计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2015.20131954
52(6):1394-1399,2015

基于单向函数的伪随机数发生器

高树静　曲英杰　宋廷强
(青岛科技大学信息科学技术学院　山东青岛　266061)
(shujing_g@126.com)

Pseudorandom Number Generators Based on One-Way Functions

Gao Shujing, Qu Yingjie, and Song Tingqiang
(*School of Information Science and Technology, Qingdao University of Science and Technology, Qingdao, Shandong* 266061)

Abstract　Pseudorandom number generators (referred as PRNG) is an important cryptographic primitive that was first introduced and formalized as BMY generator in 1982. The PRNG based on one-way functions is constructed by iterating a one-way function (OWF) on a random seed and generating pseudorandom sequences periodically. The seed length and the property of the one-way function are two important factors of this kind PRNG, which measure the efficiency and the security of the PRNG. The security of the latest PRNG of this type relies on one-way function of length preserving or one-way permutation that is hard to be obtained. This paper revisits the current randomized iteration technique and makes improvement on the iteration process by expanding the outputs of one-way function. The new technique, which is called expanded randomized iteration, eliminates the length preserving property of the one-way function. On the basis of the expanded randomized iteration, our construction uses the general compression regular one-way function and universal hash function as the main components. In the BMY case, a hardcore-bit of each iteration step is taken as the output of the pseudorandom sequence. Our scheme adopts the similar structure as the current ones but relaxes the requirement of the property of the one-way function, reduces the seed length and improves the efficiency. Finally, the security of the iteration is proved irreversible and the security of the proposed pseudorandom generator is proved undistinguishable from the real random sequence.

Key words　pseudorandom number generator(PRNG); one-way function(OWF); randomized iterate; hard-core predicate; universal Hash function

摘　要　伪随机数发生器(pseudorandom number generator, PRNG)是重要的密码学概念.基于单向函数的伪随机数发生器起始于1982年的BMY发生器,将单向函数反复迭代,周期性地输出伪随机序列.单向函数的性质和种子长度关系到发生器的可实现性和安全性,是此类发生器的2个重要参数.在分析现有工作的基础上,改进了单向函数的随机化迭代方式,基于不可逆性证明了迭代过程的安全性.迭代方式的改进消除了单向函数的长度保持性质,采用一般的压缩规范单向函数和通用散列函数构建伪随机数发生器.输出级与BMY发生器结构类似,以迭代函数的核心断言作为伪随机序列.基于与真随机序列的不可区分性,证明了伪随机数发生器的安全性.所构建的伪随机数发生器与现有同类发生器结构类似,但放松了对单向函数性质的要求,增强了可实现性,减小了种子长度,提高了效率.

收稿日期:2013-12-25;**修回日期**:2014-08-12
基金项目:山东省科技发展计划基金项目(2013YD01038)

关键词 伪随机数发生器;单向函数;随机化迭代;核心断言;通用散列函数

中图法分类号 TP309

伪随机序列(pseudorandom sequence)的应用非常广泛,如协议认证、信息加密等,是密码学的重要概念.在密码学中,用于产生统计特性与真随机序列尽量接近的确定性序列,减少敌手随机猜测的成功概率[1].

伪随机数发生器(pseudorandom number generator, PRNG)的构造方式有多种.其中由于混沌系统具有良好的伪随机性和不可预测性,因此基于混沌理论的伪随机数发生器研究引起了广泛关注[2-3].然而由于混沌系统复杂性较高,此类发生器的硬件实现比较困难.而单向函数是比较容易获得的,且依赖单向函数的不可逆性可以构成安全性较高的伪随机数发生器.文献[4]指出:伪随机数发生器是存在的,当且仅当单向函数是存在的.

基于单向函数的伪随机数发生器的安全性依赖单向函数的单向性.对此类发生器进行评价的2个主要参数为:对单向函数的性质要求和种子长度.对单向函数的性质要求越高,则PRNG的可实现性越差;种子长度越长,则PRNG的安全性越差.一般采用伪随机数发生器的种子长度 m 和单向函数的输入长度 n 的比值来衡量PRNG的安全性.

