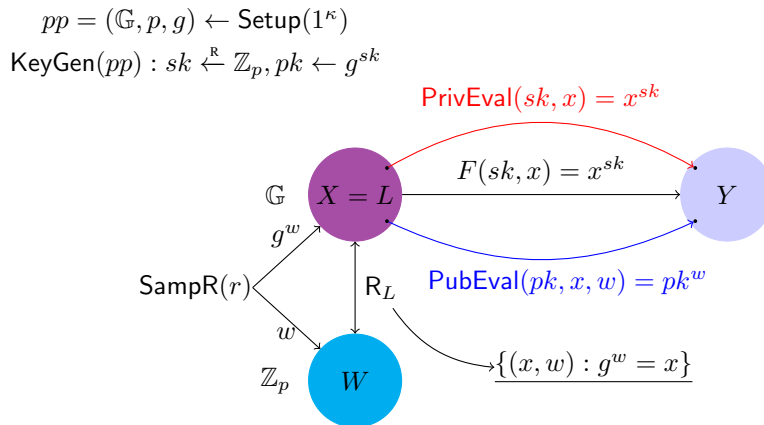


图 4.36: 基于 DDH 假设的 PEPRF

**4.5.3.2 基于 QR 假设的 PEPRF**

图 4.37 展示了基于 QR 假设的 PEPRF 构造, 其中可公开求值功能利用了语言  $L$  的 OR 型定义, 弱伪随机性建立在 QR 假设之上. 将实例化代入构造 4.17 中, 得到的正是 Goldwasser-Micali PKE 方案 [GM84] 内蕴的 KEM.

**构造 4.19 (QR-based-PEPRF)**

- $\text{Setup}(1^\kappa)$ : 输出  $pp = \kappa$ .
- $\text{KeyGen}(pp)$ : 运行  $(N, p, q) \leftarrow \text{GenModulus}(1^\kappa)$ , 选取  $z \in \mathbb{QNR}_N^{+1}$ , 输出公钥  $pk = (N, z)$  和私钥  $sk = (p, q)$ .  $pk$  还包含了以下信息: 函数定义域  $X = \mathbb{Z}_N^*$ , 值域  $Y = \{0, 1\}$ , 证据集合  $W = \mathbb{Z}_N^*$ , 语言  $L_{pk} = \{x : \exists w \in W \text{ s.t. } x = w^2 \bmod N \vee x = zw^2 \bmod N\}$  ( $\mathbb{Z}_N^*$  中 Jacobi 符号为  $+1$  的元素). 采样算法  $\text{SampRel}$  以公钥  $pk$  和随机数  $r$  为输入, 随机采样  $w \xleftarrow{R} \mathbb{Z}_p$ , 随机生成实例  $x = w^2 \bmod N$  或  $x = zw^2 \bmod N$ .
- $\text{PrivEval}(sk, x)$ : 如果  $x \in \mathbb{QR}_N$  则输出 1, 如果  $x \in \mathbb{QNR}_N^{+1}$  则输出 0.
- $\text{PubEval}(pk, x, w)$ : 如果  $x = w^2 \bmod N$  则输出 1, 如果  $x = zw^2 \bmod N$  则输出 0.

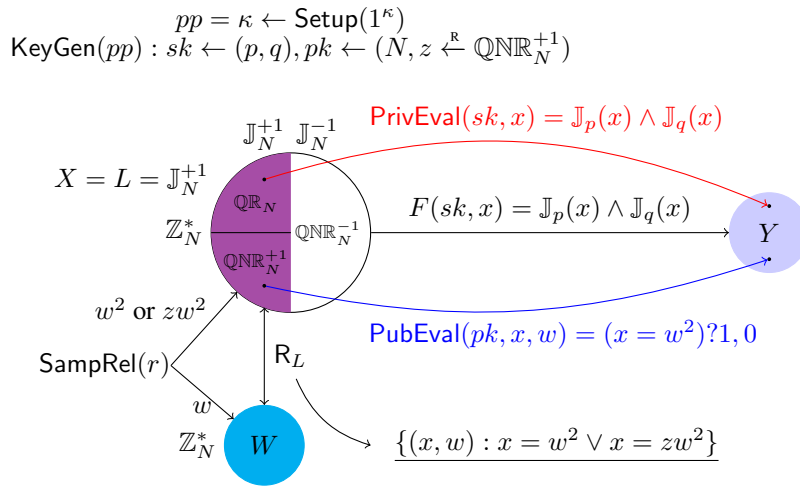


图 4.37: 基于 QR 假设的 PEPRF 构造

**4.5.3.3 基于 TDF 的 PEPRF**

通过扭转单射 TDF, 可以构造 PEPRF 如下.

**构造 4.20 (基于 TDF 的 PEPRF 构造)**

- $\text{Setup}(1^\kappa)$ : 运行  $pp = (G, EK, TD, S, U) \leftarrow \text{TDF.Setup}(\lambda)$ , 令  $\text{hc} : S \rightarrow K$  是相应的 hardcore function; 生成 PEPRF 的公开参数  $pp = (F, PK, SK, X, L, W, Y)$ , 其中  $PK = EK, SK = TD, Y = K, X = U, W = S, F_{sk}(x) = \text{hc}(G_{td}^{-1}(x))$ . 算法  $\text{TDF.Eval}$  自然定义了一族定义在  $X$  上的  $\mathcal{NP}$  语言  $L = \{L_{pk}\}_{pk \in PK}$ , 其中  $L_{pk} = \{x : \exists w \in W \text{ s.t. } x = \text{TDF.Eval}(pk, w)\}$ . 采样算法  $\text{SampRel}$  以随机数  $r$  为输入, 首先随机采样定义域中元素  $s \leftarrow \text{SampDom}(r)$ , 再计算  $u \leftarrow \text{TDF.Eval}(pk, s)$ , 输出实例  $x = u$  和证据  $w = s$ .
- $\text{KeyGen}(pp)$ : 运行  $(ek, td) \leftarrow \text{TDF.KeyGen}(pp)$ , 输出  $pk = ek$  和  $sk = td$ .
- $\text{PrivEval}(sk, x)$ : 以私钥  $sk$  和元素  $x \in X$  为输入, 输出  $y \leftarrow F_{sk}(x) = \text{hc}(\text{TDF.TdInv}(sk, x))$ .
- $\text{PubEval}(pk, x, w)$ : 以公钥  $pk$ 、实例  $x \in L_{pk}$  和证据  $w$  为输入, 输出  $y \leftarrow \text{hc}(w)$ .



构造 4.20 的正确性由陷门单向函数的正确性和单射性保证, 安全性由如下定理保证.

**定理 4.17**

如果起点 TDF 是 (自适应) 单向的, 那么构造 4.20 中的 PEPRF 是 (自适应) 弱伪随机的.



