基于 Matlab 的 Hash 算法 BLAKE 的设计与实现

李世明 包小敏

(西南大学数学与统计学院 重庆 400715)

摘 要 Hash 算法 BLAKE 是新一代安全 Hash 标准 SHA-3 全球公开征集过程中进入最后一轮的 5 个候选者之一。给出一种基于 Matlab 的带有图形界面 GUI 的 BLAKE 程序的设计与实现过程。本程序可用于实际的 BLAKE Hash 值的运算,最重要的是为 BLAKE 的教学与实验提供了更方便直观的工具。

关键词 Hash,SHA-3,BLAKE,Matlab

中图法分类号 TP309 **文献标识码** A

Design and Implementation of the Hash Algorithm BLAKE Based on Matlab

LI Shi-ming BAO Xiao-min

(School of Mathematics and Statistics, Southwest University, Chongqing 400715, China)

Abstract The hash algorithm BLAKE is one of the five candidates for SHA-3(Secure Hash Algorithm-3) competition at the last round. This paper described the design and implementation of BLAKE based on Matlab. Our program with a GUI(Graphic User Interface) can be used both in practical Balke Hash value calculation and Blake teaching or experiments.

Keywords Hash,SHA-3,BLAKE,Matlab

1 引言

Hash 函数是密码学的一个基本工具,在信息安全领域,如数字签名、和消息的完整性等方面有着广泛的应用。标准的 Hash 函数有 MD 系列和 SHA 系列两大类,MD 系列中的 MD5 是由 Ronald L. Rivest 在 1992 年 8 月向 IETF(The Internet Engineering Task Force)提交的,算法的描述和 c 语言源代码在 Internet RFC 1321 中有详细的描述;SHA 系列是由美国国家标准与技术研究所(NIST)组织设计的,1993 年,NIST 公布了安全 Hash 标准(Secure Hash Standard,SHS) SHA-0,两年之后 NIST 公布了 SHA-1,用以取代 SHA-0。2002 年 NIST 发布了修订版 FIPS 180-2[1],其中增加了 3 种新的 SHA 版本,Hash 值的长度分别为 256、384 和 512 比特,分别称为 SHA-256、SHA-384、SHA-512。2008 年,NIST 发布了修订版 FIPS 180-3[2],其中增加了 1 种新的 SHA 版本,Hash 值的长度为 224 比特,被称为 SHA-224。在 FIPS 180-3 中,SHA-224、SHA-256、SHA-384、SHA-512 被统称为安全 Hash 标准(SHS)SHA-2。新的版本与 SHA-1 具有相同的基础结构,使用了相同的压缩函数、模运算和二元逻辑运算。

Hash 函数最基本的要求之一就是要具有抗碰撞性。近年来,各国学者对 MD5、SHA-0 以及 SHA-1 等 Hash 函数的碰撞的发现,对新的 Hash 函数的研究产生了极大的刺激和推动作用。从技术上讲,虽然 SHA-1 的碰撞能被找到,但并不意味着有实际的攻击意义。无论如何,现有的方法已经使短时间内找 SHA-1 的碰撞成为可能,这对现有的 Hash 函数的安全性是一个巨大的威胁。作为信息安全领域广泛使用的 Hash 函数,其安全性对信息安全的重要性是显而易见的。美国政府在 2010 年以前就已经停止使用 SHA-1,改用还未发现碰撞的 SHA-2 家族(SHA-224、SHA-256、SHA-384 和 SHA-512)中的 Hash 算法。同时,NIST 发布了 SHA-3 家族计划,其目的就是寻找新的 Hash 函数,以增强现有的安全 Hash 标准 SHA-2,这样就有了 SHA-3 的公开竞赛。

SHA-3 公开征集的整个过程仿照高级数据加密标准(Advanced Encryption Standard,AES)的征集过程,在全球范围展开。世界各国的研究者共提交了 64 个算法,BLAKE[3] 就是其中之一。在经历了 2008 年的初选、2009 年的第一轮筛选及 2010 年的第二轮筛选后,BLAKE 成功进入 SHA-3 的最后一轮(final round),成为最后 5 个候选 Hash 算法之一。目前,针对这 5 个算法,各国相关的学者都在积极进行研究,以便在 2012 年能从这 5 个算法中选出一个作为 SHA-3 标准。因此,对 BLAKE 的研究除了其本身的理论意义之外,还有重要的现实意义。

