Blowfish 算法优化及其在 WSN 节点上的实现

谢 琦 曹雪芹
(郑州大学信息工程学院 河南 郑州 450001)

摘 要 Blowfish 加解密算法由 Bruce Schneier 于 1993 年 11 月提出,快速、安全、简单等特点使该算法得到了广泛的应用。针对 WSN 节点内存空间小、执行能力有限等特点,对 Blowfish 算法进行研究,从多个方面对算法进行优化,提出一种在 WSN 节点上高效运行并且占用较少 RAM 空间的实现方案,通过数据对比分析优化算法的性能。在 CC2530 处理器上实现的结果表明,Blowfish 算法的优化方案可在 WSN 节点上运行并且占用了较少的 RAM 空间。该方案也可应用于存储空间较小的嵌入式系统中。

关键词 Blowfish 算法研究 算法优化 WSN 节点 内存空间

中图分类号 TP393 **文献标识码** A DOI:10.3969/j.issn.1000-386x.2013.07.084

BLOWFISH ALGORITHM OPTIMISATION AND ITS IMPLEMENTATION ON WSN NODES

Xie Qi Cao Xueqin
(*School of Information Engineering*, *Zhengzhou University*, *Zhengzhou 450001*, *Henan*, *China*)

Abstract Blowfish encryption and decryption algorithm was put forward by Bruce Schneier in November 1993. Its characteristics of fast, safe and simple made it being widely used. In light of the features of the nodes of wireless sensor networks (WSN) such as small in memory space and limited in implementation capacity, we study the Blowfish algorithm and optimise it in many aspects. We propose an implementation scheme, which can effectively run on WSN nodes while consuming less random access memory (RAM). The performance of optimisation algorithm is analysed by the comparison of the data. The results achieved on CC2530 processor show that the optimised scheme of Blowfish algorithm can execute efficiently on WSN nodes and consume less RAM memory. The method can also be applied to embedded systems with constrained memory space.

Keywords Blowfish Algorithm study Algorithm optimisation WSN nodes Memory space

0 引 言

Blowfish 是一种对称密钥分组加密算法。32 位处理器诞生后,Blowfish 算法在加密速度上超越了 DES,引起了人们的关注[1]。该算法具有加密速度快、紧凑、密钥长度可变、可免费使用等特点[2,3]。这些特点使 Blowfish 被广泛应用于众多的加密软件,如国外的 Shellfish、SplashID 以及国内的一些网络计费系统、Linux 和一些网络传输控制软件中的核心加密部分都使用了该算法[4,5]。目前为止,使用 Blowfish 算法的产品超过了 150 种。

WSN 节点的计算能力、存储空间等都是有限的,在其上面实现的算法代码长度要短,并且时间复杂度和空间复杂度都不能太大,而 Blowfish 的加解密部分主要由“异或”和“加法”构成,运行时仅需 4K 多内存甚至更少,并且代码长度比较短,对 RAM 和 ROM 的需求都不是很大。

鉴于以上两点,本文选用 8051CPU 的 CC2530 F128 为硬件平台,采用此芯片设计出的 ZigBee 网络节点,完全支持 ZigBee2007/Pro 协议栈的通信模块[6,7],在此通信模块中使用 Blowfish 算法对传输数据加密,最终初步实现两个 WSN 节点之间数据的加密传输。优化算法用 C 语言实现,具有很好的移植性。

1 Blowfish 算法原理描述

Blowfish 算法是一种对称的分组加密算法,该算法有一个核心加密函数:BF_En(),该函数的输入是 64 位明文信息,经过运算,以 64 位密文信息的形式输出[8]。用 Blowfish 算法加密信息,需要两个过程:密钥预处理和信息加密。详细介绍如下。

1.1 密钥预处理

子密钥由 Blowfish 计算得来:① 由 π 的十六进制字符串初始化 P 数组和 S 盒;② P1 和密钥的第一个 32bit 异或,P2 和第二个,以此类推。密钥循环使用,直到整个 P 数组都被替换;③ 用Blowfish 加密全 0 串,子密钥为第 2 步中描述的子密钥;④ 第3 步输出替换 P1 和 P2;⑤ 用 Blowfish 加密第 3 步输出,子密钥为修改之后的;⑥ 第 5 步输出替换 P3 和 P4;⑦ 重复这个过程,直到 P 数组被完全替换,然后用连续变换的 Blowfish 输出结果顺序替换 S 盒。Blowfish 算法的子密钥和 S 盒都是由算法

