doi:10.3969/j.issn.1000-436x.2013.04.008

基于(*t*, *n*)门限和划分树的可再生散列链构造方案

黄海平[1,2,4]，戴庭[1,2]，王汝传[1,2,3]，秦小麟[4]，陈九天[1]

(1. 南京邮电大学 计算机学院，江苏 南京 210003；2. 江苏省无线传感网高技术研究重点实验室，江苏 南京 210003；
3. 宽带无线通信与传感网技术教育部重点实验室，江苏 南京 210003；
4. 南京航空航天大学 计算机科学与技术学院，江苏 南京 210016)

摘 要：针对可再生散列链解决了资源受限的缺点，但现有构造方案在安全性和复杂性等方面存在缺陷这一问题，提出“重复”、“划分”和“划分树”的定义，以及基于(*t*, *n*)-Mignotte's 门限的中国剩余定理秘密共享方案，设计了一种新的可再生散列链构造方法。从明文空间、双重认证和可证明安全 3 个方面论证了新构造方案能确保新链中种子值的安全再生并有效抵制中间人攻击。仿真实验表明新构造方案在通信、计算和存储开销等方面相比于传统方案具有相同甚至更佳的性能。

关键词：划分树；可再生散列链；(*t*, *n*)-Mignotte's 门限方案；中国剩余定理

中图分类号：TP309 **文献标识码**：A **文章编号**：1000-436X(2013)04-0070-12

Novel self-renewal hash chain scheme based on (*t*, *n*) threshold and division tree

HUANG Hai-ping[1, 2, 4], DAI Ting[1, 2], WANG Ru-chuan[1, 2, 3], QIN Xiao-lin[4], CHEN Jiu-tian[1]

(1. College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, China;
2. Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks, Nanjing 210003, China;
3. Key Lab of Broadband Wireless Communication and Sensor Network Technology of Ministry of Education, Nanjing 210003, China;
4. College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China)

Abstract: The introduction of renewal hash chain overcame resource-constrained defect in traditional hash chains, but the existing renewable schemes had still held unsatisfactory performance especially on security and complexity. The definitions of repetition, division and division-tree was proposed, and then a novel self-renewable hash chain construction scheme was put forward based on division and (*t*, *n*)-Mignotte's threshold Chinese remainder theorem secret sharing scheme. From three aspects of key space, twice authent ion and provable security, it theoretically proves that the proposed hash scheme could ensure the novel seed value regenerated safely and resisting the middle-man attack effectively. Simulation experiments demonstrate that the novel scheme obtains equal or more satisfactory performances on the costs of communication, computation and storage than typical schemes.

Key words: division-tree; renewal hash chain; (*t*, *n*)-Mignotte's threshold; Chinese remainder theorem

---

收稿日期：2012-07-09；**修回日期**：2012-11-20

基金项目：国家自然科学基金资助项目(61170065, 61003039)；江苏省科技支撑计划（工业）基金资助项目(BE2012183)；江苏省属高校自然科学研究重大基金资助项目(12KJA520002)；国家博士后基金资助项目(2012M511753)；江苏省博士后基金资助项目(1101011B)；江苏高校科技创新计划基金资助项目(CXLX12-0486)；江苏高校优势学科建设工程基金资助项目(信息与通信工程, yx002001)

Foundation Items: The National Natural Science Foundation of China (61170065, 61003039); Scientific & Technological Support Project (Industry) of Jiangsu Province (BE2012183); The Natural Science Key Fund for Colleges and Universities of Jiangsu Province (12KJA520002); Postdoctoral Foundation of China (2012M511753); Postdoctoral Foundation of Jiangsu Province (1101011B); Science & Technology Innovation Fund for Higher Educat Institutions of Jiangsu Province(CXZZ11-0486); The Priority Academic Program Development of Jiangsu Higher Education Institutions (Information & Communication Engineering, yx00 1)

1　引言

散列函数具有单向性并且计算效率高的特点，因此，散列链机制被广泛运用在各种加密应用和服务中，例如一次性口令(OTS)[1]、数字签名机制、密钥分配[2,3]、微型支付系统[4]、广播认证[5,6]、视频流安全[7,8]等。然而，这些应用大都受到一个共同的限制，即散列链的长度是有限的[9]。散列链的长度存在一个矛盾：太短则消耗过快，用尽后，系统必须重新初始化，即需要再生新的散列链，而且散列链的再生一般都还要与系统初次启动一样使用公钥签名技术，这严重有损于系统的效率[9]，造成计算复杂性提高；而散列链太长的话，首先，会造成存储开销增加，其次，会降低散列链的使用效率，最后，发送方在初始化阶段就需要进行链长次的散列运算操作，不仅增加了工作负担而且时间分配也不合理。

为解决这一矛盾，很多文献提出了新的散列链的构造方案。2004 年，文献[10]提出了可再生散列链(RHC, re-initializable hash chain)的构造方案。RHC 的构造主要思想是：当一个 RHC 用尽以后，能够以不可否认的方式安全地再生，从而得到另一个 RHC。

2006 年，文献[9]在文献[10]的基础上提出了精巧可再生散列链(ERHC, elegant re-initializable hash chain)的构造方案：1) 网络初始化时，发送方随机生成 $L+\lfloor \mathrm{lb}(L) \rfloor+1$ 个随机数，将它们的级联记作 S_U，并对这些随机数分别散列后级联，记作 P_U；再以 P_U 为根，计算一条长度为 N 的散列链，链首为 $h^N(P_U)$，以安全方式发送给接收方；2)待散列链耗尽后，发送方生成新的实 S_U'和 P_U'，并生成一条链首为 $h^N(P_U')$的新链。在发送新链的链首 $h^N(P_U')$时，只要发送需公开的部分 S_U；让接收方通过 P_U 和 S_U 的公开部分这两者的结合来验证 $h^N(P_U')$。具体算法详细过程可参见文献[9]。ERHC 在旧链消耗完毕后能够平滑地生成新链，概念上对有限长度散列链进行了自然、合理的扩展，逻辑上构造了无限长的散列链。但是，由于发送方通过透露部分 S_U 信息来认证下一条链的链首，会存在选择明文攻击。

2008 年，文献[11]提出了自更新散列链(SUHC, self-updating hash chains)的构造方案。该方案基于 Hard Core Predicate 算法，在分配第一条链的散列密钥值时，顺便分配第二条链链首的 1 个比特值。这样，当第一条密钥链用完时，第二条链首的所有比特都分配完（即得到第二条链首值）。

2008 年，文献[12]对文献[11]进行了改进，并提出了理想自更新散列链(SRHC, self-renewal hash chain)的构造方案。两方案的主要区别在于随机数选择上不同，SUHC 中是选择满足 $B(SR_i)=?\ '[i]$条件的随机数 SR_i，并且对 SR_i 做散列计算得到 PR_i，之后是对 PR_i 的操作；而 SRHC 中是选择满足 $B(h^{k-i}(s)||R_i)=?\ '[i]$条件的随机数 R_i，并直接对 R_i 进行操作。但是，SUCH 和 SRCH 中都存在以下的缺点：将新链链首值的每 1bit 映射到一个随机数，对链首值的安全公布即是对链长个随机数的安全公布。显然，两方案不可避免地要求所有的随机数都必须完整地被接收，这样才能完整地重构新链的链首，显然剔除了散列链原有的容错性的特点。

2009 年，文献[13]在文献[11]的基础上提出了新的自更新散列链(NSUHC, new self-updating hash chain)的构造方案。2010 年，文献[14]在文献[13]的基础上提出了基于纠羽码(erasure coding)的自更新散列链(SUHC-EC)的构造方案。两方案的基本思想：前者是将新密钥链的种子值(非密钥链链首)从 k 维扩充到 n 维，而后者则是将新密钥链的种子值从 1 维扩充到 n 维；接着，两方案都是无重复地选择这 n 个随机数中的一个来发布，经过 k 次之后就能恢复出新种子值。两方案都依赖于一个密钥链种子值服务器，服务器端将新的种子值通过分割、扩充维数等操作传输给客户端，客户端再用新的种子值来生成新的密钥链。从某种程度上实现了散列链可再生的概念，但是从散列链值使用者的角度来看，NSUHC 和 SUHC-EC 的构造方案与一般的 CHC (conventional hash chain)的构造并没有什么不同。

