可再生散列链的精巧构造

赵源超 李道本
(北京邮电大学信息工程学院 北京 100876)

摘 要 散列(hash)链被广泛应用于电子微支付、一次性口令等多种密码学系统中。然而，因为散列链存在有限长度的限制，当链上的散列值被用尽的时候，系统需要再生新的散列链，所以系统的设计需要尽量降低再生散列链时导致的额外开销。该文提出一种高效的完全基于单向散列函数的可再生散列链的构造方法。而且，这种构造方法能够以不可否认的方式安全地再生散列链。它的高效、安全和精巧的结构将为散列链的实际应用提供广阔的前景。

关键词 散列链，单向散列函数，不可否认性
中图分类号：TN918.1 **文献标识码**：A **文章编号**：1009-5896(2006)09-1717-04

An Elegant Construction of Re-initializable Hash Chains

Zhao Yuan-chao Li Dao-ben
(*Dept of Info. Eng.*, *Beijing Univ. of Posts and Telecom.*, *Beijing* 100876, *China*)

Abstract Hash chains are widely used in various cryptographic systems such as electronic micropayments and one-time passwords etc. However, hash chains suffer from the limitation that they have a finite number of links which when used up requires the system to re-initialize new hash chains. So system design has to reduce the overhead when hash chains are re-initialized. An efficient construction which can re-initialize hash chains is proposed, and it is entirely based on one-way hash function. In the proposed construction hash chains can be securely re-initialized in a non-repudiable manner. The method will find much pragmatic application because of its efficient, secure and elegant structure.

Key words Hash chain, One-way hash function, Non-repudiability

1 引言

散列(hash)链的方法是由Lamport[1]提出的。尽管最初是用来保护一次性口令不被窃听和重放，但是由于散列链同时拥有类似于公钥技术的性质和计算的高效率，因此，它很快就被广泛用于多种密码学系统中。

例如，在为微量金额交易设计的电子微支付方案中，需要加入安全手段实现支付信息的不可否认性，而这些又会导致额外的计算开销。与微支付相对而言的宏支付往往大量采用传统的公钥签名技术，这时签名和验证都要付出很大的计算代价。对于微支付来说，由于单次交易涉及的金额很小，所以如果大量采用传统的公钥签名技术将会使得为计算付出的代价接近或者超过交易本身的价值，因此这样的系统是不能用于微支付的。微支付方案的设计原则就是效率，散列链恰好能够满足这样的需要。在采用散列链的微支付方案中，如PayWord[2]，NetCard[3]和Pederson[4]的方案，典型的操作方法是将散列链上的每个散列值作为一个付费单位的支付信息。还可以同时使用多个散列链实现多面值支付，如Netpay[5]。

除了应用于基于口令的鉴别系统和微支付系统以外，散列链的应用还包括：服务器辅助签名[6, 7]、高效组播[8, 9]和证书撤销[10]等等。

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。而且散列链的长度不能选择得太大，因为在很多情况下发送方和接收方的计算和存储能力受到一定的限制。当散列链用尽以后，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。本文正是为了解决如何高效地再生散列链这个问题，提出一种新的构造方法，以不可否认的方式安全地高效率地再生散列链，避免了使用计算负荷大的公钥签名技术。同时这种方法是对有限长散列链概念以最自然的方式进行的扩展。

下面几节分别包括如下内容：第2节简要介绍单向散列函数、散列链和相关的工作；第3节完全基于单向散列函数提出一种新的可再生散列链的精巧构造；第4节进一步提高这种新的构造方法的性能并且与相关工作进行了比较；第6节给出了本文的结束语。

2 单向散列函数和散列链简介及相关工作

2.1 单向散列函数和散列链简介

散列链是由称作“单向散列函数”的一个公开函数 h 进行递归运算得到的。其中，h 将一个任意长度(或预先确定最大长度范围)的比特串映射为一个固定长度的比特串。而且，h 满足3个性质：(1)给定 x，容易计算 $h(x)$；(2)给定 $h(x)$，求