文献[5]是第1个关于单向函数构建伪随机数发生器的研究,由Blum等人在1982年提出,并由Yao[6]进行了具体实现,称BMY发生器.BMY发生器的最大优点是种子长度满足线性关系,即 $m=O(n)$;然而BMY发生器需要单向函数满足单向和置换的特点,此类单向函数很难获得,因此难以实现.

文献[7]在先前工作[8-9]的基础上,构建了HILL发生器,其思想是采用没有任何结构限制的任意单向函数来构建伪随机数发生器.与BMY发生器相比,这类发生器的最大优点是对所采用的单向函数限制最少;但其缺点是PRNG的构建过程复杂、计算很多、种子长度较长且为 $m=O(n^8)$、效率很差.对HILL发生器的改进都集中在减小种子数量上[10-12].其中文献[10]利用文献[11]提出的下一位伪熵(next-bit pseudoentropy)概念,简化了基于任意单向函数的伪随机数发生器构建方式,在文献[11]的研究基础上,将种子长度从 $m=O(n^4)$ 进一步降低为 $m=O(n^3)$,结果仍然不太理想.

GKL[13]发生器对单向函数的限制放松为任意的规范单向函数.为了克服BMY简单迭代的安全隐患,GKL基于BMY在单向函数的2次迭代之间引入随机因素,称为随机化迭代(randomized iterate),使得单向函数在每一次的迭代输入都具有随机性.GKL的种子长度为 $m=O(n^3)$,效率较差.此后,文献[14]采用通用单向散列函数(universal one-way Hash function, UOWHF),将GKL发生器的种子长度进一步降低为 $m=O(n\log n)$.文献[15]基于文献[11]的研究进行了改进,种子长度相同,但是将单向函数的要求放松为规范单向函数(regular OWF).文献[16]将BMY和GKL这2种技术结合在一起,将种子长度降低为 $m=O(n)$,但是要求采用的单向函数和散列函数均是长度保持的函数,也就是置换函数,因此文献[16]与BMY发生器的实质是相同的,不能获得其声称的效率.

本文采用BMY构建方式,首先对GKL随机化迭代方式进行了扩展,即扩展随机化迭代;然后采用压缩规范单向函数构建了PRNG,并证明了迭代过程的安全性和PRNG的安全性.

本文所设计的PRNG在单向函数的性质要求和种子长度2方面进行了改进.首先消除对单向函数的长度保持性限制,即为一般的压缩规范单向函数;其次使种子的长度与单向函数的输入长度保持线性关系 $O(n)$,进一步提高效率和安全性.

1 预备知识

1.1 核心断言

文献[6]给出了关于核心断言(hard-core predicate)的重要结论,对于任何单向函数 f,都存在一个断言 $b(\boldsymbol{x})$,使得根据 $f(\boldsymbol{x})$ 猜测 $b(\boldsymbol{x})$ 的难度与求 f 的反函数的难度相当.在PRNG的构建中,利用函数的核心断言获得伪随机序列的可预测性.

定义1. 核心断言[6].称函数 $b:\{0,1\}^\lambda\to\{0,1\}$ 是另外一个函数 $f:\{0,1\}^\lambda\to\{0,1\}^\lambda$ 的核心断言,如果 b 是高效可计算的,并且对于任何断言者(predicator) P 有 $|Pr[\boldsymbol{x}\leftarrow_R\{0,1\}^\lambda;\mathcal{Y}\leftarrow f(\boldsymbol{x}):P(\mathcal{Y})=b(\boldsymbol{x})]-1/2|\leqslant\varepsilon$,其中 ε 是可忽略的.

换句话说,函数的核心断言是这样一个函数:敌手要猜测函数 $b:\{0,1\}^\lambda\to\{0,1\}$ 原像的难度相当于猜测函数 $f:\{0,1\}^\lambda\to\{0,1\}^\lambda$ 原像的难度.