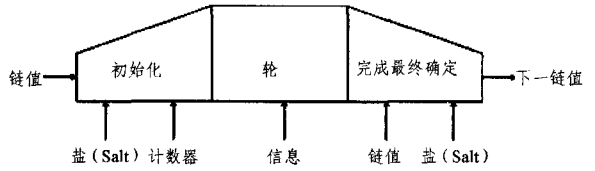

图 1 BLAKE 压缩函数的局部宽管道

李世明(1986—),男,硕士生,主要研究方向为密码学等,E-mail:lsm1986@swu.edu.cn;包小敏(1959—),男,博士,教授,主要研究方向为密码学、组合数学等。

BLAKE 采用了 HAIFA[4] 迭代框架。有人认为以往 Hash 函数所普遍使用的 Merkle-Damgard(MD)构造框架在本质上是有一定缺陷的，而 HAIFA 迭代框架则是相应的一个替换方案。其主要思想是在压缩过程中引进两个额外的参数：计数器(counter)——已经 hashed 的比特数和盐(salt)。BLAKE 的设计借鉴了 Hash 函数 LAKE[5](见图 1)和流密码 Chacha[6] 的设计方法，它的内部结构是局部宽管道(local wide-pipe)，它的压缩算法是流密码 ChaCha 的修改版。

2 BLAKE 的设计与实现

BLAKE 目前还不是一款普遍使用的 Hash 函数，其实现的语言除了 C 外还没有见到其它的。因此设计一个带有图形用户界面(Graphic User Interface，GUI)的 BLAKE 进行研究与学习是很有必要的。我们设计的图形用户界面适用于 Hash 算法的教学与科研，不仅能实现对由 BLAKE 产生的 Hash 值的输出，还能输出 Hash 运算过程中的中间数据，这对研究 BLAKE 无疑有很大的帮助。

Matlab 采用解释性语言，便于编写、调试及修改；其次，Matlab 的图形用户界面设计环境 Guide 使得图形用户界面的设计与实现变得简单且直观；再者 Matlab 对数据的分析有很多内部命令可以直接使用，用 Matlab 编写的程序所得到的数据可以直接调用其内部命令来做进一步的统计和分析；另外，Matlab 是一款常用的教学软件，很多本科生都会使用，因此用 Matlab 编写的 BLAKE 便于他们学习和研究 BLAKE。在编写过程中我们也发现 Matlab 在大数存储方面存在一定的局限性，但我们通过巧妙的方法，运用“形式”运算避免了大数运算在 BLAKE 内部处理的影响。

2.1 算法简介

BLAKE 是由 4 个函数组成的家族：BLAKE-224、BLAKE-256、BLAKE-384 和 BLAKE-512(见表 1)。如 SHA-2 一样，根据不同的初始值(initial values)、不同的填充(padding)和不同的截断输出(truncated output)，BLAKE 有 32-比特的版本(BLAKE-256、BLAKE-224)和 64-比特的版本(BLAKE-512、BLAKE-384)。

表 1 Hash 函数 BLAKE 的属性(单位：比特)

算法	字	信息	组	摘要	盐
BLAKE-224	32	$<2^{64}$	512	224	128
BLAKE-256	32	$<2^{64}$	512	256	128
BLAKE-384	64	$<2^{128}$	1024	384	512
BLAKE-512	64	$<2^{128}$	1024	512	512

如表 1 所列，BLAKE 就是把一定范围内的任意长度的输入，通过具体的算法，变换成固定长度的输出。该输出的值称为 Hash 值或消息摘要。简单地说就是一种将一定范围内的任意长度的输入消息压缩成某一固定长度的消息摘要的函数。BLAKE 提供了 4 种算法、两类版本、4 种 Hash 长度，其中 32-比特版本可以处理信息的长度$<2^{64}$；64-比特的版本可处理信息的长度$<2^{128}$。