2005-01-10 收到，2005-07-25 改回
国家自然科学基金重大项目（69931050）资助课题

出 x 在计算上是不可行的；(3)找到两个值 x 和 y，且 $x\neq y$，使得 $h(x)=h(y)$ 在计算上是不可行的。

构造长度为 N 的散列链时，首先选取一个随机的(或伪随机的)种子值 s，然后对 s 用单向散列函数 h 重复计算 N 次，得到如下的序列:

$$s,\quad h(s),\quad h^2(s),\quad \cdots\quad h^i(s),\cdots,h^{(N-1)}(s),\quad h^N(s)$$

其中 s 可记作 $h^0(s)$，$h^N(s)$ 称作散列链的根节点，作用类似于公钥技术中的公开密钥，而 s 类似于公钥技术中的私有密钥。知道 $h^N(s)$，但不知道 s，则不能生成 $h^{(N-1)}(s)$；给定 $h^{(N-1)}(s)$，则它的正确性可以很容易用 $h^N(s)$ 进行验证。依此类推，知道 $h^i(s)$，但不知道 s，则不能生成 $h^{(i-1)}(s)$；$h^{(i-1)}(s)$ 的正确性可以很容易用 $h^i(s)$ 进行验证，其中 $i=N,N-1,\cdots,1$。

以前，散列链的典型应用方式如下：首先，根节点 $h^N(s)$ 被安全地分发(即首次初始化)，这只能通过两种方式，一个是手工方式，另一个是传统的公钥签名技术，对于大规模网络通信而言，实际上只能采用后一方式。然后，从 $h^{(N-1)}(s)$ 开始，散列链上的散列值被依次释放直到到达种子值 s。此时，散列链被用尽，整个过程按上述方式重新开始，不同点在于需要一个新的随机种子值来重新初始化系统。本文的方法能够有效地避免在重新初始化系统时使用传统公钥签名技术，而仅仅使用快速的散列函数计算，同时保持了不可否认的特性，注意这里强调我们的方法应用于重新初始化阶段，不包括首次初始化。

2.2 解决散列链有限长度限制的相关工作

为了解决散列链的有限长度问题，Bicaki和Baykal[11]提出了一种基于传统公钥密码学的“无限长散列链”。尽管，这种构造方法确实解决了散列链的有限长度问题，但是由于公钥密码技术存在很重的计算负担，这恰好与使用散列链来提高效率的目的矛盾，因此这种无限长散列链是不能适用于对计算效率要求高的应用。

最近，文献[12]提出一种新的散列链构造，称作可再生散列链(Re-initializable Hash Chain, RHC)。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。这一过程能够无限次继续，实质上是将无限多个有限长度的散列链(通过Merkle[13]一次性签名的方法)“打结”在一起。这种方法具有比较精巧的结构，确实在解决散列链再生问题时仍然保持了较高的效率，应该说它是现有文献中再生散列链的最佳方法。下面一节在改进文献[12]方法的基础上，提出了更加精巧的构造方法，该构造不仅在性能上略有提高，更重要的是该构造是对有限长度散列链在概念上最自然、合理的扩展。

3 可再生散列链的精巧构造

为了下面叙述方便将(有限长度的)传统散列链(Conventional Hash Chain, CHC)简称作 CHC，将我们构造的可再生散列链如同文献[12]一样也简称作 RHC。

这里，仅仅利用CHC和Merkle[13]的一次性签名方案来构造RHC。Merkle提出的方法是这样的：对n个bit消息的每一个 bit 都生成一个随机数 $x_i(i=1,2,\cdots,n)$，计算 $y_i=h(x_i)(i=1,2,\cdots,n)$，而且公开所有 y_i；当消息的第i个bit为‘1’时，公开 x_i，若为‘0’，不公开任何值。为了防止接收者将收到‘1’伪称为收到‘0’，签名者需要也对消息中‘0’的个数进行签名，而且对消息中‘0’的个数的签名方法与对消息体的签名方法完全相同。因此，在Merkle的方法里，对n个bit消息的签名分别需要准备 $n+\lfloor\log_2(n)\rfloor+1$ 个x和y。关于这个方法更详细的描述，可以参见文献[14]。