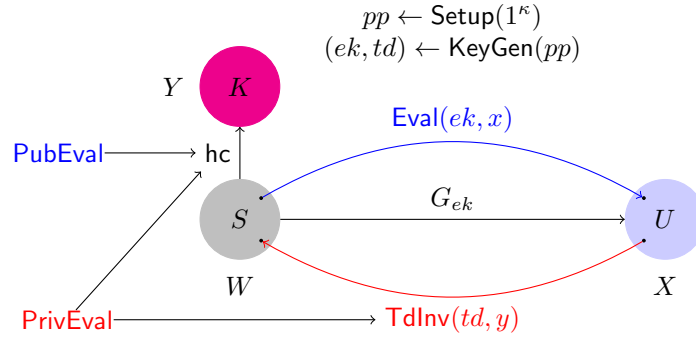


图 4.38: 基于 TDF 的 PEPRF 构造

	HPS	PEPRF
投射性	✓	not necessary
$L$ 与 $X$ 的关系	$L \subset X$	$L \subseteq X$
弱伪随机性	$x \xleftarrow{R} X \setminus L$	$x \xleftarrow{R} L$

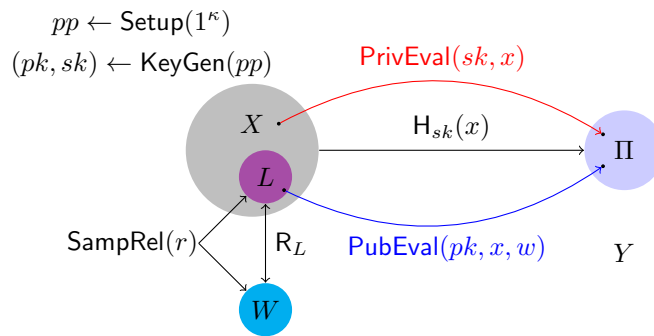
表 4.2: HPS 与 PEPRF 的不同

#### 4.5.3.4 基于 HPS 的 PEPRF 构造

本章节展示如何基于哈希证明系统构造具有不同安全性质的可公开求值伪随机函数。首先展示如何基于平滑 HPS 构造弱伪随机的 PEPRF。

##### 构造 4.21 (基于平滑 HPS 的 PEPRF 构造)

- $\text{Setup}(1^\kappa)$ : 运行  $\text{HPS.Setup}(1^\lambda)$  生成 HPS 的公开参数  $pp = (H, PK, SK, X, L, W, \Pi, \alpha)$ , 输入 PEPRF 的公开参数  $pp = (F, PK, SK, X, L, W, Y)$ , 其中  $F = H, Y = \Pi$ .
- $\text{KeyGen}(pp)$ : 运行  $(pk, sk) \leftarrow \text{HPS.KeyGen}(pp)$  生成密钥对.
- $\text{PrivEval}(sk, x)$ : 以私钥  $sk$  和元素  $x \in X$  为输入, 计算  $y \leftarrow \text{HPS.PrivEval}(sk, x)$ .
- $\text{PubEval}(pk, x, w)$ : 以公钥  $pk$ 、语言中的元素  $x \in L$  和相应的证据  $w \in W$  为输入, 计算  $y \leftarrow \text{HPS.PubEval}(pk, x, w)$ .



构造 4.21 的正确性由平滑 HPS 的正确性保证, 安全性由如下定理保证:

##### 定理 4.18

基于  $L \subset X$  上的 SMP 假设, 构造 4.21 中的 PEPRF 满足弱伪随机性。

PEPRF 与 HPS 在语法上非常相似, 但存在以下微妙的不同, 如表 4.1 所示:

下面展示如何基于平滑和一致 HPS 构造自适应伪随机的 PEPRF。



**构造 4.22 (基于平滑和一致 HPS 的 PEPF 构造)**

构造组件: 针对同一语言  $\tilde{L} \subset \tilde{X}$  的 smooth HPS<sub>1</sub> 和 2-universal HPS<sub>2</sub>:

构造如下:

- Setup( $1^\kappa$ ): 运行  $pp_1 = (H_1, PK_1, SK_1, \tilde{X}, \tilde{L}, W, \Pi_1, \alpha_1) \leftarrow \text{HPS}_1.\text{Setup}(1^\kappa)$  生成 smooth HPS 的公开参数, 运行  $pp_2 = (H_2, PK_2, SK_2, \tilde{X}, \tilde{L}, \tilde{W}, \Pi_2, \alpha_2) \leftarrow \text{HPS}_2.\text{Setup}(1^\kappa)$  生成 2-universal HPS 的公开参数, 基于  $pp_1$  和  $pp_2$  生成 PEPF 的公开参数  $pp = (F, PK, SK, X, L, W, Y)$ , 其中  $X = \tilde{X} \times \Pi_2$ ,  $Y = \Pi_1 \cup \perp$ ,  $PK = PK_1 \times PK_2$ ,  $L = \{L_{pk}\}_{pk \in PK}$  定义在  $X = \tilde{X} \times \Pi_2$  上, 其中  $L_{pk} = \{x = (\tilde{x}, \pi_2) : \exists w \in W \text{ s.t. } \tilde{x} \in \tilde{L} \wedge \pi_2 = \text{HPS}_2.\text{PubEval}(pk_2, \tilde{x}, w)\}$ , 相应的采样算法 SampRel 以公钥  $pk = (pk_1, pk_2)$  和随机数  $r$  为输入, 首先随机采样语言  $\tilde{L}$  的随机实例证据元组  $(\tilde{x}, w)$ , 计算  $\pi_2 \leftarrow \text{HPS}_2.\text{PubEval}(pk_2, \tilde{x}, \tilde{w})$ , 输出语言  $L$  的实例  $x = (\tilde{x}, \pi_2)$  和证据  $w = \tilde{w}$ . 不失一般性, 令  $pp$  包含  $pp_1$  和  $pp_2$  中的所有信息.
- KeyGen( $pp$ ): 从  $pp$  中解析出  $pp_1$  和  $pp_2$ , 运行  $(pk_1, sk_1) \leftarrow \text{HPS}_1.\text{KeyGen}(pp_1)$  和  $(pk_2, sk_2) \leftarrow \text{HPS}_2.\text{KeyGen}(pp_2)$ , 输出  $pk = (pk_1, pk_2)$  和  $sk = (sk_1, sk_2)$ .
- PrivEval( $sk, x$ ): 以私钥  $sk = (sk_1, sk_2)$  和  $x = (\tilde{x}, \pi_2)$  为输入, 如果  $\pi_2 = \text{HPS}_2.\text{PrivEval}(sk_2, \tilde{x})$  则返回  $\perp$  否则返回  $y \leftarrow \text{HPS}_1.\text{PrivEval}(sk_1, \tilde{x})$ . 该算法定义了  $F : SK \times X \rightarrow Y \cup \perp$ .
- PubEval( $pk, x, w$ ): 以公钥  $pk = (pk_1, pk_2)$ 、元素  $x = (\tilde{x}, \pi_2) \in L_{pk}$  以及证据  $w$  为输入, 输出  $y \leftarrow \text{HPS}_1.\text{PubEval}(pk_1, \tilde{x}, w)$ .

**定理 4.19**

基于  $\tilde{L} \subset \tilde{X}$  上的 SMP 假设, 构造 4.22 中的 PEPF 是自适应弱伪随机的.

**注记 4.21**

构造 4.21 相对直接, 构造 4.22 稍显复杂, 其中蕴含的设计思想与基于哈希证明系统构造 CCA-secure PKE 相似: 使用“弱”HPS 封装随机会话密钥, 使用“强”HPS 生成证明以杜绝“危险”解密询问.