(1)主体结构算法

$h^0 \leftarrow IV$

for i=0,1,…,N−1

$h^{i+1} \leftarrow$compress(h^i, m^i, s, l^i)

return h^N

其中，IV 是初始值，是一个常量；N 是消息在预处理之后的分组数；m^i 是第 i 组的消息块；s 是所添加的盐；l^i 是第 i 组的计数器。

(2) 消息预处理

(a) 消息的填充：BLAKE 对消息的输入是先进行填充再进行分块，其中 BLAKE-256 和 BLAKE-224 每块有 512 比特，填充之前的消息长度小于 2^{64} 比特；BLAKE-512 和 BLAKE-384 每块有 1024 比特，填充前的消息长度小于 2^{128} 比特。

(b) 扩展：填充之后有 N 个 512 比特(或 1024 比特)的消息块。每个消息块又分为 16 个 32 比特(或 64 比特)字。

(3) 压缩运算(compress)

(a) 输入的 4 个参数：链值 $h=h_0,\cdots,h_7$、信息块 $m=m_0,\cdots,m_{15}$、盐 $s=s_0,\cdots,s_3$ 和计数器 $t=t_0,t_1$。这 4 个参数总共有 30 个字，字的长度与具体的算法有关。

(b) 初始化：在这里将根据不同的初始参数产生 16 个初始的字 $v_0,\cdots,v_{15}$。

(c) 轮函数：在这里，所运行的轮数与 BLAKE 的具体类型有关，首先要做的就是将初始化得到的 16 个字按一定的组合提取，以 4 个字为单位构成 8 组；接下来就是将每一组带入相应的 G 函数，G 函数循环移位的量与 BLAKE 的类型有关。

(d) 终结：将轮函数中产生的新的 16 个字 $v_0,\cdots,v_{15}$ 与初始的 $h_0,\cdots,h_7$ 和 $s_0,,s_3$ 做相应的异或运算得到新的 $h_0,\cdots,h_7$。

(4) 主函数

将 BLAKE 的初始常量 IV 赋给 h，然后做 N(信息填充后的块数)次压缩运算，最后输出新的 $h_0,\cdots,h_7$。

(5) 截断输出

BLAKE-256 与 BLAKE-512 以最后输出的 $h_0,\cdots,h_7$ 为 Hash 值，BLAKE-224 以最后输出的 $h_0,\cdots,h_7$ 中的 $h_0,\cdots,h_6$ 为 Hash 值，BLAKE-384 以最后输出的 $h_0,\cdots,h_7$ 中的 $h_0,\cdots,h_5$ 为 Hash 值。

2.2 BLAKE 的程序设计

Hash 函数 BLAKE 是以不同的类型按 32(或 64)比特为一个字计算相应长度的 Hash 值。我们设计 BLAKE 程序的目的是为了教学与科研，所以在设计带有图形界面 GUI 的 BLAKE 时，对原有的 BLAKE 算法中的变量进行了记录提取，以便用户提取相应的中间数据用以研究。同时，我们还在设计 BLAKE 的 GUI 时添加了提示信息和存盘提取功能，以便用户使用。

带图形界面 GUI 的 BLAKE 程序主要分为两大部分，第一部分是用户界面程序设计；第二部分是 BLAKE 程序族设计。

2.2.1 用户界面程序设计

在 Matlab 里面，有自带的用户界面设计模板，我们在这部分的设计主要是在显示面板布局完成之后，在相应的按钮函数中添加相应的语句，实现具体的功能，让用户的使用比较方便。面板布局主要有四大板块，一是 BLAKE 类型选择，二是数据显示区，三是数据控制面板，四是控制面板。

BLAKE类型选择：这里提供了4种BLAKE类型进行选择，以便用户进行不同类型的BLAKE研究。

数据显示区：这里提供了4个数据显示窗口。首先是信息显示窗口，接受十六进制或二进制信息。其次是Salt值显示窗口，接受十六进制或二进制信息。根据不同的BLAKE类型，然后是中间数据显示窗口，该窗口可显示你具体感兴趣的中间数据值，输出形式为十六进制或二进制。最后是Hash值显示窗口，这里将根据不同的BLAKE类型显示相应的Hash值，输出形式为十六进制或二进制。