设每个CHC的长度都为N。同时，设单向散列函数h的输出的长度为L(bit)(例如：MD5[15]算法的输出是L=128bit)。下面分为两种不同的阶段进行讨论：

(1)首次启动散列链阶段　首先，发送方给长度为 L 个 bit 的消息准备一对一次性签名密钥的实例，包括产生 $L+\lfloor\log_2(L)\rfloor+1$ 个随机数，将它们的级联记作 S_U，并且分别以这些随机数作为单向散列函数 h 的输入计算出相应的 $L+\lfloor\log_2(L)\rfloor+1$ 个散列函数值，将这些散列函数值的级联记作 P_U，可以将 S_U 和 P_U 分别看作一次性签名的私钥元素和公钥元素。实际上该一次性签名密钥的实例并不是为本次散列链的启动提供签名，而是将为下次再生散列链提供签名。

然后，发送方将 P_U 作为种子值计算一个长度为 N 的 CHC，它的根为 $h^N(P_U)=h(h(h(\cdots h(P_U)\cdots)))$ (共 N 次函数计算)。注意，这里的种子值 P_U 与传统 CHC 的种子值的唯一区别是，前者的种子值长度为 $\left(L+\lfloor\log_2(L)\rfloor+1\right)\bullet L$ (bit)，而后者的种子值长度通常选择为 L(bit)。因为是首次启动散列链，所以这个根节点 $h^N(P_U)$ 需要以安全的方式分发给接收方，当然，使用传统的公钥数字签名技术是最直接的选择。将一次性签名的公钥元素 P_U 作为种子值正是为了在后续再生散列链的时候避免开销大的公钥计算。

在此之后，就可以按照通常的方式使用这个散列链了。发送方依次释放链上的各个散列值，直到发送方将这条 CHC 的种子值 P_U 发给接收方，这时，该 CHC 被耗尽，就需要进行后续的再生了，即进入下面的操作阶段。

(2)后续再生散列链阶段　首先，按照与(1)相同的方式，发送方生成一对新的一次性签名密钥的实例 S_U' (也是随机数)和 P_U'。

然后发送方将 P_U' 作为新的种子值计算一个长度为 N 的新的 CHC，它的根为 $h^N(P_U')$。这个新的根节点 $h^N(P_U')$ 将被利用一次性签名的方式安全地(不可否认地)传递给接收方，即发送方现在向接收方发出对 $h^N(P_U')$ 签名所需公开的部分 S_U (注意不是 S_U')。由于接收方在阶段(1)的最后已经得到 P_U (注意不是 P_U')，所以现在接收方能够通过 P_U 和 S_U 的公开部分这两者的结合来验证 $h^N(P_U')$。

此时，这条经过再生的散列链就可以投入使用了，直到到达它的种子值 P_U' 。

这个阶段里准备的一次性签名密钥实例将用于下一次再生新的 CHC，于是一个 RHC 就是一个 CHC 和一次性签名的组合，显然，这种再生过程能够以阶段(2)相同的方式无限次进行，即再生能力为无限次，而且每次再生时都避免了进行开销大的公钥计算，同时整个过程都保证了不可否认的安全性。

4　可再生散列链的性能

第 3 节提出的 RHC 的基本构造最大的特点就是在再生和使用散列链的时候，逐层地提供并且保持了不可否认性，而且这种不可否认性的实现中避免了公钥计算(初始启动除外)。这种特点直接来源于构造中仅仅使用了单向散列函数。实际上，与文献[12]类似地通过巧妙地利用一次性签名中私钥元素 S_U 的公开部分，可以进一步提高系统的效率，如下执行操作：