2010 年，文献[15]提出了基于公平交易(fair exchange idea)的自更新散列链(SRHC-FEI)的构造方案。该方案在每次发布散列链值的同时，利用 OTS 密钥来对新链链首值的一个比特进行加密传输。分析发现，SRHC-FEI 包含了认证、OTS 等元素，更像是一个应用而不是一个构造方案。且它虽然增强了一定的安全性及公平性，但是大大增加了开销，并且增加了系统的延时。

分析以上经典文献，不难发现，不管是 RHC、ERHC，还是 SUHC、SRHC、SRHC-FEI、都是对新链链首值的每一个比特做相应的变换或者映射成一个随机数，对新链链首值的安全发布即是对这些对应的随机数的安全发布。方案中要求所有的随

机数都必须完整地被接收才能正确地恢复出新链的链首值。显然，一定程度上减弱了系统的安全性，并且大大地增加了系统的开销。而在 NSUHC、SUHC-EC 中则是将新链链首值的维数进行适当的扩增，然后对扩充后的值进行相应的操作，虽然减少了系统的开销，但是仍然存在易受到中间人攻击等安全隐患。并且，从散列链值的使用者角度来看，只有 RHC、ERHC、SUHC、SRHC 才能真正算是可再生散列链的新颖的构造方案。

针对以上方案的一些不足之处，本文提出了一种改进的散列链再生方案——一种基于(t, n)门限和划分树的可再生散列链构造方案(SRHC-TD, novel self-renewal hash chain scheme based on (t, n) threshold and division tree)。SRHC-TD 将重复情况进行考虑提出了划分树和改进的划分树的概念，减少了系统的开销；并且将(t, n)门限、salt 值等概念相结合来保证散列链构造方法的安全性。

2 相关工作

2.1 基本定义

定义 1 (分裂) 将长度为 Lbit 的数分割成 m 个 2^l 进制数的过程称为分裂。简单表示为 $L=(m, l)$，$m=\left\lceil \frac{L}{l} \right\rceil$。如果 L 不是 2^l 的整数倍，则在 L 个数之前填充小于 2^l 个 0。

定义 2 (重复向量、取值向量、重复度)设 $m(m\geq 1)$个数的可能取值有 $n(n\geq 1)$种，其中，有 p_i $(1\leq p_i\leq m$，$\sum_{i=1}^{q} p_i=m)$个数取值为 v_i (v_i 取值各不相同)，$i=1, 2, \text{L}, q$，$1\leq q\leq \min(m,n)$，则将 $(p_1, p_2, \text{L}, p_q)$ 称为重复向量，$(v_1, v_2, \text{L}, v_q)$ 称为取值向量，$m-q$ 称为重复度，且 $m-q=\sum_i p_i-q=\sum_i (p_i-1)$。

定义 3 (重复率) 将重复度为 $m-q$ 时 m 个数的取值情况的个数记为 $S_{m,n}^{q}$，则当 q 取遍$[1, 2, \text{L}, \min(m, n)]$的所有值时，$m$ 个数的所有取值情况的个数记为 $\sum_{i=1}^{\min(m,n)} S_{m,n}^{i}$，前者与后者的比率称为重复度为 $m-q$ 时的重复率，简记为重复率，用 P_q 表示。$P_q=\frac{S_{m,n}^{q}}{\sum_{i=1}^{\min(m,n)} S_{m,n}^{i}}$，其中，$S_{m,n}^{q}=C_m^{p_1}\cdot C_{m-p_1}^{p_2}\text{L}\,C_{p_q}^{p_q}\cdot \frac{n!}{(n-q)!}$。

定义 4 (难度)设 $m(m\geq 1)$个数，重复度为 $m-q$，则将 q 称为难度。

定义 5 (平均难度)难度的平均加权和，用 D_q 表示。$D_q=\sum_{q=1}^{\min(m,n)} P_q q$。

2.2 (m, q)划分

定义 6 ((m, q)划分) 将正整数 m 无重复地划分成 q 个正整数相加的过程称为(m, q)划分。(m, q)划分对应一棵 m 划分树或者一棵改进的 m 划分树。m 划分树的树形结构如图 1 所示，改进的 m 划分树的树形结构如图 2 所示。

定义 7 (m 划分树) 将具有以下 4 个特点的树称为 m 划分树。

1) 根节点值为 m。

2) 值为 l 的节点有 l 个孩子，并且 $l\leq m$。

3) l 节点的第 i 个孩子的值为 $l-i$，i 叫做父亲节点到子节点的路径权值，并且 $i\leq l$。

4) 叶节点的值为 0。

要找到所有的(m, q)划分只需要根据 q 值直接遍历 m 划分树的第 q 层(根节点为第 0 层)的叶节点，再从叶节点回溯到根节点，如果回溯路径有重复(即至少存在 2 条回溯路径，它们的所有路径权值组合相等但排列不同)，则取其中一条回溯路径作为一个划分，这样，就找到了所有的非重复的(m, q)划分。

不妨以$(m, 2)$划分为例。如图 1 所示，直接遍历第 2 层的 $m-1$ 个叶节点，即找到了 $m-1$ 条有重复的回溯路径，分别是$(1, m-1)$, $(2, m-2)$, L, $(m-2, 2)$, $(m-1, 1)$。再进行比较可知$(i, m-i)$与$(m-i, i)$(其中，$i=1, 2, \text{L}, m-1$)是重复路径，去掉重复路径，得到$\left\lfloor \frac{m}{2} \right\rfloor$条无重复的回溯路径，即找到了$\left\lfloor \frac{m}{2} \right\rfloor$个$(m, 2)$划分。

由于 m 划分树中存在重复划分的情况，需要回溯来删除重复的划分。不妨直接将路径权值作为考虑因素来进行构造改进的 m 划分树。

定义 8 (改进的 m 划分树) 将具有以下 4 个特点的树称为改进的 m 划分树。

1) 根节点值为 m，且有$\left\lfloor \frac{m}{2} \right\rfloor+1$个孩子。

2) 若一个中间节点的值为 l $(0<l<m)$，且其父亲节点到它的路径权值为 r，当 $l\geq 2r$ 时，该节点有$\left\lfloor \frac{l}{2} \right\rfloor-r+2$ 个孩子，当 $r\leq l<2r$ 时，该节点有一个孩子(叶节点)。

3) 父亲节点到子节点的路径权值范围是 $\begin{cases}\left\{r, r+1, \cdots, \left\lfloor \frac{l}{2} \right\rfloor\right\} \cup \{l\}, l \geq 2r \\ \{l\}, r \leq l \leq 2r\end{cases}$。

4) 叶节点的值为 0。

这样再根据 q 值直接遍历第 q 层(根节点为第 0 层)的叶节点，由于避免了重复，叶节点的个数即是划分的个数。这样，就找到了所有的非重复的(m, q)划分。

不妨再以$(m, 2)$划分为例。如图 2 所示，直接遍历第 2 层的$\left\lfloor \frac{m}{2} \right\rfloor$个叶节点，找到了$\left\lfloor \frac{m}{2} \right\rfloor$条无重复的回溯路径，分别是 $(1, m-1)$, $(2, m-2)$, …, $(\left\lfloor \frac{m}{2} \right\rfloor, \left\lceil \frac{m}{2} \right\rceil)$，即找到了$\left\lfloor \frac{m}{2} \right\rfloor$个$(m, 2)$划分。