文献[17]构建了一个适用于所有单向函数的通用核心断言,称为 GL-核心断言. 对于 2 个位串 $\boldsymbol{x}(=x_1 \| \cdots \| x_m)$ 和 $\boldsymbol{r}(=r_1 \| \cdots \| r_m)$,定义 $b(\boldsymbol{x},\boldsymbol{r})=\langle \boldsymbol{x},\boldsymbol{r}\rangle$,是 $\boldsymbol{x}$ 和 $\boldsymbol{r}$ 的模 2 向量内积,即 $b(\boldsymbol{x},\boldsymbol{r})=\langle \boldsymbol{x},\boldsymbol{r}\rangle=\bigoplus_{i=1}^{m}(x_i r_i)$.

假设函数 f 和 g 具有相同的值域 $S_m \subseteq \{0,1\}^m$. 对于任意 $\boldsymbol{x}\in S_m$ 和 $\boldsymbol{r}\in\{0,1\}^m$,定义函数 $\hat{f}$ 为 $\hat{f}(\boldsymbol{x},\boldsymbol{r})=(f(\boldsymbol{x}),\boldsymbol{r})$,其 GL-核心断言 b 是 $\langle \boldsymbol{z},\boldsymbol{r}\rangle$,其中 $\boldsymbol{z}\in Preim(g,f(\boldsymbol{x}))$ 是函数 $f(\boldsymbol{x})$ 在 g 下的原像.

任意一个单向函数都可以通过这种方式转换为另外一个单向函数,使其具有核心断言.

1.2 BYM 发生器

第 1 个利用单向函数迭代构建的 PRNG 称为 BMY-发生器[5-6]. 如图 1 所示:

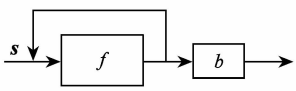


Fig. 1 BMY generator.

图 1 BMY 发生器

定义 2. BMY 发生器. 假设 $f:\{0,1\}^\lambda\to\{0,1\}^\lambda$ 是任意的单向置换,$b:\{0,1\}^\lambda\to\{0,1\}$ 是 f 的核心断言. 对于输入 $\boldsymbol{s}$,函数 f 的第 k 次递归迭代定义为 $f^{(k)}(\boldsymbol{s})=f(f^{(k-1)}(\boldsymbol{s}))$. 通过对每一次迭代计算核心断言,伪随机数发生器 $G(\boldsymbol{s},n)$ 构建为 $b(f^{(0)}(\boldsymbol{s}))\|\cdots\|b(f^{(n-1)}(\boldsymbol{s}))$,则序列 $G(\boldsymbol{s},n)\| f^{(n)}(\boldsymbol{s})$ 与 $(n+\lambda)$ 位的随机序列是不可区分的.

根据定义 2,可以得到以下推论:

推论 1. 对于任意的多项式时间算法 D, $|Pr[\boldsymbol{s}\leftarrow_R\{0,1\}^\lambda: D(G(\boldsymbol{s},n)\| f^{(n)}(\boldsymbol{s}))=1]-Pr[\boldsymbol{z}\leftarrow_R\{0,1\}^{n+\lambda}: D(\boldsymbol{z})=1]|\leqslant\varepsilon$. 其中 ε 是可忽略的.

推论的证明在文献[5]中给出.

2 基于压缩规范单向函数的伪随机数发生器

2.1 扩展随机化迭代

经过验证,只有当选择的单向置换是迭代单向保持(one-way on iterate, OWI)函数时(即当反复将函数的输入加载到单向函数上时,函数将仍然保持单向性),BMY 发生器才可以达到设计目标[18]. 大多数已知的一般单向函数不会具备这样的特点,因为其输出的随机性变小,单纯的迭代会损失熵,甚至第 2 次迭代就很容易被翻转.

为了消除 BMY 发生器对单向函数的严格要求,需要对简单的迭代方式进行改变. 文献[13]使用了一种称为随机迭代(randomized iterate)技术,在每次迭代中加入随机因素,使得每次迭代的输出均匀分布,并采用规范的长度保持单向函数构建 GKL 发生器.

然而规范的长度保持函数即是单向映射,因此 GKL 发生器及其改进[15]与 BMY 发生器的实质是相同的. 本节在 GKL 的随机迭代技术基础上,提出了一种扩展随机化迭代方法,基于压缩规范单向函数和通用散列函数构建伪随机数发生器,对其压缩过程进行扩展,形成伪长度保持函数,并证明了这种扩展随机化迭代方法是安全的.

