面向云存储的高性能数据隐私保护方法

孙辛未[1] 张 伟[1,2] 徐 涛[3]
(北京信息科技大学计算机学院 北京 100101)[1]
(北京信息科技大学网络文化与数字传播北京市重点实验室 北京 100101)[2]
(清华大学信息技术研究院微处理器与片上系统技术研究中心 北京 100084)[3]

摘 要 随着云计算和云存储技术的飞速发展,越来越多的企业和个人使用云存储来保存数据或备份数据。但用户将私有数据上传到云端的同时,也丧失了对数据的绝对控制权,用户数据的隐私保护问题成为云存储发展不得不解决的问题。为了解决这一问题,提出一种新的针对云存储的数据隐私保护方法BSBC(Bit Split Bit Combine),它在上传前,将数据按照比特位进行拆分,重新组装后形成多个数据文件,再分别上传到云存储服务器;下载时,先将所有数据文件下载,然后通过位合并再恢复成原始文件。实验证明这种方法可以保护用户数据的隐私,同时可比传统加解密获得17~35倍的性能提升。然后将核心的位拆分、位合并代码模块用汇编语言进行优化,对汇编语言进行指令调度优化,以减少数据冲突和流水线停顿。最终,采用BSBC方法比传统加解密可以获得25~35倍的性能提升。

关键词 云存储,数据安全,隐私保护,指令调度优化
中图法分类号 TP309.7 **文献标识码** A

High-performance Data Privacy Protection for Cloud

SUN Xin-wei[1] ZHANG Wei[1,2] XU Tao[3]
(School of Computer Science, Beijing Information Science and Technology University, Beijing 100101, China)[1]
(Beijing Key Laboratory of Internet Culture and Digital Dissemination Research, Beijing Information Science and Technology University, Beijing 100101, China)[2]
(Microprocessor and SoC Technology R&D Center, Tsinghua University, Beijing 100084, China)[3]

Abstract With the rapid development of technology of cloud computing and cloud storage, more and more businesses and individuals use cloud storage to store data or backup data. When uploading private data to the cloud, the user will lose the absolute control of the data, them data privacy protection becomes a problem that cloud storage has to solve. In order to solve this problem, BSBC (Bit Split Bit Combine), a new data privacy protection method was presented. Before uploading the data, BSBC splits the data according to bit and re-assembled to form a number of data files, then uploads the data to cloud storage servers; when downloading the data, BSBC downloads all the data files, then through the bit combination, revert them to the original file. Experiments show that this method can protect the privacy of users' data, obtain 17~35 times performance improvement compared with traditional encryption. Then assembly language is used is used to optimize the core codes of bit split and bit combination, and instruction scheduling optimization of assembly language to reduce the data conflict and pipeline stalls. Eventually, compared with traditional encryption, BSBC can get 25~35 times performance improvement.

Keywords Cloud storage, Data security, Privacy protection, Instruction scheduling optimization

1 概述

近年来,随着云计算[1]的不断发展,数据正以几何级数的方式增长。个人数据的存储规模已从原来的GB(1GB=2^{30}B)级上升到TB(1TB=2^{40}B)级,企业数据的存储规模从原来的PB(1PB=2^{50}B)级增长到现在的EB(1EB=2^{60}B)级[2]。为了管理和存储海量数据,存储行业推出了一种新的集中存储技术——云存储[3]。云存储的推广过程并非一帆风顺,根据Twinstrata公司2012年的云存储应用调查显示:只有20%的人愿意将自己的私有数据放在云存储中;相比之下,大约有50%的人愿意使用云存储来进行数据备份、归档存储以及灾难恢复等作业[4]。这主要是因为云存储商拥有所有存储数据

到稿日期:2013-07-08 返修日期:2013-11-04 本文受北京市教育委员会科技计划面上项目(KM201110772014),北京市优秀人才培养资助项目(2012D005007000009),北京市属高等学校创新团队建设与教师职业发展计划项目(IDHT20130519)资助。
孙辛未(1991-),女,硕士生,主要研究方向为存储安全、云存储,E-mail:sunweiwei05@gmail.com;张 伟(1980-),男,博士,讲师,主要研究方向为软硬件协同设计、存储安全、网络安全,E-mail:zhwei02@163.com(通信作者);徐 涛(1979-),男,博士生,主要研究方向为高性能存储系统设计。

的物理介质，所以云存储提供商拥有数据的优先访问权，而用户却丧失了对数据的绝对访问权[5]。

针对这一问题，目前许多云存储系统都有自己的安全策略，但用户很难完全信赖云存储服务商所提供的安全策略，因此目前很多用户采用本地加密的办法实现对数据的隐私保护。但加密算法的安全性主要依赖于密钥，一旦用户密钥丢失加密数据的安全性就无法保证。而且如果用户需要对大量文件进行加密，则会生成大量的密钥，如何对这些密钥进行高效的管理是一个很大的问题。

本文主要提出了一种新的、不依赖于密钥的高性能数据隐私保护方法——BSBC(Bit Split Bit Combine)数据隐私保护方案。通过位拆分的方式对原有的数据进行拆分，并将拆分好的数据上传到云端，实现对用户数据的隐私保护。用户想要使用原有的数据时，只要从云存储服务端把拆分好的数据下载下来，再经过相应的合并过程，就可以使用自己的数据了。接着，本文分别通过汇编语言编写核心代码以及调整代码顺序的方式对 BSBC 隐私保护技术的代码进行了优化，进一步提高了隐私保护技术的性能。