**4.5.3.5 基于 EHPS 的 PEPF 构造**

本章节展示如何基于 (ABO-)EHPS 构造 PEPF.

**构造 4.23 (基于 (ABO-)EHPS 的 PEPF 构造)**

- Setup( $1^\kappa$ ): 运行  $pp = (H, PK, SK, \tilde{L}, \tilde{W}, \Pi) \leftarrow \text{EHPS}.\text{Setup}(1^\kappa)$  生成 EHPS 的公开参数, 令  $\text{hc} : \tilde{W} \rightarrow Z$  是单向关系  $R_{\tilde{L}}$  的 hardcore function; 生成 PEPF 的公开参数  $pp = (F, PK, SK, X, L, W, Y)$ , 其中  $X = \tilde{L} \times \Pi$ ,  $Y = Z$ ,  $W = R$ ,  $L = \{L_{pk}\}_{pk \in PK}$  定义在  $X = \tilde{L} \times \Pi$  上, 其中  $L_{pk} = \{x = (\tilde{x}, \pi) : \exists w \in W \text{ s.t. } \tilde{x} = \text{SampIns}(w) \wedge \pi = \text{EHPS}.\text{PubEval}(pk, \tilde{x}, w)\}$ , 相应的采样算法以公钥  $pk$  和随机数  $w$  为输入, 首先以  $w$  作为证据生成实例  $\tilde{x} \xleftarrow{R} \tilde{L}$ , 再计算  $\pi \leftarrow \text{EHPS}.\text{PubEval}(pk, \tilde{x}, w)$ , 输出实例  $x = (\tilde{x}, \pi)$  和证据  $w$ .  $F_{sk}(x) := \text{hc}(\text{EHPS}.\text{Ext}(sk, x))$ .
- KeyGen( $pp$ ): 运行  $(pk, sk) \leftarrow \text{EHPS}.\text{KeyGen}(pp)$  生成密钥对.
- PrivEval( $sk, x$ ): 以私钥  $sk$  和  $x \in X$  为输入, 将  $x$  解析为  $(\tilde{x}, \pi)$ , 计算  $\tilde{w} \leftarrow \text{EHPS}.\text{Ext}(sk, \tilde{x}, \pi)$ , 输出  $y \leftarrow \text{hc}(\tilde{w})$ .
- PubEval( $pk, x, w$ ): 以公钥  $pk$ 、 $x \in L_{pk}$  以及相应的证据  $w$  为输入, 计算  $\tilde{w} \leftarrow \text{SampWit}(w)$ , 输出  $y \leftarrow \text{hc}(\tilde{w})$ .



**定理 4.20**

如果  $R_L$  是单向的, 基于 (ABO-)EHPS 的 PEPRF 是 (自适应) 弱伪随机的.

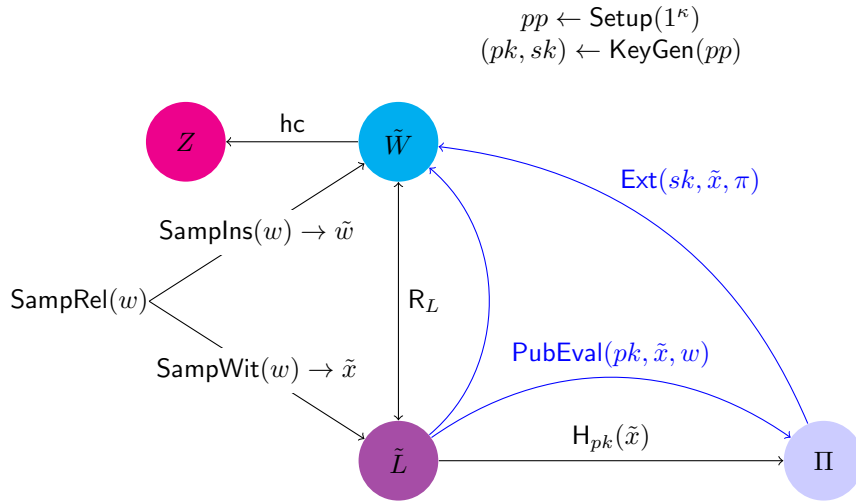


图 4.39: 基于 EHPS 的 PEPRF 构造

### 4.5.3.6 基于 $i\mathcal{O}$ 的 PEPRF 构造

本章节展示如何基于不可区分程序混淆构造 PEPRF.

**构造 4.24 (基于  $i\mathcal{O}$  和 PPRF 的 PEPRF 构造)**

构造组件: 不可区分程序混淆  $i\mathcal{O}$ 、伪随机数发生器和可穿孔伪随机函数

构造如下:

- $\text{Setup}(1^\kappa)$ : 生成对电路族  $\mathcal{C}_\kappa$  的不可区分程序混淆  $i\mathcal{O}$ , 选取长度倍增的伪随机数发生器 PRG :  $\{0,1\}^\kappa \rightarrow \{0,1\}^{2\kappa}$  和可穿孔伪随机函数 PPRF :  $K \times \{0,1\}^{2\kappa} \rightarrow Y$ ; 生成 PEPRF 的公开参数  $pp = (F, PK, SK, X, L, W, Y)$ , 其中  $PK = i\mathcal{O}(\mathcal{C}_\kappa)$ ,  $SK = K$ ,  $X = \{0,1\}^{2\kappa}$ , 其中  $F := \text{PPRF}$ ,  $W = \{0,1\}^\kappa$ ,  $L = \{x \in X : \exists w \in W \text{ s.t. } x = \text{PRG}(w)\}$ , 相应的采样算法  $\text{SampRel}$  以随机数  $r \in \{0,1\}^\kappa$  为输入, 输出实例  $x \leftarrow \text{PRG}(r)$  和证据  $w = r$ .
- $\text{KeyGen}(pp)$ : 随机采样  $k \in K$  作为私钥  $sk$ , 计算  $pk \leftarrow i\mathcal{O}(\text{Eval})$  作为公钥.
- $\text{PrivEval}(sk, x)$ : 输出  $y \leftarrow \text{PPRF}(sk, x)$ .
- $\text{PubEval}(pk, x, w)$ : 将公钥  $pk$  解析为程序, 计算  $y \leftarrow pk(x, w)$ .

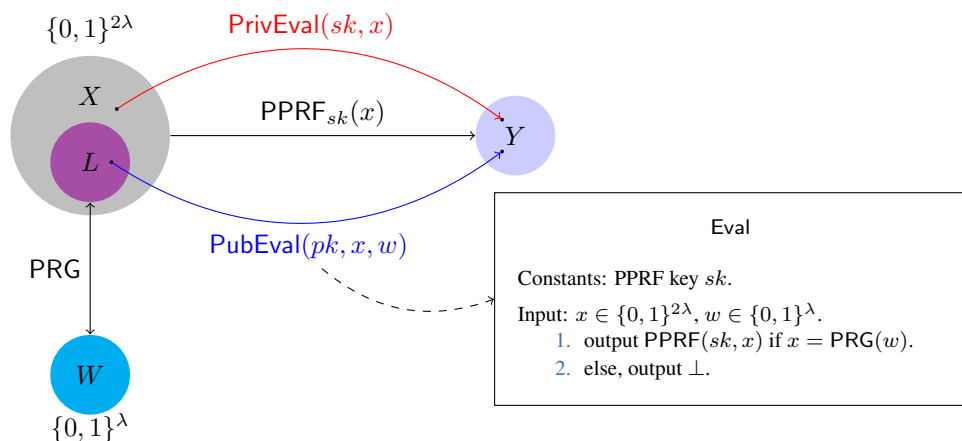
**定理 4.21**

基于不可区分程序混淆、伪随机数发生器和可穿孔伪随机函数的安全性, 构造 4.24 中的 PEPRF 满足自适应弱伪随机性.