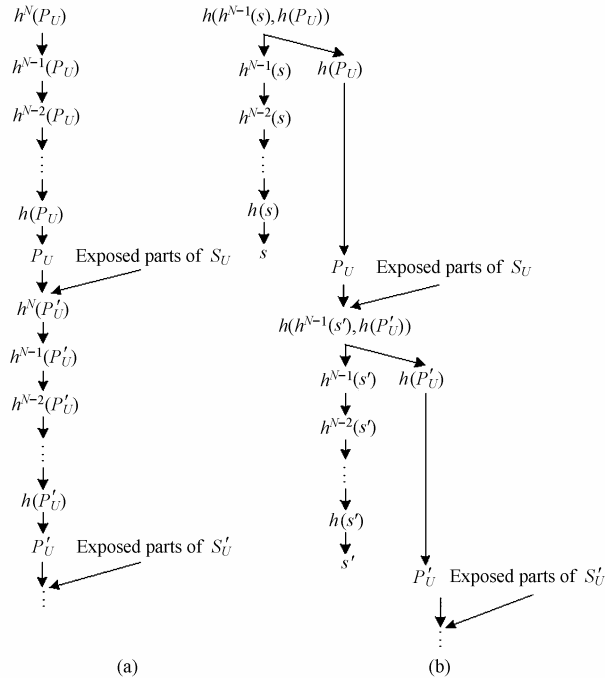

图 1　两种可再生散列链构造的比较

(a) 我们的 RHC 构造 (b) 文献[12]的 RHC 构造

Fig.1 The comparison between two constructions of RHC

(a) Our construction of RHC　(b) The construction of in Ref. [12]

(1) 首先，按通常 CHC 的方式使用 RHC 上的 N 个链值直到到达 P_U 。

(2) 然后，发送方计算出 $h^N(P_U')$，将其传递给接收方，接收方储存 $h^N(P_U')$，先不对其进行验证。接下来，发送方根据 $h^N(P_U')$ 的值确定 S_U 中需要公开的那些随机值，一次一个地(在不同的消息里)释放给接收方，接收方能够用 P_U 的相应部分对它们进行验证，这样它们就可以与前面的 N 个链值一样起到相同的作用。

(3) 待接收方接到 S_U 中需要公开的所有随机值以后，就可以对 $h^N(P_U')$ 进行验证，新的 RHC 就又可以按照(1)和(2)投入使用了。

平均来说，S_U 中需要公开的随机值的个数为 $\left(L+\lfloor\log_2(L)\rfloor+1\right)/2$，所以一条长度为 N 的 RHC 平均可以被使用 $N+\left(L+\lfloor\log_2(L)\rfloor+1\right)/2$ 次。

与文献[12]的构造相比较，本文的构造是以最自然的方式将 CHC 的概念拓展成为 RHC 的概念，这可以通过图 1 清晰简洁地说明(图中“→”的指出方为“→”的指入方提供不可否认的证据)：

很明显，本文的构造不需要像文献[12]那样特殊地使用 $h(P_U)$，在本文的构造中 $h(P_U)$ 自然地成为 RHC 上的一个链值，因此根节点的生成就不需要特殊处理，相应地接收方也不用提前存储 $h(P_U)$ 。当然文献[12]也可以不使用 $h(P_U)$，而可以直接使用 P_U，同时将根节点改为 $h(h^{N-1}(s),P_U)$，参见图 2 (这可以从图 1(b)中直观地想到)，但是在使用当前 RHC 而不能确定是否需要启动下一个 RHC 的情况下，传递 P_U 没有实际意义，尤其是 P_U 的长度较大，在不必要的情况下，当然不希望 P_U 占据通信量。尽管在文献[12]中没有明确说明其中原因，然而，$h(P_U)$ 的特殊使用只能有这样的解释才是合理的。本文的构造使得 P_U 在当前 RHC 结束的时候才被传递成为很自然的事情，从而不会造成不必要的通信量。另外，本文的构造不需要额外生成随机种子值，由于 S_U 是随机的，经过单向散列函数计算以后的结果 P_U 当然也就具有随机性，所以将 P_U 作为种子值是顺理成章的事情。由以上的分析可以看出，本文的构造在性能上比文献[12]的构造略有提高。

这里，再次强调本文的 RHC 构造在整个过程中都保持了不可否认性：散列链靠近根的节点证明远离根的节点的不可否认性；一次性签名公钥和相应私钥的公开部分联合证明下次再生时的根节点的不可否认性。

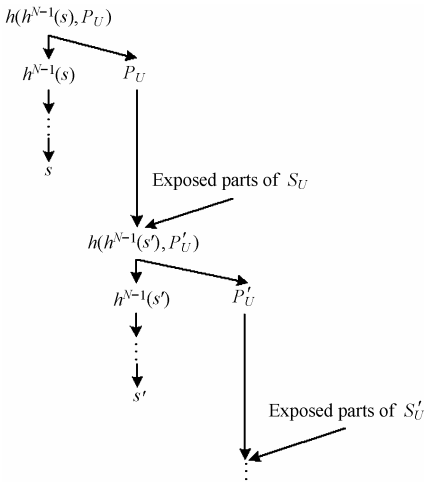

图 2　文献[12]的 RHC 构造的一个变种

Fig.2 A variation of RHC construction in Ref. [12]