数据控制面板：这里提供了相应数据显示窗口的数据控制按钮和一个存盘选择按钮。首先是对信息显示窗口的控制，这里提供了4个具体的控制端，一是控制信息显示窗口的数据类型，二是提供产生随机数据的长度，三是产生随机数按钮，四是导入数据按钮。其次是对Salt值显示窗口的控制，这里提供了3个具体的控制端，一是Salt值的数据类型，二是产生随机Salt按钮，三是导入Salt按钮。再次是对中间数据娴熟窗口的控制，这里提供了5个控制端，一是中间数据的类型，二是获取信息的最大组数按钮，三是中间数据的组数输入，四是轮数选择，五是中间数据变量选择。然后是对Hash值显示窗口的控制，这里提供1个控制端，即Hash值输出的数据类型。最后是存盘选择按钮，这里提供具体的存盘数据选择。

控制面板：控制面板中设置了4个按钮，一是具有对所有显示窗口中数据清除功能的重置按钮，二是执行Hash运算和中间数据显示的执行按钮，三是对具体的数据进行保存的存盘按钮，四是退出改程序的关闭按钮。

2.2.2 BLAKE程序族设计

根据BLAKE算法的描述，我们设计了一个BLAKE.m文件，主要功能是根据参数选择相应的BLAKE类型进行运算。而对于每一类BLAKE，我们设计了4个主要的部分，一是BLAKE的函数主文件，二是信息添加处理文件，三是压缩函数文件，四是G函数文件。

(1) BLAKE文件(BLAKE.m)

通过输入信息、盐，以及BLAKE的具体类型，得到具体BLAKE类型的Hash函数值以及中间所有数据。

(2) BLAKE函数主文件(BLAKE_*.m)

根据BLAKE的类型，我们设计了4个不同的函数主文件：BLAKE_256.m，BLAKE_224.m，BLAKE_512.m，BLAKE_384.m。每个文件里面都有相同的算法结构，只是在常量与运算的字的大小上有一定的差异。这里添加了变量BlockH来记录每一轮的h值。

(3) 信息添加处理文件(Padding_Message_*.m)

由于每一种类型的数据的处理方法不一样，我们设计了4个不同的信息添加处理文件：Padding_Message_256.m，Padding_Message_224.m，Padding_Message_512.m，Padding_Message_384.m。每个文件都添加了变量N，t分别记录信息的最大分组数N和计数器t。

(4) 压缩函数文件(compress_*.m)

BLAKE的各个类型在这部分按字的大小分为两部分，分别应用于32比特类型和64比特的类型。所以我们设计了两个不同的压缩函数文件：compress_32.m，compress_64.m。每个文件添加了变量InitialStateV，RoundV分别记录初始的V值和每一轮的V值。

(5) G函数文件(G_function_*.m)

同压缩函数一样，我们设计了两个不同的G函数文件：G_function_32.m，G_function_64.m。这里面主要用到的是字的加法运算、异或运算以及循环移位。由于MATLAB在大数存储方面的局限型，直接调用MATLAB本身的函数来实现BLAKE-512、BLAKE-384是不可能的。我们根据字的加法定义以及十进制加法的原理构造了二进制"形式"加法，与普通加法的不同之处在于，其运算过程与普通加法是一致的，只是在最后提取结果时将最高位进位的值舍去(相当于字的加法做模运算)即可。

2.3 BLAKE的实现

2.3.1 运行流程

本文实现的BLAKE，其基本功能是对给定信息用指定的Salt进行求Hash值的运算，图2是其流程图。程序运行后首先显示出图形用户界面(见图3)。用户在数据控制面板上选择相关参数，并输入待处理的数据，然后点击控制面板中的执行按钮。运行的结果将在Hash值显示窗口中显示出来。