收稿日期:2012 - 06 - 14。**谢琦**,副教授,主研领域:嵌入式技术,无线传感器网络。**曹雪芹**,硕士生。

本身生成,密钥分析很困难[9,10]。

1.2　信息加解密

信息加密在程序中用 BlowfishEncipher( )实现,加密所用的明文被分成两半,即 XL 和 XR 各为 32bit,加密过程是一个 16 层的 Feistel 网[13],加密一个 64 位的明文需要迭代 16 次,在迭代的过程中使用上面生成的子密钥 key_pbox[i],最后生成 64 位的密文。

Blowfish 解密只需把 key_pbox[i]逆序使用即可。

2　Blowfish 算法优化及实现

Blowfish 算法涉及常量多、运算量大,需要高成本的硬件支持,并且本设计是基于 ZigBee 协议栈的通信平台,协议栈程序要占去一部分的 RAM 空间,必须对算法进行优化,才能在低速、仅有 8K RAM 空间的 WSN 节点上实现该算法[14]。

本文从 Blowfish 的算法原理入手,根据原理提出更简单的实现方法。这样会带来加密之后的密文输出和标准的不一样,但并不代表有逻辑错误,同样可以达到加密的目的。优化的原则是在一个可接受的时间范围内完成加密和解密过程,并且加密之后的密文是不可识别的,同时也是没有规律的。算法主要从以下 3 方面进行优化。

2.1　算法级的优化

分两部分进行,一是改密钥动态输入为静态给出,二是修改算法的 F 函数。

在 Blowfish 中,初始密钥是由用户动态输入的,这样在用户输入密钥之后,程序要对输入的数据长度进行判断,在本设计中,为了节省时间和空间开销,我们把 Key 直接初始化成 72 位的。当然,这个初始值可以根据我们的需要在程序中修改。如果我们给的初始值不是 72 位的,那么程序运行时就会执行对 key 进行扩充或者裁剪的语句,占用大量的 RAM 空间,而我们这样做可以很大地缩减加密部分所占的内存空间,并且不会影响加密数据的安全性,因为 Blowfish 算法的雪崩效应很好,即使知道明文和密文,也不能很快地破译出密钥,并且在文献[7]中已经有大量实验证明,改变密钥的长度不会影响 Blowfish 的安全性。

在 Blowfish 的 F 函数中,32 位的 XL 被分成了 4 部分,每部分 8 位,第一个 8 位选择 Sbox1 盒的入口,即用 8 个 2 进制数选择一个长度位 256 的数组,比如这 8 个 2 进制数是 0000 0011 的时候,我们就选择 Sbox1 盒数组中的第 4 个数据,当 2 进制数是 1111 1111 时,就选择 Sbox1 盒的最后一个 32 位数据。众所周知,在 Blowfish 算法中,S 盒数组所需空间为 256 × 4 × 4 = 4096B,在本设计中,RAM 仅为 8K,为了减少 S 盒所占的存储空间,我们采用下面方法:在密钥初始化时我们仅仅把 S 盒初始化成 S[4][64]的数组,为了不影响算法的正常执行,我们采用下面方法来实现用长度为 64 的数组满足 8 个 2 进制的数据域的范围,首先,判断 8 位二进制数(用 i 来表示,i 为 int 型)是多少,当 i 为 0 ~ 63 时,直接从 S1 盒的 64 个初始化常量中选择;当 i 为 64 ~ 127 时,对 i 进行模 64 的运算,运算结果即为 S1 盒的数组的下标值,然后将该单元中的 32 位数据循环左移一位,结果即为 S1 盒的 32 位输出;当 i 为 128 ~ 191 时,采用上面相同方法选择数组单元,然后将单元中的数据循环左移 2 位,结果为 S1 盒的 32 位输出,当 i 为 192 ~ 255 时,循环左移 3 位即可。对于 Sbox2、Sbox3、Sbox4,采用相同的方法实现。Sbox1 具体实现流程如图 1 所示。