定义 3. 扩展随机化迭代. $f:\{0,1\}^m\to\{0,1\}^n$ 是规范单向函数,$H:\{0,1\}^m\to\{0,1\}^n$ 是通用散列函数族,其中 $m>n$ 且 m 是 n 的倍数,并假设 $m/n=L$,即 f 和 H 都是压缩函数. 输入数据 $\boldsymbol{x}\in\{0,1\}^m$ 被 f 经过 L 步压缩运算,每一步得到 n 位输出,最后一个输出是函数值. 将 L 次压缩得到的 L 个 m 位输出作为 $h\in H$ 的 m 位输入. 同样,散列函数的 L 次压缩结果又作为 f 的输入.

假设在第 i 次迭代中,f 在第 j 次压缩的输出为 $\hat{\boldsymbol{x}}_j^i$,长度为 n 位;第 i 次迭代结束的扩展输出为 $\hat{\boldsymbol{x}}^i=(\hat{\boldsymbol{x}}_1^i,\cdots,\hat{\boldsymbol{x}}_j^i,\cdots,\hat{\boldsymbol{x}}_L^i)$,长度为 $L\times n=m$ 位,而 f 的输出为 n 位的第 L 次压缩的结果 $\boldsymbol{x}^i=\hat{\boldsymbol{x}}_L^i$.

f 和 H 的每一次压缩相当于长度保持的函数运算. 为了便于描述,定义长度保持函数 $\hat{f}:\{0,1\}^n\to\{0,1\}^n$ 和 $\hat{H}:\{0,1\}^n\to\{0,1\}^n$,$\hat{f}':\{0,1\}^m\to\{0,1\}^m$ 和 $\hat{H}':\{0,1\}^m\to\{0,1\}^m$. 其中 $\hat{f}'$ 和 $\hat{H}'$ 分别是 $\hat{f}$ 和 $\hat{H}$ 的各次迭代函数.

假设第 $k-1$ 次迭代已经结束,得到 f 的扩展输出为 $\hat{\boldsymbol{x}}^{k-1}=(\hat{\boldsymbol{x}}_1^{k-1},\cdots,\hat{\boldsymbol{x}}_L^{k-1})$,而 f 的输出为 $\boldsymbol{x}^{k-1}=\hat{\boldsymbol{x}}_L^{k-1}$.

在接下来的第 k 次迭代中,从通用散列函数 $H:\{0,1\}^m\to\{0,1\}^n$ 中选择的是函数 h_k,则上述迭代过程可以描述为 $\hat{\boldsymbol{x}}^k=\hat{f}(\hat{h}_k(\hat{\boldsymbol{x}}^{k-1}))$,$\hat{f}(\hat{h}_k(\hat{\boldsymbol{x}}^{k-1}))=(\hat{f}'(\hat{h}_k'(\hat{\boldsymbol{x}}_1^{k-1})),\cdots,\hat{f}'(\hat{h}'(\hat{\boldsymbol{x}}_L^{k-1})))=\hat{\boldsymbol{x}}^k$.

$\hat{\boldsymbol{x}}^k=\hat{f}^k(\boldsymbol{x},h_1,\cdots,h_k)$,第 k 次扩展随机迭代 $\hat{f}^k:\{0,1\}^m\times H^k\to \mathrm{Im}(\hat{f})$ 可以递归地描述为

$$\hat{f}^k(\boldsymbol{x},h_1,\cdots,h_k)=\hat{f}(\hat{h}_k(\hat{f}^{k-1}(\boldsymbol{x},h_1,\cdots,h_{k-1}))),$$

其中 $\hat{f}^0(\boldsymbol{x})=\hat{f}(\boldsymbol{x})$.

如图 2 所示,在扩展随机化迭代中,$f:\{0,1\}^m\to\{0,1\}^n$ 和 $H:\{0,1\}^m\to\{0,1\}^n$ 都是压缩函数,均需要 L 步将 m 位输入压缩为 n 位结果.