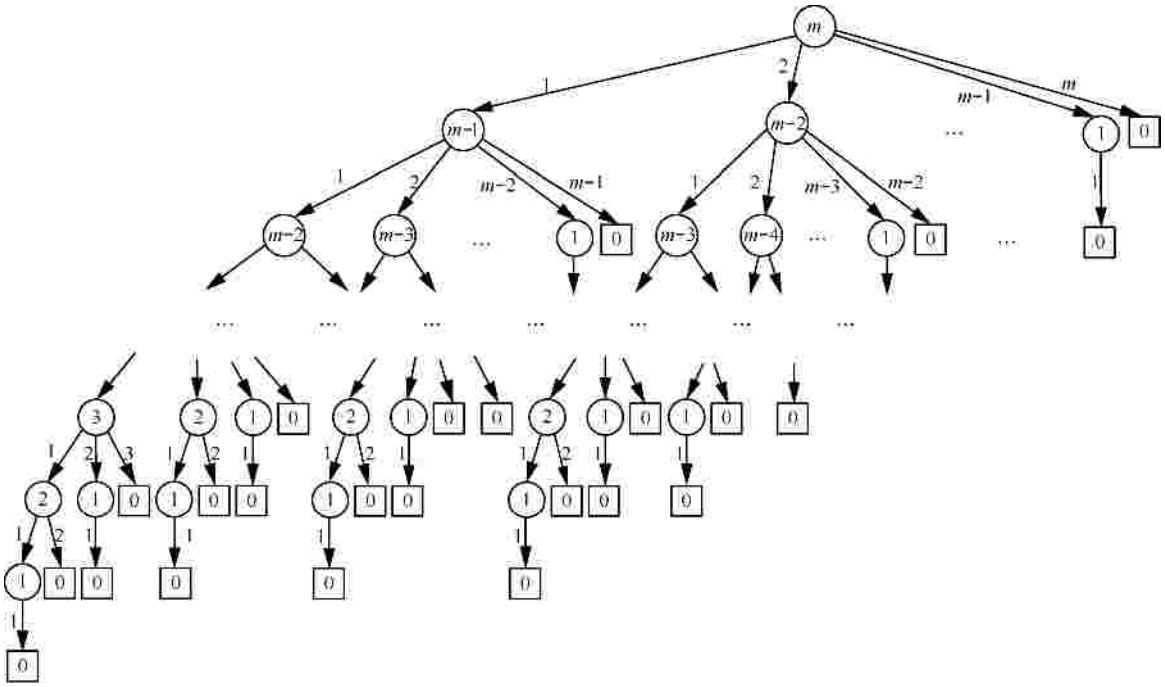

图 1　m 划分树

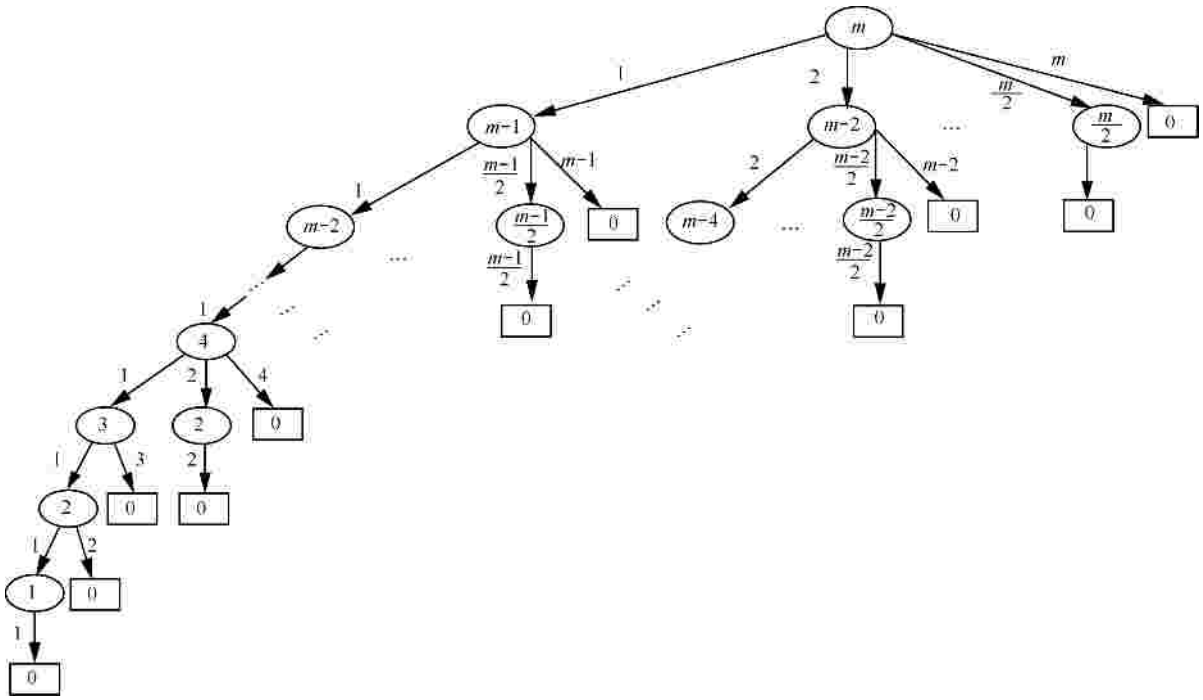

图 2　改进的 m 划分树

2.3 (*t*, *n*)-Mignotte's 门限秘密共享方案

定理 1 (中国剩余定理)对于所有的 $1 \le i, j \le k$，当 $(m_i, m_j) = 1$，得到中国剩余定理的标准形式。

设 $m_1, m_2, \perp, m_k$ 是两两互质的 k 个正整数，$k \ge 2$，则同余方程组

$$\begin{cases} X \equiv b_1 \bmod m_1 \\ X \equiv b_2 \bmod m_2 \\ \quad \vdots \\ X \equiv b_k \bmod m_k \end{cases}$$

有模 $M = m_1 m_2 \perp m_k$ 的唯一解 $X = \sum_{i=1}^{k} b_i M_i (M_i^{-1} \bmod m_i) \bmod M$，其中，$M_i = \dfrac{M}{m_i}$，$1 \le i \le k$。

定义 9 ((*t*, *n*)-Mignotte's 序列) 设 n 是一个整数，$n \ge 2$，如果给定正整数序列 $m_1 < m_2 < \perp < m_n$，$(m_i, m_j) = 1$，$1 \le i < j \le n$，满足 $m_{n-t+2} m_{n-t+3} \perp m_n < m_1 m_2 \perp m_t$ 条件，就称该正整数序列是一个(*t*, *n*)-Mignotte's 序列。

如果给定一个(*t*, *n*)-Mignotte's 序列，秘密共享方案运行如下[16]。

1) 随机选择一个整数 S 作为秘密，且要满足：$m_{n-t+2} m_{n-t+3} \perp m_n < S < m_1 m_2 \perp m_t$ 条件，如果设 $a = m_1 m_2 \perp m_t$，$b = m_{n-t+2} m_{n-t+3} \perp m_n$，即满足 $b < S < a$。

2) 根据 $I_i = S \bmod m_i$，可以得到秘密份额 I_i，$1 \le i \le n$。

3) 任意选定 t 个不同的秘密份额，分别记为 $I_{i_1}, I_{i_2}, \perp, I_{i_t}$，由中国剩余定理可以恢复出秘密 S，且在模 $m_1 \cdot m_2 \perp m_t$ 下是唯一的。

$$\begin{cases} X \equiv I_{i_1} \bmod m_{i_1} \\ X \equiv I_{i_2} \bmod m_{i_2} \\ \quad \vdots \\ X \equiv I_{i_t} \bmod m_{i_t} \end{cases}$$

3 方案描述

设单向散列函数 h 的输出的长度为 Lbit(例如，MD5 算法的输出是 $L = 128$ bit)，并且发送接收双方协定将 L 分裂成 (m_L, l_L)。下面分为 4 个阶段进行描述。

3.1 密钥初始化阶段

Step1 发送方选择一个合适的 (t, m_L) -Mignotte's 序列 $\{x_1, x_2, \perp, x_{m_L}\}$，将它们的级联记作 S_U，并且分别对序列中的每个正整数进行散列运算，计算出相应的 m_L 个散列值，将这些散列值的级联记作 P_U。