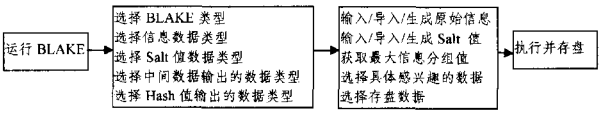

图2 BLAKE步骤

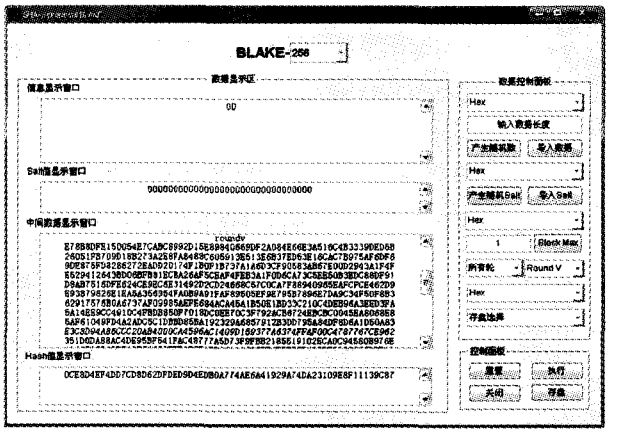

图3 BLAKE图形用户界面GUI

2.3.2 输出Hash值

该系统的基本功能是对给定信息用指定的Salt进行求Hash值的运算。运行程序显示出主图形用户界面(见图3)后会看到正上方有BLAKE-256字样，下面将详细介绍如何输出Hash值。

首先是BLAKE的类型选择，在"BLAKE-"字样之后选择相应的BLAKE类型。然后是信息的输入，信息输入过程中一是选择信息的数据类型，二是输入信息，可直接在信息显示窗口中输入十六进制或二进制信息。也可在输入随机信息长度后产生符合类型的随机信息，这可为今后的随机性分析打下数据来源的基础；或直接从文件中导入数据，这可以方便用

户对已有的信息进行 Hash 运算，也适用于大量数据的检测输入。

接着是 Salt 值的输入，Salt 值的输入过程中一是选择 Salt 值的数据类型，二是输入 Salt 值，可直接在 Salt 值显示窗口中输入十六进制或二进制的 Salt 值。也可用产生随机 Salt 按钮来随机生成一个长度符合要求的 Salt 值。或用导入按钮，导入一个固定的 Salt 值。

最后是 Hash 值的输出，选择 Hash 值输出数据类型，然后点击控制面板里面的执行按钮，那么在 Hash 值显示窗口中将会显示出 Hash 值。

在本系统中的每一个按钮都设计了具体的错误提示，会提示用户具体错在哪里，如何进行修改。例如用户想要随机产生一个 576 比特的二进制信息串，而用户忘记选择信息的数据类型而点击了产生随机数按钮，那么就会弹出错误信息——"请选择数据的类型"。

2.3.3 *输出中间数据*

设计本软件的主要目的之一是为研究 Hash 算法 BLAKE 的内部构造与设计提供一个工具。就像前面提到的，重点研究每一轮的输出以及 G 函数内部构造。研究轮效应就需要将各轮的输出记录下来，然后通过相应的工具进行分析，这里我们可以对随机分析提供足够的数据来源。

用户若对某些中间数据感兴趣，就需要对中间数据输出控制端做出一些选择。首先，选择中间值输出的数据类型(Hex/Bin)；然后输入感兴趣的组数(信息组数的最大值在输入信息之后，点击"BlockMax"按钮，则该信息的最大组数值 N 将显示在旁边的窗口中，查询的时候可在该窗口输入 1～N 之间的任意整数)；再选择在这一分组下的轮数，接着选择具体感兴趣的数据位置。设置好之后，用户只需要点击控制面板的执行按钮，用户感兴趣的数据就会显示在中间数据显示窗口中。

举例来说，如果选择 BLAKE 的类型是 256，16 进制的信息为'00'，16 进制的 Salt 值为'00000000 00000000 00000000 00000000'，数据类型都选择"Hex"，组数输入"1"，轮数选择"所有轮"，中间数据选择"RoundV"，点击控制面板的执行按钮，那么用 BLAKE-256 进行 Hash 运算时压缩函数每一轮的数据与 Hash 值将在各个窗口中显示。此时，图形用户界面的状态如图 3 所示。