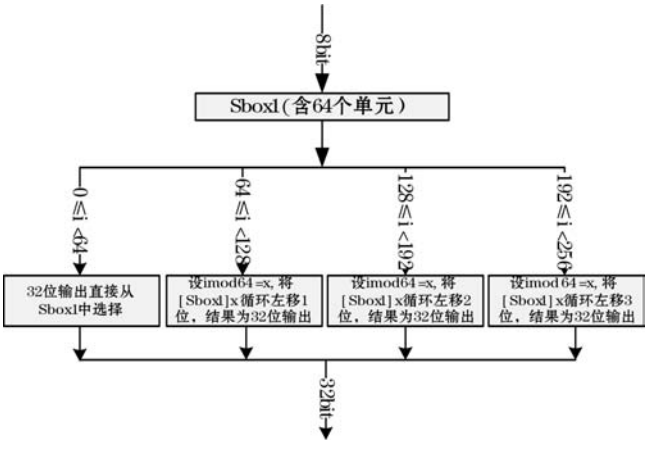

图 1　Sbox1 执行流程

至于初始化 S[4][64]数组里的每一行的 64 个元素,就涉及到了密钥库的选择问题。利用原来的 256 个元素借助于随机函数来产生 64 个真随机数,选择一个优良周期 T(T > >256),K[i]表示密钥库的任意元素,任意选择随机函数的种子和密钥库的起点生成含有 64 个元素的新序列。由于密钥库和随机函数的起点可任意选择,可组合出形形色色的 64 位随机数,使得相同 8 位的数选出的数组元素是不同的,保证了输入相同的密钥和明文产生不同的密文。

解密时,其他都不变,只需将循环左移改成循环右移即可。这样可以大大减少数据所占存储空间,修改之前 S 盒需要占 4096B 的存储空间,采用此方法修改之后,只需占用 64 × 4 × 4 = 1024B 的存储空间,虽然循环移位和判断语句会占一定的空间,但很少。并且,每个单元是 32 位的,循环移动 3 次之后,数据也不会出现重复,保证了 S 盒的安全性。

2.2　代码级的优化

算法的关键代码指的是程序中最耗时、占总运算量比重比较大的函数[12],对关键代码优化可事半功倍。通过 PC 机上的 C 语言模拟程序,用通用处理器运算计时中的 profile 工具统计程序各个部分所需时间占算法总耗时,得出 Blowfish 中最耗时的关键函数是 BlowfishKeyInit( )。

BlowfishKeyInit( )函数主要由 ADD 和 XOR 指令运算组成,大多数的运算操作是通过 C51 的汇编指令完成的。如在 ADD 函数中溢出控制的代码可以用一条带溢出控制的 ADDC 的指令实现。采用汇编语言修改关键部分代码可能是有效的方法之一,但是也可综合运用其他一些代码优化的方法,如编译器的优化编译选项设置,可选择最大优化级别等。我们也可将函数设置为内联函数,节省压栈和出栈的操作,但代码量可能会增加。如果需要,尽量使用自加、自减指令,此类指令可生成高质量的程序代码,编译器通常可生成 inc 和 dec 之类的指令。另外可改写算法中的循环语句以减少循环体内总的指令执行周期数,而对于多层嵌套循环,可将循环次数少的作为外循环,提高循环的执行效率;并且,实现同样的循环功能,for 循环的功耗比 while、go to 的要大, go to 语句的功耗最低。减少运算的强度,可使用运算量小但功能相同的表达式替换原来复杂的表达式,例如:将 ltemp = xl; xl = xr; xr = ltemp;改为 xl = xl + xr;xr = xl-xr;xl = xl- xr,替换之后,可实现交换数据的目的,并且可减少一个变量的开销。

2.3 针对硬件资源的优化

我们选用的 CC2530 单片机采用 8051 CPU,该 CPU 有 4 种不同的存储空间,分别是:CODE,DATA,XDATA,SFR,为了让 DMA 能访问到所有存储区,所有的存储器(包括 Flash 和 RAM、SFR)都映射到了 XDATA 上,所以,在单片机中实现加密算法时,如果程序所占空间超出了硬件要求的范围,会经常遇到 XDATA 溢出的问题。那么我们就根据这些限定的硬件条件,来对我们的算法进行局部的优化。