**注记 4.22**

上述构造实质上展示了  $i\mathcal{O}$  可以将 Minicrypt 中的可穿孔伪随机函数编译为 Cryptomania 中的可公开求值伪随机函数.





#### 4.5.4 小结

本章中引入了 **PEPRF** 这一全新的密码组件, 展示了它与已有密码组件之间的联系以及它的应用, 如图 4.40 所示. 引入 **PEPRF** 最大的理论意义在于它不仅首次阐明了经典的 **Goldwasser-Micali PKE** 和 **ElGamal PKE** 的构造机理, 还统一了几乎所有已知的构造范式. 作为首个实用的公钥加密, **RSA PKE** 影响深远, 令单向陷门函数的概念深入人心, 使得人们常有“构造公钥加密必须有陷门”的错觉. **PEPRF** 树立了正确的认知, 指出构造公钥加密的实质在于构造可公开求值的伪随机函数, 核心技术是“令同一函数存在两种求值方法”. 基于 **PEPRF** 的 **PKE** 构造恰与 **Minicrypt** 中基于 **PRF** 的 **SKE** 构造形成完美的形式契合与思想共鸣.

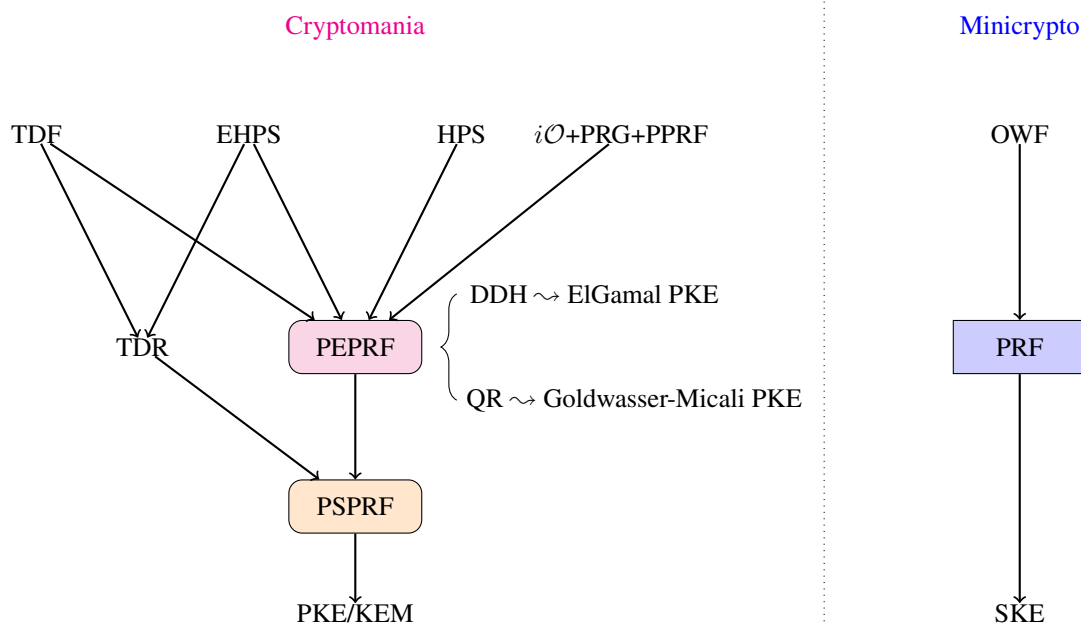


图 4.40: PPRF 的构造与应用

**PEPRF** 的强大威力来源于其高度抽象, 它诠释了公钥加密设计的“万法同源, 殊途同归”.

**笔记** 抽象的概念是美妙的, 相信读者能够通过 **PEPRF** 感受到“大繁至简”的优雅与“高屋建瓴”的力量. 然而抽象概念是果, 具体构造是因. 切不可刻意过度的抽象而忽视具体构造, 正是多种多样具体构造才让我们能够有机缘洞见事物本质, 使得高度凝练的概念内涵丰富、意义深刻.

# 第五章 公钥加密的安全性增强

内容提要

- ☐ 抗泄漏安全

☐ 抗篡改安全
- ☐ 消息依赖密钥安全

## 5.1 抗泄漏安全

写作思路: 先参考 Kalai 和 Reyzin 的综述 [KR19] 先给出一段关于抗泄漏 PKE 的综述, 然后聚焦 bounded leakage model, 先介绍 [NS09], 再介绍你的 ASIACRYPT、PKC 和我的 ASIACRYPT 合并起来. 可选的: 介绍一下 randomness leakage resilience, 方案就是我们以前讨论过的 HPS 对偶方案.

## 5.2 抗篡改安全

写作思路: 先参考 Damgård 等的 [Dam+13] 简介 PKE 的抗篡改安全, 然后可以介绍一下 Wee 的 [Wee12b], 然后再介绍你的 PKC 或者直接介绍我们的 NMF

## 5.3 消息依赖密钥安全

可以先介绍 Wee 的 [Wee16], 再介绍韩帅等的 [HLL16].



# 第六章 公钥加密的功能性扩展

## 内容提要

- 确定性公钥加密
- 可搜索公钥加密
- 代理重加密【待补充】
- 门限公钥加密【待补充】

## 6.1 确定性公钥加密

可以介绍 Wee 的 [Wee12a].

## 6.2 可搜索公钥加密

这部分你是专家, 可以在介绍完你的工作后介绍一下我关于 PKE-PEKS 的集成方案 [Che+16].

## 第七章 标准化及工程实践

### 内容提要

□ 公钥加密的标准化工作简介

□ 公钥加密的工程实践经验

## 7.1 标准化与工程实践

纸上得来终觉浅, 绝知此事要躬行.

— 宋·陆游《冬夜读书示子聿》

### 7.1.1 公钥加密的标准化

标准化对密码技术的实际落地应用具有重要意义, 否则, 即使是同一密码方案/协议也可能由于参数选取、接口设计缺乏统一的规范而无法互联互通. 以下首先简要介绍与密码领域相关较为密切的国内外标准化组织.

#### 7.1.1.1 国内外标准化组织简介

##### 1. 国际标准化组织与国际电工委员会

国际标准化组织 (ISO) 与国际电工委员会 (IEC) 联合成立了名为 ISO/IEC JTC 1 的委员会, 重点关注信息技术领域的标准化, 联合制定了一系列 ISO/IEC 标准. 其中的 ISO/IEC 18033 系列标准规定了加密算法、密码协议和密钥管理技术. ISO/IEC 标准通常由世界各地的成员国共同参与制定, 涉及多轮草案和投票, 标准化过程严格, 需经过彻底的审查和意见反馈与修订. 正因如此, ISO/IEC 标准具有广泛的国际认可度, 在实施全球化技术和安全政策方面均具有强大的影响力, 可确保来自不同供应商的产品和服务可以安全有效地协同工作. 符合 ISO/IEC 标准的密码产品通常质量和安全方面具有较高的置信度, 这对于金融交易、医疗保健和国家安全等关键应用至关重要.