Step2 发送方以 P_U 为初始种子值，生成一条长度为 N 的密钥链

$$P_U, h(P_U), h^2(P_U), \perp, h^i(P_U), \perp, h^{N-1}(P_U), h^N(P_U)$$

其中，$N-1$ 为 m_L 的整数倍。

Step3 发送方按照 Step1 的方法重新选择一个合适的(t', m_L)-Mignotte's 序列，再生成一对新的密钥实例 S_U'和 P_U'；并且以 P_U'为初始种子值，生成一条长度为 N 的密钥链

$$P_U', h(P_U'), h^2(P_U'), \perp, h^i(P_U'), \perp, h^{N-1}(P_U'), h^N(P_U')$$

将 $h^N(P_U')$分割成 m_L 个 2^{l_L} 进制数 $c_1, c_2, \perp, c_{m_L}$，这 m_L 个 2^{l_L} 进制数的重复度记为 $m_L - q_L$，难度记为 q_L。

Step4 如果满足条件 $b < h^N(P_U') < a$ 则以 $h^N(P_U')$作为秘密（主密钥）S，并且跳转到 Step5，否则跳转到 Step2；其中，$a = x_1 x_2 \perp x_t$，$b = x_{m_L - t + 2} x_{m_L - t + 3} \perp x_{m_L}$，$t = q_L$。

Step5 根据 $I_{i_m} = S \bmod x_{i_m}$，可以得到 m_L 个秘密份额（子密钥）I_{i_m}，$1 \le i_m \le m_L$。

3.2 密钥发送阶段

Step1 对于 2^{l_L} 进制数 c_{i_m}，找到对应的(*t*, m_L)- Mignotte's 序列值 $x_{c_{i_m}+1}$，及对应的子密钥值 $I_{c_{i_m}+1}$，其中，$1 \le i_m \le m_L$。

在公布第一条密钥链的第 i 个密钥值 $h^{N-i}(P_U)$ 之前，先公布联合散列值 $HUT_i = h(h^{N-i}(P_U) \| ((x_{c_{k_m}+1} \| I_{c_{k_m}+1}) << r_i))$，其中，$1 \le i \le N-1$，$k_m = (i-1) \bmod m_L + 1$，$<<$ 表示左移运算符，r_i 表示 P_U 值的第 0 比特、第 i 比特、第 $2i$ 比特、…组成的数。

Step2 等到接收者收到 HUT_i 后，再公布密钥值 $h^{N-i}(P_U)$ 以及 $((x_{c_{k_m}+1} \| I_{c_{k_m}+1}) << r_i)$。

Step3 发送方最后公布第一条密钥链的种子值 P_U。

3.3 密钥认证阶段

Step1 接收者收到 HUT_i 时，先将其存储下来。

Step2 经过一个时间周期后，接收者收到密钥公布数据分组 $\{h^{N-i}(P_U), ((x_{c_{k_m}+1} \| I_{c_{k_m}+1}) << r_i)\}$，如果超过时间阈值，则丢弃该密钥公布数据分组和存储的

HUT_i，等待下一次的 HUT_i 和密钥公布数据分组。

Step3　如果在时间阈值内，首先计算 $h(h^{N-i}(P_U))$ 和 $h(h^{N-i}(P_U) \| ((x_{c_{k_m+1}} \| I_{c_{k_m+1}}) << r_i))$，如果分别等于已经认证的密钥值 $h^{N-i+1}(P_U)$ 和接收到的 HUT_i，表示认证成功，跳转到 Step5；否则认证失败，结束。

Step4　如果超过了时间阈值，在下一次（时间阈值内）收到密钥公布数据分组 $\{h^{N-i'}(P_U),\ ((x_{c'_{k_m+1}} \| I_{c'_{k_m+1}}) << r'_i)\}$ 时，$i' \geq i$，首先计算 $h^{i'-i+1}(h^{N-i'}(P_U))$ 和 $h(h^{N-i'}(P_U) \| ((x_{c_{k_m+1}} \| I_{c_{k_m+1}}) << r'_i))$，如果分别等于已经认证的密钥值 $h^{N-i+1}(P_U)$ 和接收到的 HUT_i'，则表示密钥公布数据分组中的 $h^{N-i'}(P_U)$ 以及 $((x_{c'_{k_m+1}} \| I_{c'_{k_m+1}}) << r'_i)$ 是正确的，并通过计算 $h^{i''-i'}(h^{N-i'}(P_U))$ 可以得到之前的密钥 $h^{N-i''}(P_U)$，$i'' \leq i'$，跳转到 Step5；否则认证失败，结束。

Step5　接收方最终得到至多 m_L 个 $((x_{c_{k_m+1}} \| I_{c_{k_m+1}}) << r_i)$ 以及初始种子值 P_U，P_U 的认证过程与 Step3 和 Step4 过程相类似，如果 P_U 认证成功则跳转到 Step6，否则认证失败，结束。

Step6　对于满足条件 $i_1 - i_2 \equiv 0 \bmod q_L$ 所对应的 $((x_{c_{k_{m_1}+1}} \| I_{c_{k_{m_1}+1}}) << r_{i_1})$ 和 $((x_{c_{k_{m_2}+1}} \| I_{c_{k_{m_2}+1}}) << r_{i_2})$ 应满足 $x_{c_{k_{m_1}+1}} = x_{c_{k_{m_2}+1}}$ 并且 $I_{c_{k_{m_1}+1}} = I_{c_{k_{m_2}+1}}$，验证成功则跳转到密钥重组阶段，否则认证失败，结束。

Step7　接收方通过移位运算能够得到 q_L 个 $x_{c_{k_m+1}}$ 和 $I_{c_{k_m+1}}$。

3.4　密钥重组阶段

密钥认证成功后会得到 q_L 个 $x_{c_{k_m+1}}$ 和 $I_{c_{k_m+1}}$ 以及 q_L 个 $x_{c_{k_m+1}}$ 分别所在的位置关系，即得到一个包含 m_L 个元素的有序序列 $\{x_{d_1}, x_{d_2}, \text{L}, x_{d_{m_L}}\}$，其中，$x_{d_r} = x_{c_{k_m+1}}$，$1 \leq r \leq m_L$，$k_m = (i-1) \bmod m_L + 1$，$1 \leq i \leq N-1$（实际上，序列 $\{x_{d_1}, x_{d_2}, \text{L}, x_{d_{m_L}}\}$ 是一个 (m_L, q_L) 划分）。

有序验证法：对于有序的序列 $\{x_{d_1}, x_{d_2}, \text{L}, x_{d_{m_L}}\}$，其所有元素的下标值 d_r 的合并对应着新生的第二条密钥链的链首，暂记为 $h^N(P_U')_1$，即 $h^N(P_U')_1 = d_1 d_2 \text{L}\, d_{m_L}$。

中国剩余定理验证法：q_L（t=q_L）对 $\{x_{c_k \bmod 2^{l_L}}, I_{c_k \bmod 2^{l_L}}\}$ 可得到同余方程组

$$\begin{cases} x \equiv I_{c_1 \bmod 2^{l_L}} \bmod x_{c_1 \bmod 2^{l_L}} \\ x \equiv I_{c_2 \bmod 2^{l_L}} \bmod x_{c_2 \bmod 2^{l_L}} \\ \qquad\qquad \text{M} \\ x \equiv I_{c_{q_L} \bmod 2^{l_L}} \bmod x_{c_{q_L} \bmod 2^{l_L}} \end{cases}$$

利用中国剩余定理，可以得到唯一解 S，即新生的第二条密钥链的链首，暂记为 $h^N(P_U')_2$。

1) 如果 $h^N(P_U')_1 = h^N(P_U')_2$，则表示重组成功。

2) 如果 $x_{c_{k_m+1}}$ 和 $I_{c_{k_m+1}}$ 在传输过程中有少量缺失，则可通过中国剩余定理先得到一个解 $h^N(P_U')_2$，然后对比未缺失的 $x_{c_{k_m+1}}$ 值与 $h^N(P_U')_2$ 的位置关系，如果一一对应则重组成功。