因为我们的初始密钥是以初始化的方式直接给定的,也就是密钥不会频繁的更换,在不手动更改初始密钥的情况下,子密钥每次生成的结果是一样的,所以,为了节省存储空间的开销,我们将生成的子密钥 key_pbox[ ]、key_sbox 和初始密钥 sbox[ ]、pbox[ ]放在同一个 Flash 存储空间,存储时需要根据 flash 每次的擦除大小控制分多少次写入,这样可以避免每次计算 512 次的生成子密钥,节省 RAM 空间的开销,提高算法效率。

再者,8051 CPU 可直接访问的存储空间只有 64K,为了访问 64k 以外的存储空间,CC2530 提出了映射的概念,CC2530F128 把 Flash 分成了 4 个 BANK,每个 bank 的大小是 32KB,通过设置 SFR 寄存器的对应位 MEMCTR. FMAP 控制不同编号的 bank 映射到 CODE 上,解决寻址空间受限问题。

3 算法性能分析

我们先在 PC 机上对改进加密算法的安全性进行验证。在加密算法没有优化之前,设定初始密钥为“12345678”,明文为“abdfdenm”时,在 Microsoft Visual C ++ 6. 0 下生成的密文如图 2 所示。

```
20ffffff9813ffffffd3777ffffff80ffffffd2
```

图 2 密文

进行多次输入相同明文,生成的密文都和上图显示是相同的,由于 Blowfish 算法的代码是公开的,那么,如果我们对算法不改进,敌人一旦获取密文,很容易就能得出明文。算法改进之后,我们使用相同密钥和明文,生成的密文如表 1 所示。

表 1 密文对比

明文	密文
abdfdenm	10fe35886d4c2aeb43dfe34ae504ffcd
abdfdenm	efbaa169b52c95278de3g523aade0048
abdfdenm	34fffa50d34aabb6998e42100ff2eabe
abdfdenm	feea403dae9c67940ddee20014d349ba

对比图 2 和表 1,不难看出,算法优化之后生成的密文和算法改进之前是不同的,并且算法优化之后的密文是没有规律可循的。再者,相同密钥和明文情况下,由于在加密过程中密钥库的选择不同,生成的密文也是不一样的,如表 1 所示。相对于改进之前每次生成密文都相同的算法,很明显改进之后的算法安全性能更好。并且,算法优化之前,所占空间大约 5KB,优化之后算法所占 RAM 空间少于 2KB,在 WSN 节点上加密的时间大约为 1s。

4 实验结果

WSN 节点选用的是 CC2530F128 的片上系统,具备在各种供电方式下数据保持的能力,支持硬件在线调试,系统存储结构如表 2 所示。

表 2 CC2530F128 系统存储结构

特征	CC2530F128
FLASH_SIZE	128KB
SRAM_SIZE	8KB
USB	不包括

WSN 网络选用星型拓扑结构,具体拓扑结构如图 3 所示。星型拓扑网络由一个 PAN 协调器的中央控制器和多个从设备组成,主协调器必须是 FFD 设备,从设备可以是 FFD 或者 RFD。在星状拓扑中,所有终端设备都与唯一的中央控制设备—PAN 协调器通信,终端设备之间的通信通过 PAN 协调器的转发来完成,终端设备之间不能直接进行通信。

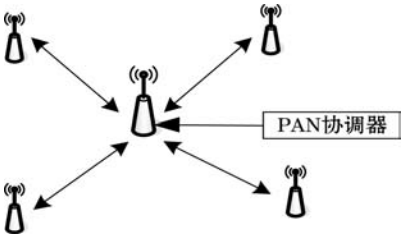

图 3 星状网络拓扑

图中只给出了一个协调器和四个节点设备的网络拓扑结构,当然节点设备可以根据需要进行加入和撤离,PAN 协调器是全功能设备(FFD),各个节点既可以是全功能设备(FFD)也可以是简化功能设备(RFD),箭头表示数据传输的方向,由图可知,协调器和节点之间可以在没有任何物理线路的情况下,通过节点的无线收发单元进行数据的发送和接收,各个终端节点之间不能互发数据。协调器之间也可通过无线网络进行通信,在此不做讨论。