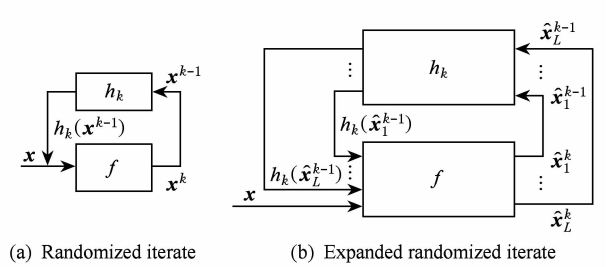


Fig. 2 Randomized iterate and the expansion.

图 2 随机化迭代及其扩展随

在这里,我们将证明上述随机化迭代的安全性,即按照定义 3,对规范压缩单向函数 f 进行了 k 次扩展随机化迭代之后,即使敌手在已知所有散列函数的情况下,最后一次迭代也很难被求逆.

定理 1. 假设 f 是规范单向函数,H 是通用散列函数族,且 $\boldsymbol{x}^k$ 是函数 f 根据定义 3 的第 k 次随机化迭代结果. 则对于任意概率多项式时间敌手(probabilistic polynomial time adversary, PPT A)和任意的 k 有:

$$Pr_{(\boldsymbol{x},h_1,\cdots,h_k)\leftarrow(U_n,H^k)}[A(\boldsymbol{x}^k,h_1,\cdots,h_k)=\hat{\boldsymbol{x}}^{k-1}]<\varepsilon.$$

证明过程见附录 A.

2.2 PRNG 构建及安全性证明

我们构造的基于扩展的随机化迭代 PRNG 如图 3 所示. 其中,LFSR(linear feedback shift register)是线性反馈移位寄存器.

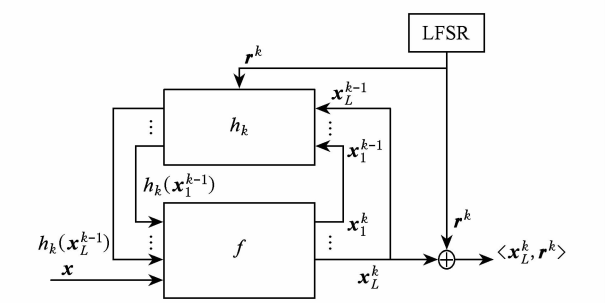


Fig. 3 PRNG based on expanded randomized iterate.

图 3 基于扩展随机化迭代的 PRNG

根据扩展随机化迭代的定义,第 k 次迭代时,长度保持函数 $\hat{f}:\{0,1\}^n\to\{0,1\}^n$ 的输出为 $\hat{\boldsymbol{x}}^k=(\boldsymbol{x}_1^k,\cdots,\boldsymbol{x}_L^k)$,函数 f 的输出是 $\hat{f}:\{0,1\}^n\to\{0,1\}^n$ 的最后 n 位 $\boldsymbol{x}^k=\hat{\boldsymbol{x}}_L^k$. 为了描述 PRNG 的构成,进行如下定义:

$\boldsymbol{x}^k=f^k(\boldsymbol{x},h_1,\cdots,h_k)$,第 k 次随机迭代 $f^k:\{0,1\}^m\times H^k\to\mathrm{Im}(f)$ 可以递归地描述为 $f^k(\boldsymbol{x},h_1,\cdots,h_k)=f(\hat{h}_k(\hat{f}^{k-1}(\boldsymbol{x},h_1,\cdots,h_{k-1})))$,其中 $f^0(\boldsymbol{x})=f(\boldsymbol{x})$.

我们已经证明,上述的随机化迭代方式很难被翻转. 下面将按照与 BMY 相同的方式,采用上述的扩展随机化迭代方式,构建伪随机数发生器,即在每一次迭代之后,将函数 f 的核心断言逐位输出,构成伪随机数序列.

定理 2. 假设 $f:\{0,1\}^m\to\{0,1\}^n$ 是一个规范单向函数,H 是一个通用散列函数族,设 G 为 $G(\boldsymbol{x},H,\boldsymbol{r})=(b_r(\boldsymbol{x}^0),\cdots,b_r(\boldsymbol{x}^n),h_1,\cdots,h_n,\boldsymbol{r})$,其中 $\boldsymbol{x}\in\{0,1\}^m$ 且 $h_1,\cdots,h_n\in H$. $\boldsymbol{x}^0=f(\boldsymbol{x})$ 且 $\boldsymbol{x}^i=f(\hat{h}_i(\hat{f}^{i-1}(\boldsymbol{x},h_1,\cdots,h_{i-1})))(1\leqslant i\leqslant n)$. $b_r(\boldsymbol{x}^i)$ 代表 $\boldsymbol{x}^i$ 的 GL-核心断言.