##### 2. 互联网工程任务组

互联网工程任务组 (IETF) 是一个开放的标准组织, 负责开发和推广互联网标准, 特别是维护 TCP/IP 协议族的标准. 与 ISO/IEC 不同, 它不依赖于任何特定国家或管理机构, 没有正式的会员资格或会员资格要求. IETF 将其技术文档发布为征求意见稿 RFC (Requests for Comments), IETF 制定的 RFC 全方位涵盖了计算机网络体系, 在安全性与隐私方面, IETF 制定的技术标准和实践文档致力于抵御已知和新出现的威胁, 为互联网的安全和隐私提供了重要的基础要素. IETF 针对安全方面正在进行的一些工作包括: 最新版本的传输层安全协议 TLS 1.3、自动证书管理环境协议 (最近发布为 RFC 8555) 和消息传递分层安全协议等. IETF 标准具有高度的包容性, 任何人都可参与到标准制定的过程中, 且标准制定更多的基于实施和部署规范方面的实际经验, 强调实用性和执行性. IETF 标准在万物互联互通中起到了至关重要的作用, 标准化的通信协议确保不同的系统可以无缝地协同工作, SSL/TLS 等就是 IETF 标准的典范工作.

##### 3. 美国电气电子工程师学会

美国电气电子工程师学会 (Institute of Electrical and Electronics Engineers, IEEE) 的标准化组织为推出了公钥密码学标准 IEEE P1363. 该标准包括传统公钥密码学 (IEEE Std 1363-2000 and 1363a-2004)、格基公钥密码学 (IEEE Std 1363.1-2008)、口令基公钥密码学 (IEEE Std 1363.2-2008)、使用双线性映射的公钥密码学 (IEEE Std 1363.3-2013).

##### 4. 中国国家标准局

中国国家标准局现称为中国国家标准化管理委员会 (SAC), 是中国国务院直属的政府机构, 它负责起草和管理国家标准, 并代表中国加入 ISO/IEC 等国际标准化组织. 中国国家标准局一直积极制定信息安全国家标准, 包括密码算法和协议. SAC 的标准对国内行业具有重大影响, 并且经常被用作中国境内法规的基础. 随着中国在全球贸易中的地位不断提升, SAC 标准在国际上的影响力也越来越高.

##### 5. 美国国家标准局

美国国家标准与技术研究院 (National Institute of Standards, NIST) 成立于 1901 年, 现隶属于美国商务部. NIST 是美国最古老的物理科学实验室之一, 成立之初的目的是消除当时美国工业在测量基础设施方面的短板. 当前, 从智能电网和电子健康记录到原子钟、先进纳米材料和计算机芯片, 无数产品和服务都在某种程度上依赖于 NIST 提供的技术、测量和标准. NIST 致力于制定与信息技术各个方面相关的标准和指南, 还专门为联邦机构和广大公众制定密码标准和指南, 包括哈希算法、随机数生成算法、加密方案、签名方案

和后量子密码方案等。尽管 NIST 是美国机构, 但具有国际影响力, 其标准和指南不仅被美国联邦机构广泛采用, 还被私营部门组织和全球其他政府广泛采用。

#### 6. 美国国家标准学会

美国国家标准学会 (American National Standards Institute, ANSI) 成立于 1918 年, 是美国非盈利民间标准化团体。作为自愿性标准体系中的协调中心, ANSI 的主要职能是协调国内各机构团体的标准化活动、审核批准美国国家标准、代表美国参加国际标准化活动、提供标准信息咨询服务等。在密码学领域, 该组织制定了基于椭圆曲线的公钥密码学标准 ANSI X9.63。

#### 7. RSA 公司

1990 年起, RSA 公司发布了一系列公钥密码技术标准 PKCS(Public Key Cryptography Standards), 旨在推广公司拥有专利的密码算法, 如 RSA 加密算法与签名、Schnorr 签名等。尽管 PKCS 系列不是工业标准, 但其中的部分算法已经在纳入若干标准化组织 (如 IETF 和 PKIX 工作组) 的正式标准进程中。

### 7.1.1.2 公钥加密标准方案

选择密文安全常简称为 CCA 安全, 自上世纪 90 年代起即成为公钥加密的事实标准 (de factal standard), 正因如此, 绝大多数标准化组织制定的公钥加密标准均具备选择密文安全。以下首先介绍基于数论类假设的公钥加密标准方案。

- 基于整数分解类困难问题的公钥加密方案

PKCS#1 [Pkc] 是 PKCS 系列标准中最早也应用最广泛的一个, 制定了 RSA 加密和签名标准, 最新的版本号为 v2.2。PKCS#1 中定义了 RSA 公钥和私钥应如何表示和存储, 规定了基本的 RSA 操作, 包括加密和解密, 签名和验证。特别的, 标准中为 RSA 加密方案引入填充机制 OAEP(Optimal Asymmetric Encryption Padding), 得到可证明 IND-CCA 安全的 RSA-OAEP, 解决了早期版本中存在的安全问题, 如针对 PKCS#1 v1.5 填充的自适应选择明文攻击和 Bleichenbacher 攻击。

- 基于离散对数类困难问题的公钥加密方案

离散对数类困难问题根据代数结构的不同, 划分为数域和椭圆曲线两个子类, 在同样的安全级别下, 后者的参数规模更为紧致, 因此构建于其上的密码方案相比前者具有显著的性能优势, 但是由于数学结构复杂, 工程实践的难度也更大。DHIES (Diffie-Hellman Integrated Encryption Scheme, DHIES) 是 DHIES [ABR99] 的标准化方案, 采用混合加密方式, 密钥封装机制基于数域循环群上的 ElGamal PKE 和哈希函数构造, 数据封装机制基于消息验证码和堆成加密方案构造。DHIES 整体方案在随机谰言机模型中基于 CDH 假设具备可证明的选择密文安全。ECIES (Elliptic Curve Encryption Scheme, ECIES) 是 DHIES 在椭圆曲线循环群上的对于版本。DHIES 和 ECIES 被纳入 IEEE 1363a、ANSI X9.63 和 ISO/IEC 18033-2 标准。ECIES 还被椭圆曲线密码标准组 (Standards for Efficient Cryptography Group, SECG) 纳入到椭圆曲线密码学标准 SEC 1 [Seca] 中。NIST 在联邦信息处理标准 (Federal Information Processing Standards Publication) FIPS 186-5 [Fipa]、SECG 在 SEC 2 [Secb] 和 ECC Brainpool 在 RFC 5639 [Rfc] 中分别给出了推荐的椭圆曲线参数选择。中国密码管理局为满足国内电子认证服务系统等应用需求, 于 2010 年 12 月 17 日发布了《SM2 椭圆曲线公钥密码算法》[Sm2], 2016 年成为中国国家密码标准。SM2 标准中包括推荐椭圆曲线参数和包括公钥加密方案在内的各种类型公钥密码方案。

- 基于格类困难问题的公钥加密方案

Shor 算法的出现意味着在后量子时代基于数论类困难问题的密码方案将不再安全, 因此设计能够抵抗量子攻击的密码方案成为当前密码学的前沿热点, 其中格基方案是抗量子安全密码学中的主流。NIST 自 2016 年开始了后量子密码学标准方案的征集。经过最新一轮的评审, NIST 于 2023 年 8 月 24 号发布了 3 个 FIPS 草案拟定了抗量子密码系列方案, 其中 FIPS 203 [Fipb] 定义了基于 LWE 困难问题的公钥加密方案 CRYSTALS-KYBER [Bos+18]。

如前所述, 绝大多数标准中的公钥加密方案都满足 IND-CCA 安全。然而, IND-CCA 安全与同态性无法共存, 在分布式计算环境和大数据应用等密态数据的可操作性比机密性保护更重要的场景中, 迫切需要标准化的同态

公钥加密方案.