5 结束语

由于传统散列链存在有限长度的限制，所以大多数应用散列链的密码学系统都迫切需要一种精巧的方法来安全地再生散列链，同时又要求保持较高的效率。本文提出一种新的可再生散列链的构造方法，它能够以不可否认的方式安全地再生散列链，而且这种再生过程能够无限次地进行。同时，本文的可再生散列链构造避免了使用计算开销大的公钥技术，计算中只使用了计算效率高的单向散列函数。这里的效率和不可否认性都直接来源于单向散列函数的性质。这种精巧的构造是对传统的有限长度散列链概念的最自然的拓展，将为散列链在各种密码学系统中的应用提供广阔的前景。

参 考 文 献

[1] Lamport L. Password authentication with insecure communication. *Communications of the ACM*, 1981, 24(11): 770−772.

[2] Rivest R, Shamir A. Payword and MicroMint: two simple micropayment schemes. Proceedings of the 4th Security Protocols International Workshop (Security Protocols), Lecture Notes in Computer Science, Cambridge, UK, 1996, 1189: 69−87.

[3] Anderson R, Manifavas H, Sutherland C. NetCard—a practical electronic cash system. Proceedings of the 4th Security Protocols International Workshop (Security Protocols), Lecture Notes in Computer Science, Cambridge, UK, 1996, 1189: 49−57.

[4] Pedersen T. Electronic payments of small amounts. Proceedings of the 4th Security Protocols International Workshop (Security Protocols), Lecture Notes in Computer Science, Cambridge, UK, 1996, 1189: 59−68.

[5] Dai X, Lo B. Netpay—An efficient protocol for micropayments on the WWW. URL: http://ausweb.scu.edu.au/aw99/papers/dai /paper.html.

[6] Asokan N, Tsudik G, Waidners M. Server-supported signatures. Proceedings of the Fourth European Symposium on Research in Computer Security, Lecture Notes in Computer Science, Berlin, Germany, 1996, vol. 1146: 131−143.

[7] Ding X, Mazzocchi D, Tsudik G. Experimenting with server-aided signatures. Proceedings of Network and Distributed Systems Security Symposium 2002, San Diego, CA, USA, Feb. 2002: 1−15.

[8] Perrig A, Canetti R, Song D, Tygar D. Efficient authentication and signing of multicast streams over lossy channels. Proceedings of IEEE Security and Privacy Symposium S&P 2000, Oakland, CA, USA, May 2000: 56−73.

[9] Perrig A, Canetti R, Song D, Tygar D. Efficient and secure source authentication for multicast. Proceedings of Network and Distributed System Security Symposium 2001, San Diego, CA, USA, Feb. 2001: 35−46.

[10] Aiello W, Lodha S, Ostrovsky R. Fast digital identity revocation. Proceedings of Advances in Cryptography – Crypto '98, Lecture Notes in Computer Science, Berlin, Germany, 1998, vol. 1462: 137−152.

[11] Bicakci K, Baykal N. Infinite length hash chains and their applications. Proceedings of the 11th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'02), Pittsburgh, USA, June 2002: 57−61.

[12] Goyal V. How to re-initialize a hash chain. URL: http://eprint.iacr.org/ 2004/097.pdf

[13] Merkle R. A digital signature based on a conventional encryption function. Proceedings of Advances in Cryptology – CRYPTO '87, Lecture Notes in Computer Science, Santa Barbara, CA, USA, 1988, vol. 293: 369−378.

[14] Menezes A, van Oorschot P, Vanstone S. Handbook of Applied Cryptography. Boca Raton, Florida, USA: CRC Press, 1996.

[15] Rivest R. The MD5 message digest algorithm. RFC 1321, April 1992.

赵源超： 男，1971 年生，博士生. 从事移动通信安全方面的研究.

李道本： 男，1939 年生，教授，博士生导师. 主要从事 Las-CDMA 移动通信系统的研究.