3) 否则表示重组失败，那么认证也失败。

4　安全性及复杂性分析

4.1　安全性

SRHC-TD 的构造方案的安全性主要体现在以下 2 个方面。

1) 考虑重复情况的明文空间比较

散列链有一个特点：链上的每一个值（除了初始种子值以及已公布的链首值）既是散列计算的输出（cybertext）也是输入（plaintext）。所以，在不考虑初始种子值的情况下，散列链上的每一个值长度相同。这样带来的一个坏处是会减少明文空间（key space），从而减少破解的时间。

而 SRHC-TD 在散列链中增加了 salt 值，即 $((x_{c_{k_m+1}} \| I_{c_{k_m+1}}) << r_i)$，从而增大了 key space，这恰恰是其他的构造方案——RHC、ERHC、SUHC、SRHC 中所没有提及到的。

以 MD5 为例，假设输出 *L*bit(128bit)分裂成(32, 4)，链长 $N = 97$。选择 1 000 组(*t*, 32)-Mignotte's 序列（$t = 10, 11, \text{L}, 16$），并且假设序列值的平均长度为 4bit，每组分别计算 MD5 散列级联值 P_U，再将 P_U 进行 97 次 MD5 散列，记录下这 97 000 个 cybertext 中的(32, *q*)划分情况。并且按照每 9 700 个 cybertext 为一组 test，每组 test 中计算 *q* 取到不同数值时的概率。将第 *i* 组 test 中 *q* 值取 *j* 时的概率记为 p_{ij}。得到如图 3 的统计概率。

由图 3 可知，*q* 的取值范围大概是{10,11,12,13, 14,15,16}，当然不排除会取值到不大于 10 的自然数的情况存在。图 3 中具体的数值如表 1 所示。

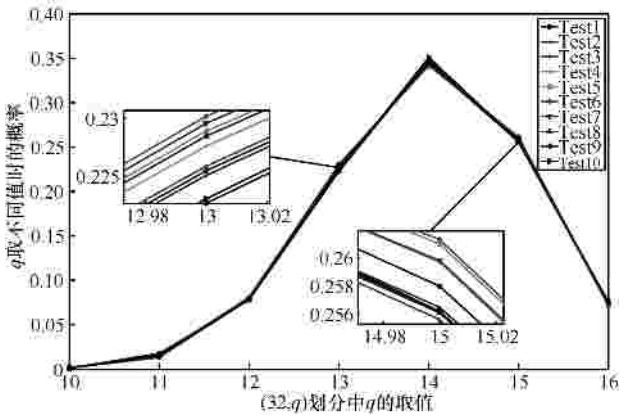

图 3 (32, q)划分中 q 取不同值的概率统计

由 SRHC-TD 中的 salt 值可知，明文空间平均增加了 128bit（128=4×32），这样，明文空间的大小就变成了 2^{256}。基于改进的 m 划分树的概念，根据表 1 中 q 取不同值时的平均概率可以得出一般 RHC[7~13](没有增加 salt 值)的平均 key space 以及 SRHC-TD(加了 salt 值)的平均 key space，如表 2 所示。

由表 2 可知，SRHC-TD 中增加了 salt 值，key space 比 RHC、ERHC、SUHC、SRHC 增长了 161 995 868 252 801 528 012 638 215 965 348 282 400 倍，这样不管是在暴力破解或者彩虹表破解时都增加了破解时间，从而增加了散列链的安全性，性能更加优于文献[7~13]中没有 salt 值 RHC 构造方案。

2) 双重认证

第 3 节的密钥重组阶段中，接收方根据已知(已认证）的信息用 2 种不同的方法来计算得到新生的第二条密钥链的链首值 $h^N(P_U')$。

① 其中，有序验证法是在 ERHC 方案的基础上改进得到的。与 ERHC 类似，它同样也具有不可否认性的特点。

在 ERHC 中，假设散列函数输出值长 L=8bit 时，用 $\lfloor \mathrm{lb}L \rfloor+1=4$ bit 来表示私钥中“1”的个数。当第二条密钥链的链首值 $h^N(P_U')$ 中“1”的个数为 5 时，校验位为 0101，中间人可以伪造成“1”的个数为 0、1、4，校验位改为 0000 或 0001 或 0100，不透露或只透露 S_U 的部分信息。而出现这种情况的概率很大，因为只要 $h^N(P_U')$ 中“1”的个数不为 0，就都会有可能受到此类的选择明文攻击。

而 SRHC-TD 则有效地防止了此类攻击。这是因为，SRHC-TD 中每次传输的部分 S_U 值 $x_{c_k \bmod 2^{l_L}}$ 和 Mignotte's 序列值 $I_{c_k \bmod 2^{l_L}}$ 都是左移过 r_i 位的，而每次的 r_i 值各不相同且又与 P_U 值有关，在 P_U 尚未公布之前，所有获得的 S_U 值和序列值都是密文。这样，散列链上的每个散列值都不相同，salt 值也都不相同。因此，将“左移操作”和“r_i 值的选择”这 2 个操作结合起来，能够更加有效地防止选择明文攻击。

② 而中国剩余定理验证法是基于 (t, n)-Mignotte's 门限秘密共享方案的。它虽然弱化了“任意 t 个子密钥都能计算得出唯一的主密钥”[17]这一属性。但却能由散列链的单向性和延迟发布性（先发布联合散列值再发布联合值）这 2 个特性来保证 t 个子密钥的完整性；并且通过延迟发布 P_U 值来进一步保证 t 个子密钥的完整性和正确性。从而，保证主密钥的唯一性和正确性。

表 1 (32, q)划分中 q 取不同值的概率统计

q	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	$p_{\mathrm{Avg}_{q=j}}=\frac{1}{10}\times\sum_{i=1}^{10}p_{ij}$
10	0.001 3	0.001 3	0.001 4	0.001 2	0.001 2	0.001 2	0.001 6	0.001 6	0.001 3	0.001 3	0.001 3
11	0.017 6	0.016 8	0.015 9	0.014 2	0.013 3	0.013 5	0.013 7	0.015 3	0.015 3	0.015 5	0.015 1
12	0.078 1	0.080 6	0.081 5	0.082 1	0.080 7	0.079 3	0.078 4	0.078 5	0.079 3	0.078 3	0.079 7
13	0.222 7	0.225 1	0.225 9	0.227 6	0.228 9	0.230 2	0.229 6	0.228 5	0.225 5	0.223 1	0.226 7
14	0.348 0	0.344 8	0.346 6	0.344 5	0.343 8	0.341 8	0.343 5	0.344 0	0.346 4	0.350 6	0.345 4
15	0.258 0	0.256 5	0.255 6	0.259 7	0.261 0	0.261 3	0.259 8	0.256 1	0.256 2	0.256 2	0.258 0
16	0.074 3	0.074 9	0.073 1	0.070 7	0.071 1	0.072 7	0.073 4	0.076 0	0.076 0	0.075 0	0.073 7