- 部分同态加密方案标准

ISO/IEC 18033-6 [Iso] 标准中定义了 Exponential ElGamal 和 Paillier [Pai99] 两个加法同态加密方案.

- 全同态加密方案标准

全同态加密尚处于飞速发展阶段, 然而工业界的应用需求更为迫切. 2017 年, 来自 IBM、Microsoft、Intel 和 NIST 和其它开放组织的研究人员共同成立了全同态标准化联盟 (Homomorphic Encryption Standardization Consortium), 并发布了同态加密标准文档 [Alb+18]. 该文档涵盖了适用于整数运算的 Brakerski-Gentry-Vaikuntanathan (BGV) [BGV14] 和 Brakerski/Fan-Vercauteren (BFV) [Bra12; FV12]、适用于浮点数运算的 Cheon-Kim-Kim-Song (CKKS) [Che+17] 以及适用于 Boolean 电路求值的 Ducas-Micciancio (FHEW) [DM15] 和 Chillotti-Gama-Georgieva-Izabachene (TFHE) [Chi+20]. 该文档尽管不是官方标准, 但基本可以看成事实上的标准.

## 7.1.2 公钥加密的工程实践

实现密码算法对程序员的素质要求较高, 既需要专业的密码知识以确保实现的忠实性和安全性, 也需要精湛的编程技术以确保实现的效率. 在一般情况下, 不建议非专业程序员自行从底层起构建密码算法, 如此不仅可省去重复制造轮子的无用功, 更能避免造出方形轮子的错误.

### 7.1.2.1 重要方案的优秀开源实现

工程实践中经常需要使用已有的公钥加密方案, 以下推荐部分常用方案的优秀开源实现供一线程序员按图索骥.

标准公钥加密方案

- RSA-OAEP: OpenSSL 库 [Opea] 中提供了 C 语言版本的实现.
- Paillier: mpc4j 库 [Mpc] 提供了 Java 语言的实现.
- ElGamal: Kunlun 库 [Lib] 中给出了 ElGamal PKE 及其多个衍生方案的 C++ 实现, 同时给出了配套的零知识证明实现, 可直接部署应用于密态计算场景.

属性加密

- FAME(Fast Attribute-based Message Encryption) [AC17]: 首个基于标准假设完全安全的密文策略和密钥策略 ABE 方案 (对策略类型或属性没有任何限制), 构建于 Type-III 双线性映射上. 相应的开源实现可参考: <https://github.com/sagrawal87/ABE>

全同态加密: Microsoft 的 SEAL (Simple Encrypted Arithmetic Library) 库 [Sea] 给出了 BGV、BFV 和 CKKS 方案的优秀实现, PALISADE 的后继者 OpenFHE [Opeb] 则包含了所有主流全同态加密方案的实现.

### 7.1.2.2 重要的开源密码库

下面的内容适用于程序员在实现自研公钥加密方案时, 为如何选择合适的密码算法库做出参考.



表 7.1: 常用开源密码算法库

库名	编程语言	支持算子类型				易用性	实时性	国密算法支持
		对称密码	大整数运算	椭圆曲线	双线性映射			
OpenSSL	C	✓	✓	✓	✗	★★★★★	★★★★★	SM2/SM3/SM4
tongsuo	C	✓	✓	✓	✗	★★★★★	★★★★★	SM2/SM3/SM4
gmSSL	C	✓	✓	✓	✗	★★★	★★★	SM2/SM3/SM4/SM9/ZUC
mcl	C/C++	✓	✓	✓	✓	★★★	★★★★★	—
MIRACL	C/C++	✓	✓	✓	✓	★★★★★	★★★	—
NTL	C++	✗	✓	✗	✗	★★★★★	★★★★★	—
Bouncy Castle	Java/C#	✓	✓	✓	✓	★★★★★	★★★★★	SM2/SM3/SM4
Crypto++	C++	✓	✓	✓	✓	★	★★★★	SM3/SM4
Botan	C++	✓	✓	✓	✗	★★★★★	★★★★★	SM2/SM3/SM4
libsodium	C	✗	✓	✓	✗	★★★	★★	—
libgcrypt	C	✗	✓	✓	✗	★★★★	★★★★	SM2/SM3/SM4

7.1.3 密码学的工程实践经验

请伟嘉在此处补充: 其实工程实践经验应该和密码学中的对象关联不强, 适用于公钥加密的经验应该同样适用于其他种类方案.β

## 参考文献

- [ABR99] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. “DHAES: An Encryption Scheme Based on the Diffie-Hellman Problem”. In: (1999). <https://eprint.iacr.org/1999/007>.
- [AC17] Shashank Agrawal and Melissa Chase. “FAME: Fast Attribute-based Message Encryption”. In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017*. ACM, 2017, pp. 665–682.
- [Ajt96] Miklós Ajtai. “Generating Hard Instances of Lattice Problems (Extended Abstract)”. In: *STOC 1996*. ACM, 1996, pp. 99–108.
- [Alb+18] Martin Albrecht et al. *Homomorphic Encryption Security Standard*. Tech. rep. <https://homomorphicencryption.org/wp-content/uploads/2018/11/HomomorphicEncryptionStandardv1.1.pdf>. HomomorphicEncryption.org, 2018.
- [Bar+01] Boaz Barak et al. “On the (Im)possibility of Obfuscating Programs”. In: *Advances in Cryptology - CRYPTO 2001*. Vol. 2139. LNCS. Springer, 2001, pp. 1–18.
- [BCP14] Elette Boyle, Kai-Min Chung, and Rafael Pass. “On Extractability Obfuscation”. In: *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014*. Vol. 8349. LNCS. Springer, 2014, pp. 52–73.
- [BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. “Functional Signatures and Pseudorandom Functions”. In: *17th International Conference on Practice and Theory in Public-Key Cryptography, PKC 2014*. Vol. 8383. LNCS. Springer, 2014, pp. 501–519.
- [BGV14] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. “(Leveled) Fully Homomorphic Encryption without Bootstrapping”. In: *ACM Trans. Comput. Theory* 6.3 (2014), 13:1–13:36.
- [Ble98] Daniel Bleichenbacher. “Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS #1”. In: *Advances in Cryptology - CRYPTO 1998*. Vol. 1462. Lecture Notes in Computer Science. Springer, 1998, pp. 1–12.
- [Bos+18] Joppe W. Bos et al. “CRYSTALS - Kyber: A CCA-Secure Module-Lattice-Based KEM”. In: *2018 IEEE European Symposium on Security and Privacy, EuroS&P 2018*. IEEE, 2018, pp. 353–367.
- [Bra12] Zvika Brakerski. “Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP”. In: *Advances in Cryptology - CRYPTO 2012*. Vol. 7417. Lecture Notes in Computer Science. Springer, 2012, pp. 868–886.
- [BSW16] Mihir Bellare, Igors Stepanovs, and Brent Waters. “New Negative Results on Differing-Inputs Obfuscation”. In: *Advances in Cryptology - EUROCRYPT 2016*. Vol. 9666. LNCS. Springer, 2016, pp. 792–821.
- [Bün+18] Benedikt Bünz et al. “Bulletproofs: Short Proofs for Confidential Transactions and More”. In: *2018 IEEE Symposium on Security and Privacy, SP 2018*. 2018, pp. 315–334.
- [Bün+20] Benedikt Bünz et al. “Zether: Towards Privacy in a Smart Contract World”. In: *Financial Cryptography and Data Security - FC 2020*. Vol. 12059. Springer, 2020, pp. 423–443.
- [BW13] Dan Boneh and Brent Waters. “Constrained Pseudorandom Functions and Their Applications”. In: *Advances in Cryptology - ASIACRYPT 2013*. Vol. 8270. LNCS. Springer, 2013, pp. 280–300.
- [Che+16] Yu Chen et al. “Generic constructions of integrated PKE and PEKS”. In: *Des. Codes Cryptography* 78.2 (2016), pp. 493–526.