用户可根据需要存储相关的数据，只需通过数据控制面板中的存盘选择按钮选择具体需要保存的数据，然后点击控制面板中的存盘按钮进行相关的操作。

结束语 本文在全球公开征集新一代安全 Hash 标准 SHA-3 的最后阶段，对可能成为 SHA-3 标准的 BLAKE 进行了初步的研究，并利用 Matlab 中的 Guide，开发了带有图形用户界面的 BLAKE。利用这个程序可以根据需要很方便地输出 BLAKE 运行的内部结果。这对 BLAKE 的教学以及 BLAKE 的深入研究都具有积极的意义。

参 考 文 献

[1] NIST. Secure Hash Standard[S]. Federal Information Processing Standards Publication(FIPS) 180-2,2002

[2] NIST. Secure Hash Standard[S]. Federal Information Processing Standards Publication(FIPS) 180-3,2008

[3] Aumasson J-P, Henzen L, Meier W, et al. SHA-3 proposal BLAKE* version 1. 3[EB/OL]. http://www. 131002. net/blake/blake. pdf,2010

[4] Biham E, Dunkelman O. A framework for iterative hash functions-HAIFA[R]. ePrint report 2007/278. 2007

[5] Aumasson J-P, Meier W, Phan R C-W. The hash function family LAKE[C]// FSE. 2008

[6] Aumasson J-P, Fischer S, Khazaei S, et al. New features of Latin dances: analysis of Salsa, ChaCha, and Rumba[C]// FSE. 2008

[7] Stinson D R. Cryptography Theory and Practice(Third Edtion)[M]. Taylor & Francis Group, FL, USA, 2006

(上接第 37 页)

[2] Chen L, Harrison K, Soldera D, et al. Applications of multiple trust authorities in pairing based cryptosystems [C]// Proc of the International Conference on Infrastructure Security 2002. Berlin: Springer, 2002: 260-275

[3] Goldberg K I. A distributed private-key generator for identity-based cryptography [R]. University of Waterloo, 2007

[4] Gentry C. Certificate-based encryption and the certificate-revocation problem [C]// Proc of Eurocrypt 2003. Berlin: Springer, 2003: 272-291

[5] Al-Riyami S S, Paterson K G. Certificateless public key cryptography [C]// Proc of the 9th International Conference on the Theory and Application of Cryptology and Information Security. Berlin: Spring, 2003: 452-473

[6] Lee B, Boyd C, Dawson E, et al. Secure key issuing in id-based cryptography [C]// Proc of the 2nd Australasian Information Security Workshop. Australia: CRPIT, 2004

[7] Gangishetti R, Gorantla M C, Das M, et al. Threshold key issuing in identity-based cryptosystems[J]. Computer Standards & Interfaces, 2007, 29(2): 260-264

[8] Chunxiang X, Junhui Z, Zhiguang Q. A Note on Secure Key Issuing in ID-based Cryptography [EB/OL]. http://eprint. iacr. org/2005/180

[9] Lee B. Unified public key infrastructure supporting both certificate-based and ID-based cryptography [C]// Proc of the 2010 International Conference on Availability, Reliability and Security. Poland: IEEE, 2010: 54-61

[10] Boyen X. A tapestry of identity-Based encryption: practical frameworks compared[J]. International Journal of Applied Cryptography, 2008, 1(1): 3-21

[11] Boyen X. The BB1 identity-based cryptosystem: A standard for encryption and key encapsulation [EB/OL]. http:// grouper. ieee. org/groups/1363

[12] Martin L. Introduction to Identity-Based Encryption [M]. Boston: Artech House, 2008

[13] Kate A, Goldberg I. Asynchronous Distributed Private-Key Generators for Identity-Based Cryptography [EB/OL]. http://eprint. iacr. org/2009/355

[14] Boneh D, Boyen X. Efficient Selective-ID Secure Identity Based Encryption Without Random Oracles [C]// Proc of EUROCRYPT 2004. Switzerland: Springer, 2004: 223-238