表 2　一般 RHC 的 key space 与 SRHC-TD 的 key space 对比

（RHC、ERHC、SUHC、SRHC 没有 salt 值的 key space 记为：KSu, SRHC-TD 有 salt 值的 key space 记为：*KSa*）

q		key space	倍数
10	KSu_{10}	553 333 286 463 413 155 119 380 296 207 872 000	143 020 679 939 872 390 880 211 676 554 285
	KSa_{10}	79 138 102 863 361 537 004 679 285 453 617 238 668 120 868 942 959 692 426 007 139 840 000	
11	KSu_{11}	5 187 080 514 550 099 279 678 121 388 076 646 400	3 661 891 040 325 389 282 674 949 544 144 523
	KSa_{11}	18 994 523 661 677 418 591 170 431 781 409 944 281 563 099 229 921 542 416 319 529 347 072 000	
12	KSu_{12}	26 806 046 942 907 604 997 886 958 590 845 952 000	75 742 751 459 000 734 383 329 857 043 540 793
	KSa_{12}	2 030 363 751 194 957 174 069 912 361 433 154 199 951 665 857 317 337 853 414 530 889 487 360 000	
13	KSu_{13}	76 637 826 616 149 945 551 514 260 280 729 600 000	1 324 110 606 676 344 898 872 293 485 267 251 454
	KSa_{13}	101 476 959 095 066 836 883 574 848 357 011 830 828 587 992 686 398 439 770 172 405 579 571 200 000	
14	KSu_{14}	117 608 245 430 622 401 776 527 064 367 308 800 000	20 264 471 610 378 457 664 64 760 637 922 372 791
	KSa_{14}	2 383 268 950 675 269 627 341 439 620 767 777 205 252 014 589 252 078 877 364 719 119 523 225 600 000	
15	KSu_{15}	88 465 874 655 133 624 601 621 999 967 436 800 000	279 240 713 721 406 725 647 529 271 034 190 240 345
	KSa_{15}	24 703 273 978 688 019 410 535 964 115 469 307 361 564 474 752 449 796 080 640 681 676 767 395 840 000	
16	KSu_{16}	24 991 467 359 322 430 689 958 992 833 556 480 000	3 545 290 824 160 424 396 558 195 520 471 044 725 362
	KSa_{16}	88 602 019 911 310 565 452 710 747 271 750 192 445 143 064 404 384 392 174 672 129 629 351 424 000 000	
$KSu_{\text{Avg}} = \sum_{q=10}^{16}\left(p_{\text{Avg}_q}\cdot KSu_q\right)$		84 877 236 261 416 553 573 765 570 028 577 116 785	161 995 868 252 801 528 012 638 215 965 348 282 400
$KSa_{\text{Avg}} = \sum_{q=10}^{16}\left(p_{\text{Avg}_q}\cdot KSa_q\right)$		13 749 761 583 066 344 526 126 464 575 649 604 504 003 162 279 445 035 355 884 359 119 207 993 553 972	

同时，由于 SRHC-TD 中将散列链的长度 $N-1$ 定义为 m_L(对应(*t*, *n*)门限中的 *n* 值)的倍数，这样，使得 *t* 个子密钥（实际上是 *t* 个子密钥变形后生成的 *n* 个子密钥）在整条散列链的使用过程中重复传输，从而避免出现由于某一个子密钥丢失而造成主密钥无法求解的情况。说明新方案符合并且严格遵循(*t*, *n*)门限的 *t*-consistency[17]这一属性。

以上的属性都是 RHC、ERHC、SUHC、SRHC 中所没有的。

3) 可证明安全

由于 SRHC-TD 是由前后多条链的首尾相接组成的。所以，在研究其可证明安全性时，需要分别考虑 2 种情况。① 在一条链上的前后散列值的可证明安全性；② 前后 2 条链的首尾相接处的散列值的可证明安全性。

为简便起见，选择 Random Oracle 模型来分析 SRHC-TD 的理论模型。在该模型中，攻击者拥有关于某个安全参数 *k* 的多项式计算能力。方案中所使用的所有算法的安全强度均由安全参数 *k* 决定，并且破解散列算法的概率为关于参数 *k* 的指数函数的倒数，破解中国剩余定理算法的概率为关于参数 *k* 的幂函数的倒数，破解划分算法的概率为关于参数 *k* 的对数函数的倒数。

①在第 *i* 个时间周期结束时刻，接收者（包含

攻击者）收到 HUT_{i+1}，以及 $\{h^{N-i}(P_{\mathrm{U}}),\ ((x_{c_{k_m}+1} \| I_{c_{k_m}+1}) << r_i)\}$。攻击者需要在第 $i+1$ 个时间周期内成功伪造 $h_{\mathrm{fake}}^{N-i-1}$ 和 $redu_{\mathrm{fake}}^{N-i-1}$，使得 $h\left(h_{\mathrm{fake}}^{N-i-1}\right)=h^{N-i}(P_{\mathrm{U}})$ 以及 $h(h_{\mathrm{fake}}^{N-i-1} \| redu_{\mathrm{fake}}^{N-i-1})=HUT_{i+1}$ 才能攻击成功。设攻击者的计算能力界限是多项式 $T_{\mathrm{adv}}(k)$，在每个时间段内可查询 Oracle 任意次，又假设破解 2 个散列算法的概率均为 e^{-k}，则攻击成功的概率为

$$\Pr[Adv(k)=1]=\Pr[h_{\mathrm{fake}}^{N-i-1}]=h^{N-i-1}(P_U),$$
$$h(h_{\mathrm{fake}}^{N-i-1} \| redu_{\mathrm{fake}}^{N-i-1})=HUT_{i+1}]=\frac{T_{\mathrm{adv}}(k)}{\mathrm{e}^k \mathrm{e}^k}$$

$T_{\mathrm{adv}}(k)$ 具有以下一般形式，$T_{\mathrm{adv}}(k)=a_n k^n+a_{n-1}k^{n-1}+\mathrm{L}+a_1 k+a_0$。于是，有 $\Pr[Adv(k)=1]=\frac{a_n k^n+a_{n-1}k^{n-1}+\mathrm{L}+a_1 k+a_0}{\mathrm{e}^{2k}}$。对 $\Pr[Adv(k)=1]$ 取极限得到

$$\lim_{k\to\infty}\Pr[Adv(k)=1]=\lim_{k\to\infty}\frac{a_n k^n}{\mathrm{e}^{2k}}=\lim_{k\to\infty}\frac{a_n n k^{n-1}}{2\mathrm{e}^{2k}}$$
$$=\mathrm{L}=\lim_{k\to\infty}\frac{a_n n!}{2^n \mathrm{e}^{2k}}=0$$

②在第 $N-1$ 个时间周期结束时刻，第一条链中仅剩 P_{U} 值没有公布。接收者（包含攻击者）得到 m_L 个 $((x_{c_{k_m}+1} \| I_{c_{k_m}+1}) << r_i)$ 值。攻击者需要在第 N 个时间周期内成功伪造 P_{U} 值 $P_{\mathrm{U}_{\mathrm{fake}}}$，使得 $h(P_{\mathrm{U}_{\mathrm{fake}}})=h(P_{\mathrm{U}})$；且当 $i_1-i_2\equiv 0 \bmod q_L$ 时，$((x_{c_{k_{m_1}}+1} \| I_{c_{k_{m_1}}+1}) << r_{i_1})$ 和 $((x_{c_{k_{m_2}}+1} \| I_{c_{k_{m_2}}+1}) << r_{i_2})$ 中 $x_{c_{k_{m_1}}+1}=x_{c_{k_{m_2}}+1}$，$I_{c_{k_{m_1}}+1}=I_{c_{k_{m_2}}+1}$；且最后用有序验证法得到的解 $h^N(P_{\mathrm{U}}')_1$ 要和中国剩余定理验证法得到的解 $h^N(P_{\mathrm{U}}')_2$ 相等。攻击者只有伪造了满足以上 3 个条件的 $P_{\mathrm{U}_{\mathrm{fake}}}$ 值才能攻击成功。设攻击者的计算能力界限是多项式 $T_{\mathrm{adv}}(k)$，在每个时间段内可查询 Oracle 任意次，又假设破解散列算法的概率为 e^{-k}，破解中国剩余定理算法的概率为 $k^{-\mathrm{e}}$，破解划分算法的概率为 $\frac{1}{\ln k}$，则攻击成功的概率为

$$\Pr[Adv(k)=1]=\Pr[h(P_{\mathrm{U}_{\mathrm{fake}}})=h(P_{\mathrm{U}}),(x_{c_{k_{m_1}}+1}=x_{c_{k_{m_2}}+1},$$
$$I_{c_{k_{m_1}}+1}=I_{c_{k_{m_2}}+1}),h^N(P_{\mathrm{U}}')_1=h^N(P_{\mathrm{U}}')_2]=\frac{T_{\mathrm{adv}}(k)}{\mathrm{e}^k k^{\mathrm{e}} \ln k}$$