- [Che+17] Jung Hee Cheon et al. “Homomorphic Encryption for Arithmetic of Approximate Numbers”. In: *Advances in Cryptology - ASIACRYPT 2017*. Vol. 10624. Lecture Notes in Computer Science. Springer, 2017, pp. 409–437.
- [Che+20] Yu Chen et al. “PGC: Pretty Good Confidential Transaction System with Auditability”. In: *The 25th European Symposium on Research in Computer Security, ESORICS 2020*. <https://eprint.iacr.org/2019/319>. 2020, pp. 591–610.
- [Chi+20] Ilaria Chillotti et al. “TFHE: Fast Fully Homomorphic Encryption Over the Torus”. In: *J. Cryptol.* 33.1 (2020), pp. 34–91.
- [CKS08] David Cash, Eike Kiltz, and Victor Shoup. “The Twin Diffie-Hellman Problem and Applications”. In: *Advances in Cryptology - EUROCRYPT 2008*. Vol. 4965. LNCS. Springer, 2008, pp. 127–145.
- [CS02] Ronald Cramer and Victor Shoup. “Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption”. In: *Advances in Cryptology - EUROCRYPT 2002*. 2002, pp. 45–64.
- [CS98] Ronald Cramer and Victor Shoup. “A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack”. In: *Advances in Cryptology - CRYPTO 1998*. 1998, pp. 13–25.
- [CWZ18] Yu Chen, Yuyu Wang, and Hong-Sheng Zhou. “Leakage-Resilient Cryptography from Puncturable Primitives and Obfuscation”. In: *Advances in Cryptology - ASIACRYPT 2018*. 2018, pp. 575–606.
- [CZ14] Yu Chen and Zongyang Zhang. “Publicly Evaluable Pseudorandom Functions and Their Applications”. In: *9th International Conference on Security and Cryptography for Networks, SCN 2014*. 2014, pp. 115–134.
- [Dam+13] Ivan Damgård et al. “Bounded Tamper Resilience: How to Go beyond the Algebraic Barrier”. In: *Advances in Cryptology - ASIACRYPT 2013*. Vol. 8270. LNCS. Springer, 2013, pp. 140–160.
- [DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. “Non-Malleable Cryptography (Extended Abstract)”. In: *STOC*. ACM, 1991, pp. 542–552.
- [Den17] Yi Deng. “Magic Adversaries Versus Individual Reduction: Science Wins Either Way”. In: *Advances in Cryptology - EUROCRYPT 2017*. Vol. 10211. LNCS. 2017, pp. 351–377.
- [DH76] Whitefield Diffie and Martin E. Hellman. “New Directions in Cryptography”. In: *IEEE Transactions on Information Theory* 22(6) (1976), pp. 644–654.
- [DM15] Léo Ducas and Daniele Micciancio. “FHEW: Bootstrapping Homomorphic Encryption in Less Than a Second”. In: *Advances in Cryptology - EUROCRYPT 2015*. Vol. 9056. Lecture Notes in Computer Science. Springer, 2015, pp. 617–640.
- [Dod+08] Yevgeniy Dodis et al. “Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data”. In: *SIAM J. Comput.* 38.1 (2008), pp. 97–139.
- [DR99] Yevgeniy Dodis and Matthias Ruhl. *GM-Security and Semantic Security Revisited*. <http://people.csail.mit.edu/ruhl/papers/drafts/semantic.html>. 1999.
- [ElG85] Taher ElGamal. “A Public Key Cryptosystem and a Signature Scheme based on Discrete Logarithms”. In: *IEEE Transactions on Information Theory* 31 (1985), pp. 469–472.
- [Fau+19] Prastudy Fauzi et al. “Quisquis: A New Design for Anonymous Cryptocurrencies”. In: *Advances in Cryptology - ASIACRYPT 2019*. Vol. 11921. Lecture Notes in Computer Science. Springer, 2019, pp. 649–678.
- [Fipa] *Digital Signature Standard (DSS)*. <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5.pdf>.
- [Fipb] *Module-Lattice-Based Key-Encapsulation Mechanism Standard*. <https://csrc.nist.gov/pubs/fips/203/ipd>.