则 G 为伪随机数发生器.

证明. 根据伪随机性的定义,若一个序列与真随机序列是不可区分的,则为伪随机序列. 而文献[18]进一步用混合论证法证明,一个序列与真随机序列是不可区分的,当且仅当在已知序列所有前缀的情况下很难预测序列的下一位,将不可区分性转换为不可预测性.

因此,对于本文所构建的伪随机数发生器 G,需要证明在已知 $G(\boldsymbol{x},H,\boldsymbol{r})=(b_r(\boldsymbol{x}^n),\cdots,b_r(\boldsymbol{x}^i),h_1,\cdots,h_{k-1},\boldsymbol{r})$的情况下,不可能预知序列的下一个位 $b_r(\boldsymbol{x}^{i-1})$.

在这里,我们利用反证法,即假设根据 $G(\boldsymbol{x},H,\boldsymbol{r})=(b_r(\boldsymbol{x}^n),\cdots,b_r(\boldsymbol{x}^i),h_1,\cdots,h_{k-1},\boldsymbol{r})$,可以预知序列的下一位 $b_r(\boldsymbol{x}^{i-1})$.

假设存在一个算法 D,根据 $(b_r(\boldsymbol{x}^n),\cdots,b_r(\boldsymbol{x}^i),h_1,\cdots,h_{k-1},\boldsymbol{r})$ 成功预测 $b_r(\boldsymbol{x}^{i-1})$ 的概率定义为 $Pr[D(b_r(\boldsymbol{x}^n),\cdots,b_r(\boldsymbol{x}^i))=b_r(\boldsymbol{x}^{i-1})]$. 设存在一个算法 D',在已知 $\boldsymbol{x}^n,\cdots,\boldsymbol{x}^i$ 的条件下预测 $\boldsymbol{x}^{i-1}$,预测成功的概率定义为 $Pr[D'(\boldsymbol{x}^n,\cdots,\boldsymbol{x}^i)=\boldsymbol{x}^{i-1}]$. 由于 $b_r(\boldsymbol{x}^{i-1})=\langle\boldsymbol{x}^{i-1},\boldsymbol{r}\rangle$ 为函数 f 的核心断言,因此算法 D 的预测建立在 D' 的预测基础上,因此有 $Pr[D'(\boldsymbol{x}^n,\cdots,\boldsymbol{x}^i)=\boldsymbol{x}^{i-1}]>Pr[D(b_r(\boldsymbol{x}^n),\cdots,b_r(\boldsymbol{x}^i))=b_r(\boldsymbol{x}^{i-1})]$. 在随机化迭代中已经证明,在已知 $\boldsymbol{x}^i$ 的条件下求 $\hat{\boldsymbol{x}}^{i-1}$ 的概率为 $Pr_{(\boldsymbol{x},h_1,\cdots,h_k)\leftarrow(U_n,H^k)}[A(\boldsymbol{x}^k,h_1,\cdots,h_k)=\hat{\boldsymbol{x}}^{k-1}]<\varepsilon$.

因此 $Pr[D(b_r(\boldsymbol{x}^n),\cdots,b_r(\boldsymbol{x}^i))=b_r(\boldsymbol{x}^{i-1})]<Pr[D'(\boldsymbol{x}^n,\cdots,\boldsymbol{x}^i)=\boldsymbol{x}^{i-1}]<\varepsilon$. 即伪随机序列是不可预测的. 证毕.

在构建的 PRNG 中,我们采用 LFSR 的输出作为布尔向量 $\boldsymbol{r}$,需要种子数量为 n;第 1 次调用单向

函数 f 时需要输入 $\boldsymbol{x}$ 为 m 位的;迭代过程中的通用散列函数选择也采用布尔向量 $\boldsymbol{r}$ 进行,LFSR 采用本原多项式进行构造,确保足够长的周期性.因此总共需要的 PRNG 种子长度为 $m=O(n)$,获得了与 BMY 相当的效率.