取 $T_{\mathrm{adv}}(k)=a_n k^n+a_{n-1}k^{n-1}+\mathrm{L}+a_1 k+a_0$，对 $\Pr[Adv(k)=1]$ 取极限得到

$$\lim_{k\to\infty}\Pr[Adv(k)=1]$$
$$=\lim_{k\to\infty}\frac{a_n k^n}{\mathrm{e}^k k^{\mathrm{e}} \ln k}=\lim_{k\to\infty}\frac{a_n k^{n-\mathrm{e}}}{\mathrm{e}^k \ln k}=\lim_{k\to\infty}\frac{a_n (n-\mathrm{e})k^{n-\mathrm{e}-1}}{\mathrm{e}^k\left(\ln k+\frac{1}{k}\right)}=\mathrm{L}$$
$$=\lim_{k\to\infty}\frac{a_n (n-\mathrm{e})!}{\mathrm{e}^k\left(\ln k+\frac{n-\mathrm{e}}{k}-\mathrm{L}+\frac{(n-\mathrm{e}-1)!}{k^{n-\mathrm{e}}}\right)}=\lim_{k\to\infty}\frac{a_n (n-\mathrm{e})!}{\mathrm{e}^k \ln k}=0$$

综合以上 2 种情况，且由极限的定义可知，存在正整数 N，当 $k>N$ 时，对于任意的 $e>0$，有 $\Pr[Adv(k)=1]<e$。所以攻击者攻击成功的概率可忽略不计，即本文所提出的 SRHC-TD 方案是可证明安全的。

4.2 复杂性

散列链的保存方法有 2 种：1) 生成一条链，然后分配专门的空间保存整条链值；2)只保存种子值，每次用到某个值时再通过种子值来计算。前者降低了计算开销但是增加了存储开销，后者降低了存储开销但是增加了计算开销。

以下将 SRHC-TD 与 RHC、ERHC、SUHC、SRHC 进行对比，分别考虑在散列链的 2 种保存方法下各方案的性能比较，如表 3 所示。

其中，L 表示散列函数的输出长度，例如 MD5 的 L 值等于 128；n 表示散列链的长度；m 表示 SRHC-TD 中 Mignotte's 序列值的个数。

H 表示散列函数的计算开销；U 表示级联操作的计算开销；R、R_B、$R_{B'}$、$R_{t,m}$ 分别表示 RHC 和 ERHC、SUHC、SRHC、SRHC-TD 中生成一个随机数的计算开销；B、B' 分别表示 SUHC，SRHC 中从随机数重构 1bit 的计算开销；I、P_r 分别表示 SRHC-TD 中计算子密钥和移位数 r 的计算开销；M 表示移位操作的计算开销；C 表示计算一个同余方程组的计算开销。

len_H 表示 $k$bit 的存储/通信开销，例如 MD5 中 $len_H=16$ byte；len_s 表示散列链种子值的存储/通信开销；len_r 表示生成的随机数的存储/通信开销；$len_{r'}$ 表示 Mignotte's 序列值的存储/通信开销；len_I 表示 SRHC-TD 中子密钥的存储/通信开销。

为简便起见，不妨假设 $L\approx n$，$m\le\frac{n}{4}$，R；R_{B}；$R_{\mathrm{B}'}$；$R_{t,m}$，B；B'，$H>R$，$H>B$，$H>C>I>P_r$；M，len_H；len_S；len_r；$len_{r'}$；len_I。对表 3 的开销数据进行简化，并对以上 5 个方案的开销进行定性的比较，如表 4 所示。

表 3　RHC、ERHC、SUHC、SRHC、SRHC-TD 复杂性比较(1)

方案	散列链保存方法		保存整条链	只保存种子值
RHC	初始化	计算	$(2L+3)H+2R$	$(3L+2)H+2R$
		通信	$2len_H$	$2len_H$
		存储	$2(len_s+len_r)+(L+4)len_H$	$2(len_s+len_r)+4len_H$
	发送—认证—重组	计算	$(2L+3)H+2R$	$\frac{1}{2}(L^2+3L-4)H+2(L-1)R$
		通信	$2Llen_r+(6L-2)len_H$	$2Llen_r+(6L-2)len_H$
		存储	$2(len_s+len_r)+(L+4)len_H$	$len_r+(L+2)len_H+L$
ERHC	初始化	计算	$2(n+L+\lfloor \mathrm{lb}L \rfloor+1)H+2(L+\lfloor \mathrm{lb}L \rfloor+1)R+2U$	$2(n+L+\lfloor \mathrm{lb}L \rfloor+1)H+2(L+\lfloor \mathrm{lb}L \rfloor+1)R+2U$
		通信	0	0
		存储	$(n+2L+1)len_H+2Llen_r$	$(2L+1)len_H+2Llen_r$
	发送—认证—重组	计算	$\frac{1}{2}(2n+L+\lfloor \mathrm{lb}L \rfloor+1)H$	$\frac{1}{2}(n^2+n+L+\lfloor \mathrm{lb}L \rfloor+1)H$
		通信	$2(n+L+\lfloor \mathrm{lb}L \rfloor+1)len_H+(L+\lfloor \mathrm{lb}L \rfloor+1)len_r$	$2(n+L+\lfloor \mathrm{lb}L \rfloor+1)len_H+(L+\lfloor \mathrm{lb}L \rfloor+1)len_r$
		存储	$(n+L+\lfloor \mathrm{lb}L \rfloor+1)len_H+L$	$(n+L+\lfloor \mathrm{lb}L \rfloor+1)len_H+L$
SUHC	初始化	计算	$2(L+1)H+R_B$	$(3L+1)H+R_B$
		通信	$2len_H$	$2len_H$
		存储	$2len_s+len_r+(L+3)len_H$	$2len_s+len_r+4len_H$
	发送—认证—重组	计算	$(5L-2)H+(L-1)R_B+L_B$	$\frac{1}{2}(L^2+6L-4)H+(L-1)R_B+L_B$
		通信	$(6L-3)len_H+2Llen_r$	$(6L-3)len_H+2Llen_r$
		存储	$(L+2)len_H+len_r+L$	$(L+2)len_H+len_r+L$
SRHC	初始化	计算	$(2L+1)H+R_{B'}$	$3LH+R_{B'}$
		通信	$2len_H$	$2len_H$
		存储	$2len_s+len_r+(L+2)len_H$	$2len_s+len_r+3len_H$
	发送—认证—重组	计算	$(3L-1)H+(L-1)R_{B'}+L_{B'}$	$\frac{1}{2}(L^2+5L-2)H+(L-1)R_{B'}+L_{B'}$
		通信	$(4L-2)len_H+2Llen_r$	$(4L-2)len_H+2Llen_r$
		存储	$Llen_H+2len_r+L$	$Llen_H+2len_r+L$
SRHC-TD	初始化	计算	$2(n+m)H+2mR_{t,m}+2U+mI$	$2(n+m)H+2mR_{t,m}+2U+m_I$
		通信	0	0
		存储	$(n+2m+1)len_H+2mlen_{r'}+mlen_{\mathrm{I}}$	$(2m+1)len_H+2mlen_{r'}+mlen_I$
	发送—认证—重组	计算	$3nH+2nM+2nP_r+C$	$\frac{1}{2}(n^2+5n)H+2nM+2nP_r+C$
		通信	$4nlen_H+2nlen_{r'}+2nlen_{\mathrm{I}}$	$4nlen_H+2nlen_{r'}+2nlen_I$
		存储	$(n+2m)len_H+nlen_{r'}+nlen_I+L$	$(n+2m)len_H+nlen_{r'}+nlen_I+L$

表 4 RHC、ERHC、SUHC、SRHC、SRHC-TD 复杂性比较(2)