- [FV12] Junfeng Fan and Frederik Vercauteren. *Somewhat Practical Fully Homomorphic Encryption*. IACR Cryptol. ePrint Arch. <http://eprint.iacr.org/2012/144>. 2012.
- [Gar+13] Sanjam Garg et al. “Candidate Indistinguishability Obfuscation and Functional Encryption for all circuits”. In: *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013*. IEEE Computer Society, 2013, pp. 40–49.
- [Gen09] Craig Gentry. “Fully homomorphic encryption using ideal lattices”. In: *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009*. ACM, 2009, pp. 169–178.
- [GM82] Shafi Goldwasser and Silvio Micali. “Probabilistic Encryption and How to Play Mental Poker Keeping Secret All Partial Information”. In: *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, STOC 1982*. ACM, 1982, pp. 365–377.
- [GM84] Shafi Goldwasser and Silvio Micali. “Probabilistic Encryption”. In: *J. Comput. Syst. Sci.* 28.2 (1984), pp. 270–299.
- [GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. “Trapdoors for hard lattices and new cryptographic constructions”. In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, STOC 2008*. ACM, 2008, pp. 197–206.
- [Hal17] Shai Halevi. “Homomorphic Encryption”. In: *Tutorials on the Foundations of Cryptography*. Springer International Publishing, 2017, pp. 219–276.
- [Har+10] Kristiyan Haralambiev et al. “Simple and Efficient Public-Key Encryption from Computational Diffie-Hellman in the Standard Model”. In: *Public Key Cryptography - PKC 2010*. 2010, pp. 1–18.
- [HK09] Dennis Hofheinz and Eike Kiltz. “Practical Chosen Ciphertext Secure Encryption from Factoring”. In: *Advances in Cryptology - EUROCRYPT 2009*. Vol. 5479. LNCS. Springer, 2009, pp. 313–332.
- [HLL16] Shuai Han, Shengli Liu, and Lin Lyu. “Efficient KDM-CCA Secure Public-Key Encryption for Polynomial Functions”. In: *Advances in Cryptology - ASIACRYPT 2016*. Vol. 10032. LNCS. Springer, 2016, pp. 307–338.
- [Hof12] Dennis Hofheinz. “All-But-Many Lossy Trapdoor Functions”. In: *Advances in Cryptology - EUROCRYPT 2012*. Vol. 7237. Lecture Notes in Computer Science. Springer, 2012, pp. 209–227.
- [Iso] *IT Security Techniques – Encryption algorithms - Part 6: Homomorphic Encryption*. <https://www.iso.org/standard/67740.html>.
- [Kia+13] Aggelos Kiayias et al. “Delegatable pseudorandom functions and applications”. In: *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS 2013*. ACM, 2013, pp. 669–684.
- [Kil07] Eike Kiltz. “Chosen-Ciphertext Secure Key-Encapsulation Based on Gap Hashed Diffie-Hellman”. In: *Public Key Cryptography - PKC 2007*. Vol. 4450. LNCS. Full version is available at ePrint Archive: Report 2007/036. Springer, 2007, pp. 282–297.
- [KMO10] Eike Kiltz, Payman Mohassel, and Adam O’Neill. “Adaptive Trapdoor Functions and Chosen-Ciphertext Security”. In: *Advances in Cryptology - EUROCRYPT 2010*. 2010, pp. 673–692.
- [Kom16] Ilan Komargodski. “Leakage Resilient One-Way Functions: The Auxiliary-Input Setting”. In: *Theory of Cryptography - 14th International Conference, TCC 2016-B*. Vol. 9985. LNCS. Springer, 2016, pp. 139–158.
- [KR19] Yael Tauman Kalai and Leonid Reyzin. “A survey of leakage-resilient cryptography”. In: *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*. 2019, pp. 727–794.
- [Lib] *Kunlun*. <https://github.com/yuchen1024/Kunlun>.

- [Mic10] Daniele Micciancio. “Duality in Lattice Cryptography (invited talk)”. In: *Public Key Cryptography - PKC 2010*. Vol. 6056. Lecture Notes in Computer Science. Springer, 2010.
- [Mpc] <https://github.com/alibaba-edu/mpc4j/>.
- [NR04] Moni Naor and Omer Reingold. “Number-theoretic constructions of efficient pseudo-random functions”. In: *J. ACM* 51.2 (2004), pp. 231–262.
- [NS09] Moni Naor and Gil Segev. “Public-Key Cryptosystems Resilient to Key Leakage”. In: *Advances in Cryptology - CRYPTO 2009*. Vol. 5677. LNCS. Springer, 2009, pp. 18–35.
- [NY90] Moni Naor and Moti Yung. “Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attacks”. In: *Proceedings of the 22th Annual ACM Symposium on Theory of Computing, STOC 1990*. ACM, 1990, pp. 427–437.
- [Opea] <https://github.com/openssl>.
- [Opeb] *OpenFHE*. <https://github.com/openfheorg/openfhe-development>.
- [Pai99] Pascal Paillier. “Public-Key Cryptosystems Based on Composite Degree Residuosity Classes”. In: *Advances in Cryptology - EUROCRYPT 1999*. 1999, pp. 223–238.
- [Pkc] *PKCS#1: RSA Cryptography Specifications Version 2.2*. <https://www.rfc-editor.org/rfc/rfc8017.html>.
- [PW08] Chris Peikert and Brent Waters. “Lossy trapdoor functions and their applications”. In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, STOC 2008*. 2008, pp. 187–196.
- [Rab79] Michael Rabin. “Digitalized Signatures and Public-Key Functions as Intractable as Factorization”. In: *MIT Laboratory for Computer Science, Technical Report TR-212* (1979).
- [RAD78] R. Rivest, L. Adleman, and M. Dertouzos. “On Data Banks and Privacy Homomorphisms”. In: *Foundations of Secure Computation* (1978), pp. 169–179.
- [Reg05] Oded Regev. “On lattices, learning with errors, random linear codes, and cryptography”. In: *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, STOC 2005*. ACM, 2005, pp. 84–93.
- [Rfc] *Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation*. <https://datatracker.ietf.org/doc/html/rfc5639>.
- [RS09] Alon Rosen and Gil Segev. “Chosen-Ciphertext Security via Correlated Products”. In: *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009*. Vol. 5444. LNCS. Springer, 2009, pp. 419–436.
- [RS91] Charles Rackoff and Daniel R. Simon. “Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack”. In: *Advances in Cryptology - CRYPTO 1991*. Vol. 576. LNCS. 1991, pp. 433–444.
- [Sea] *SEAL*. <https://github.com/microsoft/SEAL>.
- [Seca] *SEC 1: Elliptic Curve Cryptography Ver. 1.0*. <https://www.secg.org/SEC1-Ver-1.0.pdf>.
- [Secb] *SEC 2: Recommended Elliptic Curve Domain Parameters*. <https://www.secg.org/sec2-v2.pdf>.
- [Sho] .
- [Sho04] Victor Shoup. *Sequences of games: a tool for taming complexity in security proofs*. IACR Cryptology ePrint Archive. <http://eprint.iacr.org/2004/332>. 2004.
- [Sho97] Peter W. Shor. “Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer”. In: *SIAM Journal on Computation* 26.5 (1997), pp. 1484–1509.
- [Sm2] *SM2 椭圆曲线公钥密码算法*. <https://www.rfc-editor.org/rfc/rfc8017.html>.

- [SW14] Amit Sahai and Brent Waters. “How to use indistinguishability obfuscation: deniable encryption, and more”. In: *Symposium on Theory of Computing, STOC 2014*. ACM, 2014, pp. 475–484.
- [Wee10] Hoeteck Wee. “Efficient Chosen-Ciphertext Security via Extractable Hash Proofs”. In: *Advances in Cryptology - CRYPTO 2010*. Vol. 6223. 2010, pp. 314–332.
- [Wee12a] Hoeteck Wee. “Dual Projective Hashing and Its Applications - Lossy Trapdoor Functions and More”. In: *Advances in Cryptology - EUROCRYPT 2012*. Vol. 7237. LNCS. Springer, 2012, pp. 246–262.
- [Wee12b] Hoeteck Wee. “Public Key Encryption against Related Key Attacks”. In: *Public Key Cryptography - PKC 2012*. 2012, pp. 262–279.
- [Wee16] Hoeteck Wee. “KDM-Security via Homomorphic Smooth Projective Hashing”. In: *Public-Key Cryptography - PKC 2016*. Vol. 9615. LNCS. Springer, 2016, pp. 159–179.
- [Zha16] Mark Zhandry. “The Magic of ELFs”. In: *Advances in Cryptology - CRYPTO 2016*. Vol. 9814. LNCS. Springer, 2016, pp. 479–508.

## 后记

终于糊弄完了.