3 总 结

本文设计了一个基于单向函数的伪随机数发生器.我们首先将 GKL 的随机化迭代技术进行扩展,称为扩展随机化迭代,使得单向函数可以是非长度保持函数;然后以 BMY 发生器结构为基础,结合扩展随机化迭代,设计了伪随机数发生器.经过证明,所设计的伪随机数发生器的输出序列与真随机序列是不可区分的,且种子长度为 $O(n)$,与单向函数的输入保持线性关系,进一步提高了安全性.

参 考 文 献

[1] Ishai Y, Kushilevitz E, Li X, et al. Robust Pseudorandom Generators [G] //Automata, Languages, and Programming. Berlin: Springer, 2013: 576-588

[2] Liu Jiandong, Yang Kai, Yu Youming. Improved coupled tent map lattices model and its characteristics analysis [J]. Journal of Computer Research and Development, 2011, 48(9): 1667-1675 (in Chinese)
(刘建东,杨凯,余有明. 改进型耦合帐篷映像格子模型及其性能分析[J]. 计算机研究与发展, 2011, 48(9): 1667-1675)

[3] Wang Fulai. A new pseudo-random number generator and application to digital secure communication scheme based on compound symbolic chaos [J]. Acta Physica Sinica, 2011, 6(11): 110517 (in Chinese)
(王福来. 基于复合符号混沌的伪随机数生成器及加密技术[J]. 物理学报, 2011, 6(11): 110517

[4] Iftach H, Omer R, Salil V. Efficiency improvements in constructing pseudorandom generators from one-way functions [C] //Proc of the 42nd Annual ACM Symp on Theory of Computing (STOC). New York: ACM, 2010: 437-446

[5] Blum M, Micali S. How to generate cryptographically strong sequences of pseudo random bits [C] //Proc of Annual Symp on Foundations of Computer Science. Piscataway, NJ: IEEE, 1982: 112-117

[6] Yao A C. Theory and application of trapdoor functions [C] // Proc of IEEE Symp on Foundations of Computer Science. Piscataway, NJ: IEEE, 1982: 80-91

[7] Håstad J, Impagliazzo R, Levin L A, et al. A pseudorandom generator from any one-way function [J]. SIAM Journal of Computing, 1999, 29(4): 1364-1396

[8] Impagliazzo R, Levin L A, Luby M. Pseudo-random generation from one-way functions(extended abstracts)[C] // Proc of the 21st Annual ACM Symp on Theory of Computing. New York: ACM, 1989: 12-24

[9] Håstad J. Pseudo-random generators under uniform assumptions [C] //Proc of the 21st Annual ACM Symp on Theory of Computing. New York: ACM, 1990: 395-404

[10] Vadhan S, Zheng C J. Characterizing pseudoentropy and simplifying pseudorandom generator constructions [C] //Proc of the 44th Symp on Theory of Computing. New York: ACM, 2012: 817-836

[11] Haitner I, Reingold O, Vadhan S. Efficiency improvements in constructing pseudorandom generators from one-way functions [C] //Proc of the 42nd ACM Symp on Theory of Computing. New York: ACM, 2010: 437-446

[12] Baron J, Ishai Y, Ostrovsky R. On Linear-Size Pseudorandom Generators and Hardcore Functions [G] // Computing and Combinatorics. Beilin: Springer, 2013: 169-181

[13] Goldreich O, Krawczyk H, Luby M. On the existence of pseudorandom generators [J]. SIAM Journal of Computing, 1993, 22(6): 1163-1175

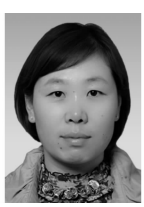
[14] Haitner I, Harnik D, Reingold O. On the Power of the Randomized Iterate [G] //Advances in Cryptology-CRYPTO 2006. Berlin: Springer, 2006: 22-40

[15] Boldyreva A, Kumar V. A new pseudorandom generator from collision-resistant hash functions [C] //Proc of the Cryptographers' Track at the RSA. Berlin: Springer, 2012: 187-202