我们用优化的 Blowfish 算法对 WSN 的 MAC 层的数据包进行加密,即对 MSDU 部分进行加密。MAC 层帧结构如图 4 所示。

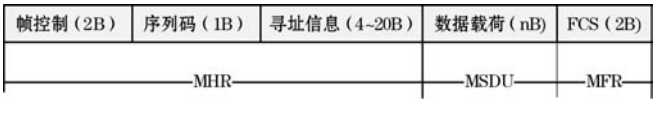

图 4 MAC 层数据帧结构

实验由 3 个节点完成无线环境区域内的星型组网连接,3 个节点硬件采用相同构成,都是全功能设备(FFD)。实验结果图如图 5 所示。

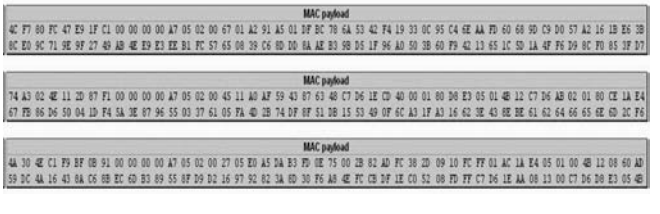

图 5 MAC 层数据包

图 5 是用 Sniffer 抓包工具对两个节点之间空中传输数据进行抓包的结果图,是在相同密钥和明文条件下的 3 次抓包结果,即密钥为“12345678”,明文为“abdfdenm”。通过图 5 可以看出 3 次抓包截图的数据部分是不一样的,并且是密文,即跟上节中验证的加密算法的安全性是一致的,从而达到了传输数据的保

(下转第 330 页)

应用具有高度的一致性:UI 设计一旦定型,Form、Controls、Table、Fields 等要素在设计业务的实现,而软件共性单元由元胞自动机托管。

(2) 规则明晰,提高业务专注度 FSA 通过抽象与业务解耦,实现基于表单、控件与数据模型的直接关联,开放式配置型的接口设计则可以普适业务逻辑扩展,从而尽最大可能的使程序员专注于自身差异化业务的实现,而软件共性单元由元胞自动机托管。

参 考 文 献

[1] 赵松年. 非线性科学—它的内容、方法和意义[M]. 北京:科学出版社,1994:69-76.

[2] Bastien C, Michel D. Cellular Automata Modeling of Physical Systems. (物理系统的元胞自动机模拟)[M]. 祝玉学,赵学龙,译. 北京:清华大学出版社,2003.

[3] 余亮,陈荣,何宜柱. 元胞自动机与经济学应用[J]. 系统工程,2003,21(1):90-93.

[4] Nagel K, Schreckenberg M. A cellular automaton model for freeway traffic[J]. J Phys I,1992,2(2):221-229.

[5] 宋卫国,范维澄,汪秉宏. 中国森林火灾的自组织临界性[J]. 科学通报,2001,6(1).

[6] 韩筱璞,周涛,汪秉宏. 基于元胞自动机的国家演化模型研究[J]. 复杂系统与复杂性科学,2004,1(4):74-78.

[7] 周恺卿,乐晓波,潘小海,等. 基于元胞自动机的线性遗传程序设计算法[J]. 计算机工程,2011,37(16):161-163.

[8] 贾斌,高自友,李克平,等. 基于元胞自动机的交通系统建模与模拟[M]. 北京:科学出版社,2007:48-54.

[9] 周成虎,孙站利,谢一春. 地理元胞自动机研究[M]. 北京:科学出版社,2001.

[10] 曹兴芹. 复杂系统的元胞自动机方法研究[D]. 武汉:华中科技大学,2005.

(上接第 320 页)

密要求。

5 结 语

本文提出的 Blowfish 算法的优化方案可在 WSN 节点上正常运行。此优化方案仅对算法的实现方法进行优化,不改变算法本身结构,因此没有降低算法的安全性。并且我们是用 C 语言来实现该算法,具有很好的移植性。ZigBee 协议栈本身有安全规范,但是需要在 AES 协处理器的支持下才能完成。本文通过算法的优化,大大缩减算法执行空间,最后在 RAM 空间很小的 WSN 节点上实现了该算法,达到数据的加密要求。至于 Blowfish 算法密钥的安全性,可结合 MD5 算法实现,但所需存储空间需增大,在此不做讨论。

参 考 文 献

[1] 钟黔川,朱清新. Blowfish 密码系统分析[J]. 计算机应用,2007,27(12):2940-2944.