散列链保存方法		保存整条链	只保存种子值
初始化	计算	SRHC < SUHC < RHC < SRHC-TD < ERHC	SRHC < SUHC < RHC < SRHC-TD < ERHC
	通信	ERHC = SRHC-TD < RHC = SRHC = SUHC	ERHC= SRHC-TD < RHC = SRHC = SUHC
	存储	SRHC < SUHC < RHC < SRHC-TD < ERHC	SRHC < SUHC < RHC < SRHC-TD < ERHC
发送—认证—重组	计算	ERHC < RHC < SRHC < SRHC-TD < SUHC	ERHC < RHC < SRHC < SRHC-TD < SUHC
	通信	SRHC-TD < SRHC < ERHC < RHC < SUHC	SRHC-TD < SRHC < ERHC < RHC < SUHC
	存储	SRHC < SUHC < RHC < SRHC-TD < ERHC	SRHC < SUHC = RHC < SRHC-TD < ERHC

由表 4 可知，SRHC-TD 在通信开销上面要低于 RHC、ERHC、SUHC、SRHC。但是在计算开销和存储开销上面要稍微大一点，由表 3 可知，SRHC-TD 的计算和存储开销与 RHC、SUHC、SRHC 很接近，且大概是 ERHC 的一半。

5 结束语

可再生散列链能够解决散列链资源受限的问题，近年来多处文献都提出了新的可再生散列链的构造方案，如 RHC、ERHC、SUHC、SRHC、NSUHC、SUHC-EC、SRHC-FEI 等。以上的一些构造方案，从散列链值的使用者角度来看，只有 RHC、ERHC、SUHC、SRHC 才能真正算是可再生散列链的新颖的构造方案。但这 4 种构造方案都或多或少的存在着一些性能消耗过多或者安全性不高的缺陷。本文针对它们的缺点，结合划分树、(t, n)-Mignotte's 门限、中国剩余定理等概念提出了一种新型的可再生散列链的构造方案 SRHC-TD，并且从安全性和复杂性等方面对 SRHC-TD 以及 RHC、ERHC、SUHC、SRHC 进行了详细的对比。得出本文的新方案较之具有同数量级(计算开销、存储开销)甚至更低的功耗(通信开销)、且具有更高安全性(不可否认性、t-consistency、可证明安全性)等特点，并且具有很广的应用前景。由复杂性分析可知，本方案的 SRHC-TD 在初始化时的计算开销略大，因此，在日后的工作中需要进一步的改进，使得其计算开销更小但是安全性不变或者更强。

附录：定义 3 证明

证明 由重复度的定义可知，一个重复度为 m- q 的过程即是一个(m, q)划分过程。由改进的 m 划分树可以找到所有的(m, q)划分，即找到了重复度为 m- q 的所有划分情况，再考虑各个划分的取值 v_i 的情况，可得到所有重复情况，记为 $S_{m,n}^{q}$。显然 $S_{m,n}^{q}=(C_m^{p_1}C_n^1)(C_{m-p_1}^{p_2}C_{n-1}^1)\cdots(C_{m-p_1-p_2-\cdots-p_{q-1}}^{p_q}C_{n-q+1}^1)=C_m^{p_1}C_{m-p_1}^{p_2}\cdots C_{p_q}^{p_q}\frac{n!}{(n-q)!}$。再根据重复率的定义可得 $P_q=\frac{S_{m,n}^{q}}{\sum_{i=1}^{\min(m,n)}S_{m,n}^{i}}$。

参考文献：

[1] MOHAMED H E, MUHAMMAD K K, KHALED A. One-time password system with infinite nested hash chains[J]. Communications in Computer and Information Science, 2010, 122:161-170.

[2] RAMKUMAR M, MEMON N. An efficient random key pre-distribution scheme[A]. IEEE Global Telecommunications Conference (GLOBECOM'04)[C]. Los Angeles, USA, 2004.2218-2223.

[3] SUN Y, CAO Y F, TANG L R. A multi-phase key pre-distribution scheme based on hash chain[A]. 2012 9th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)[C]. Sichuan, China, 2012.2061-2064.

[4] CHEN L, ZHANG H J, LIU N. Authentication and micropayment protocols based on self-updating hash chains[A]. The Sixth International Conference on Grid and Cooperative Computing (GCC)[C]. Los Alamitos, CA, USA, 2007.467-472.

[5] LIU D G, NING P. Multi-level μ TESLA: broadcast authentication for distributed sensor networks[J]. ACM Transactions on Embedded Computing System (TECS), 2004, 3(4):800-836.

[6] OSCAR D M, AMPARO F S, JOSE M S. A light-weight authentication scheme for wireless sensor networks [J]. Ad Hoc Networks, 2011, 9(5):727-735.

[7] EMAD A E, MOHAMMED B, HOSSAM A. Hash chain links resynchronization methods in video streaming security: performance comparison[J]. Journal of Mobile Multimedia, 2011, 7(1):89-112.

[8] GABRIELE O, STEFANO C, ROBERTO D P, *et al*. Robust and efficient authentication of video stream broadcasting[ACM Transactions on Information and System Security (TISSEC), 2 11, 14(1): 1-25.

[9] **赵源超**, **李道本**. **可再生散列链的精巧构造**[J]. **电子与信息学报**, 2006, 28(9):1717-1720.
ZHAO Y C, LI D B. An elegant construction of re-initializable hash chains[J]. Journal of Electronics & Information Techno y, 2006, 28(9):1717-1720.

[10] GOYAL V. How to re-initialize a hash chain[EB/OL]. http://

eprint.iacr.org/2004/097.pdf, 2004.

[11] ZHANG H J, ZHU Y F. Self-updating hash chains and their implementations[J]. Lecture Notes in Computer Science, 2006, 4255:387-397.

[12] ZHANG H J. A novel self-renewal hash chain and its implementation[A]. IEEE/IFIP International Conference on Embedded and Ubiquitous Computing (EUC'08)[C]. Shanghai, China, 2008.144-149.

[13] ZHANG M Q, DONG B, YANG X Y. A new self-updating hash chain scheme[A]. International Conference on Computational I telligence and Security (CIS'09)[C]. Beijing, China, 2009. 315-318.

[14] ZHANG W. Self-updating hash chains based on erasure coding[A]. 2010 International Conference on Computer, Mechatronics, Control and Electronic Engineering (CMCE)[C]. Changchun, China, 2010. 173-175.

[15] YANG X Y, WANG J J, CHEN J Y. A self-renewal hash chain scheme based on fair exchange idea (SRHC-FEI)[A]. 2010 3rd IEEE International Conference on Computer Science and Infor ion Technology (ICCSIT'10)[C]. Chengdu, China, 2010.152-156.

[16] ULUTAS M, NABIYEV V V, ULUTAS G. A new secret image sharing technique based on Asmuth Bloom's scheme[A]. International Conference on Application of Information and Communication Technologies (AICT)[C]. Baku, Azerbaijan, 2009.1-5.

[17] LEIN H, LIN C L. Strong (*n*, *t*, *n*) verifiable secret sharing scheme[J]. Information Sciences, 2010, 180:3059-3064.

作者简介：

黄海平（1981-），男，福建三明人，博士，南京邮电大学副教授、硕士生导师，主要研究方向为无线传感器网络、计算机软件在通信中的应用和信息安全。

戴庭（1989-），男，江苏高邮人，南京邮电大学硕士生，主要研究方向为可再生散列链、无线传感器网络广播认证。

王汝传（1943-），男，安徽合肥人，南京邮电大学教授、博士生导师，主要研究方向为计算机软件、计算机通信、信息安全、无线传感器网络、移动 agent 技术等。

秦小麟（1953-），男，江苏南京人，南京航空航天大学教授、博士生导师，主要研究方向为数据库技术、物联网和信息安全。

陈九天（1990-），男，江苏盐城人，南京邮电大学硕士生，主要研究方向为无线传感器网络、信息安全。