[16] Goldreich O, Levin L A. A hard-core predicate for all one-way functions [C] //Proc of the 21st Annual ACM Symp on Theory of Computing. New York: ACM, 1989: 25-32

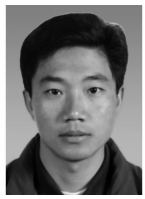
[17] Levin L A. One-way functions and pseudorandom generators [J]. Combinatorica, 1987, 7(4): 357-363

[18] Ames S, Gennaro R, Venkitasubramaniam M. The generalized randomized iterate and its application to new efficient constructions of UOWHFs from regular one-way functions [G] //Advances in Cryptology-ASIACRYPT 2012. Berlin: Springer, 2012: 154-171



Gao Shujing, born in 1976. Received her PhD from Shandong University. Lecturer in Qingdao University of Science and Technology. Her main research interests include information security, RFID system and application.

Qu Yingjie, born in 1964. PhD and professor in Qingdao University of Science and Technology. His main research interests include information security, IC design and computer architecture.



Song Tingqiang, born in 1971. PhD and associate professor in Qingdao University of Science and Technology. His main research interests include RFID application and embedded system.

附录 A. 迭代安全性证明.

证明. 采用反证法.

假设存在一个算法 M^A,给定 $\boldsymbol{x}^k=f^k(\boldsymbol{x},h_1,\cdots,h_k)$,能够以至少为 ε 的概率计算出 $\hat{\boldsymbol{x}}^{k-1}$,其中 ε 满足 $\varepsilon(n)=1/poly(n)$,即

$$\underset{(\boldsymbol{x},h_1,\cdots,h_k)\leftarrow(U_n,H^k)}{Pr}[f(\hat{h}_k(A(\boldsymbol{x}^k,h_1,\cdots,h_k)))=\boldsymbol{x}^k]\geqslant\varepsilon.$$

我们的目的是采用这个算法来打破单向函数 $\hat{f}$ 的单向性.

算法 M^A 的输入为 $z\in\mathrm{Im}(\hat{f})$.

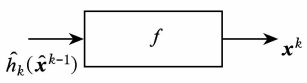
1) 随机地选择 $\hat{h}_1,\cdots,\hat{h}_k\in H$;

2) 计算 $\mathcal{Y}=A(z,\hat{h}_1,\cdots,\hat{h}_k)$;

3) 如果 $\hat{f}(\hat{h}_k(\mathcal{Y}))=z$,则输出 $h_k(\mathcal{Y})$,否则结束.

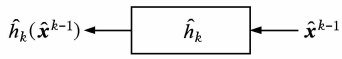
M^A 的求逆过程分为 2 个阶段:

1) 对 f 求逆.



设对 f 求逆成功的概率为 P_1.

2) 对 $\hat{h}_k$ 求逆.



对 $\hat{h}_k$ 求逆成功的概率为

$$P_2=Pr[\hat{h}_k(A(\hat{h}_k(\hat{\boldsymbol{x}}^{k-1})))=\hat{\boldsymbol{x}}^{k-1}],$$

其中 $\hat{\boldsymbol{x}}^{k-1}=(\hat{\boldsymbol{x}}_1^{k-1},\cdots,\hat{\boldsymbol{x}}_L^{k-1})$.

敌手 A 攻击成功的概率为

$$\underset{(\boldsymbol{x},h_1,\cdots,h_k)\leftarrow(U_n,H^k)}{Pr}[f(\hat{h}_k(A(\boldsymbol{x}^k,h_1,\cdots,h_k)))=\boldsymbol{x}^k]=P_1P_2\geqslant\varepsilon,$$

$$P_1\geqslant\frac{\varepsilon}{P_2}.$$

由于 $P_2\leqslant1$,因此有 $P_1\geqslant\varepsilon$. 这与 f 的单向性冲突,因此假设不成立,即

$$\underset{(\boldsymbol{x},h_1,\cdots,h_k)\leftarrow(U_n,H^k)}{Pr}[f(\hat{h}_k(A(\boldsymbol{x}^k,h_1,\cdots,h_k)))=\boldsymbol{x}^k]<\varepsilon.$$

证毕.