[2] 李桂满,李国. 加解密算法 Blowfish 在单片机上的应用[M]. 单片机与嵌入式系统应用,2007(10):12-14.

[3] 尚华益,姚国祥,官全龙. 基于 Blowfish 和 MD5 的混合加密方案[J]. 计算机应用研究,2010,27(1):231-233.

[4] B Schneier. The Blowfish Encryption Algorithm[OL]. (2008-10-25). http://www.schneier.comlblowsh.html.

[5] 刘永平. 保密传真机 Blowfish 加解密算法的实现[J]. 信息与电脑,2010(10):102.

[6] 彭燕. 基于 ZigBee 的无线传感器网络研究[J]. 现代电子技术,2011,34(5):49-51.

[7] 李俊斌,胡永忠. 基于 CC2530 的 ZigBee 通信网络的应用设计[J]. 电子设计工程. 2011,19(16):108-111.

[8] Sindhuja A, Logeshwari R, ThirunadanaSikamaniA K. Secure PMS based on Fingerprint Authentication and Blowfish Cryptographic Algorithm[C]//2010 International Conference on Signal and Image Processing: 424-429.

[9] 张月华,张新贺,刘鸿雁. AES 算法优化及其在 ARM 上的实现[J]. 计算机应用,2011, 31(6):1539-1542.

[10] Allam Mousa. Data Encryption Performance Based on Blowfish[C]// 47th International Symposium ELMAR-2005:8-10.

[11] Harsh Kumar Verma, Ravindra Kumar Singh. Performance Analysis of RC5, Blowfish and DES Block Cipher Algorithms[J]. International Journal of Computer Applications (0975 - 8887),2012,42(16).

[12] Allam Mousa. Data Encryption Performance Based on Blowfish[C]// 47th International Symposium ELMAR-2005:8-10.

[13] 谭伟,马琪. 基于嵌入式 CPU 的 G. 722. 1 宽带语音编解码算法优化[J]. 机电工程,2010,27(11):71-74.

[14] 王智明. PLC 基于开关量实现模拟量输出的方法[J]. 机电工程,2009,26(5): 105-107.

(上接第 323 页)

中查找小字符集模式串的多模式匹配算法,即本文提出的改进算法,针对处理结果进行科学评价。

该改进算法充分利用了单模式匹配算法 MBMH 的快速移动,相比 AC 算法和 FS 算法减少了内存空间的占用,而在匹配速度和运行时间上与它们相比却有了显著的提高。最后,通过实验对其匹配速度的提高进行了分析,并通过开源网络入侵检测系统 Snort 对其性能进行了测试,证明了该算法很适合在具有大规模特征集的入侵检测系统中使用。

参 考 文 献

[1] 刘卫国,胡勇刚. DHSWM:一种改进的 WM 多模式匹配算法[J]. 中南大学学报:自然科学版,2011,42(12):3765-3771.

[2] 王培凤,李莉. 一种改进的多模式匹配算法在 Snort 中的应用[J]. 计算机科学,2012,39(2):72-74,79.

[3] Kanniya Raja N, Arulanandam K, Raja Rajeswari B, et al. Centralized Parallel form of Pattern Matching Algorithm in Packet Inspection by Efficient Utilization of Secondary Memory in Network Processor[J]. International Journal of Computer Applications ,2012,40(5).

[4] Liu Zaiqiang, Lin Dongdai, Guo Fengdeng, et al. A Method for Locating Digital Evidences with Outlier Detection Using Support Vector Machine [J]. International Journal of Network Security,2010,6(3).

[5] 刘云峰. 模式匹配及其改进算法在入侵检测系统中的应用[J]. 电脑开发与应用,2011,24(4):41-43.

[6] 舒银东. 基于有限状态自动机的多模式匹配算法研究[D]. 合肥工业大学,2011.

[7] Guinde N B, Ziavras S G. Efficient hardware support for pattern matching in network intrusion detection[J]. Computers Security,2010,29(7):756-769.