与传统的加密方法相比，BSBC 隐私保护技术既保证了用户数据的私密性又不需要对密钥进行管理操作，还大大缩短了对数据进行隐私保护的时间。

本文主要贡献在于：对用户存放在云端的数据提供隐私保护，让用户可以放心地将自己的数据上传到云存储平台。

2 相关工作

2.1 现有云存储系统隐私保护方式

目前主流的云存储平台主要有：Amazon S3、Dropbox、iCloud、Google Drive、Microsoft SkyDrive、SugarSync，以及国内的金山快盘、百度云盘等等。随着用户对数据隐私性的要求越来越高，为用户提供隐私保护技术的云存储平台也开始越来越多。Amazon S3[6,7] 和 Dropbox[8] 采用密钥程度为 256-bit 的 AES 加密算法为用户提供数据隐私保护服务；iCloud[9] 采用密钥长度为 128-bit 的 AES 加密算法为用户提供数据隐私保护服务；而国内推出的各种网盘暂时还不为用户提供加密机制来对用户的数据进行隐私保护。

主流云存储系统为用户提供的隐私保护方式很难取得用户的信任。首先用户的数据需要先上传到云存储系统中，再由系统为数据进行加密。这种方式虽然可以防止用户的数据被攻击者获取，但是由于使用云存储系统为用户的数据进行加密，云存储系统就拥有数据的密钥，对于用户来说自己的数据对于云存储系统是不涉密的。因此很少有用户会选择使用云存储系统所提供的加密方式对自己的数据进行隐私保护，而多选用现在主流的加密算法，在数据上传之前对数据进行隐私保护。

2.2 主流隐私保护技术

目前，随着用户对数据隐私性的要求越来越高，出现了许多的加密算法实现对数据的隐私保护。根据密钥的类型不同，通常把现代密码技术分为两大类：对称加密算法和非对称加密算法。

目前常用的对称加密算法有：DES[10]、3DES[11]、IDEA[12]、AES[13,14] 等。目前广泛应用的非对称加密算法有 RSA[15]、DSA[16]。

不论是对称加密算法还是非对称加密算法，都需要用户对密钥进行保护，一旦密钥泄漏加密数据的安全性就不能得到保证。此外，每对用户每次使用对称加密算法对数据进行加密时，都需要使用唯一的密钥对数据进行加密，这就会使得加密方和解密方所拥有的密钥数量呈几何级数的增长，密钥的管理工作将会成为用户的一大负担。如何保证密钥的安全性，以及如何对大量的密钥进行高效的管理是当前需要解决的一大问题。

3 BSBC 隐私保护技术

3.1 BSBC 数据隐私保护技术基本思想

3.1.1 位拆分(BS)技术核心思想

主流的数据加密算法通过移位和扩散的方式实现数据的隐私保护。本文继续延用移位和扩散的基本思想，设计出了位拆分技术对数据进行隐私保护。本文设计了 2 种位拆分方式，一种通过位拆分的方式将数据拆分成 2 份，分别存到 2 个新的文件中，然后再传到云端进行存储；另外一种是通过位拆分的方式将数据拆分成 4 份，分别存到 4 个新的文件中，再传到云端进行数据的存储。将文件拆分成越多份，攻击者拿到全部数据的可能性就越小，数据的隐私性也就更好。

将一份文件数据拆分成 2 份的过程如图 1 所示：

(1)读数据过程：从文件中读入 2 个字节的数据。

(2)拆分、移位过程：分别分离两个数据的高四位和低四位，并将高四位与低四位数据调换顺序。将原来的高四位变为低四位；原来的低四位变为高四位。

(3)重组过程：将第一个数据的高四位和第二个数据的低四位相加，重组成一个新的数据；将第一个数据的低四位和第二个数据的高四位相加，重组成一个新的数据。

(4)写数据过程：把得到的新的数据分别写入文件 1 和文件 2 中。

(5)判断是否到文件尾，如果未到文件尾，重复(1)—(5)的过程，如果已经到达文件尾则结束。

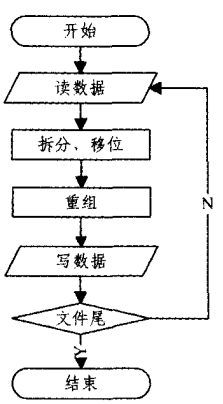

图 1 BS 技术流程图

这样本来一个字节的数据就被扩散到了文件 1 和文件 2 中，且位拆分后的数据顺序发生了变化，从而实现了数据隐私保护的作用。

将一份文件数据拆分成 4 份的过程与拆分成 2 份的过程类似，只是拆分的时候需一次对 4 个字节的数据进行操作，把拆分好的数据分别放入 4 个新文件内。

3.1.2 位合并(BC)技术核心思想

位合并技术是位拆分技术的逆过程，其目的是把拆分好

的数据恢复成原数据，使得加密者可以通过位合并的方式还原自己的数据。本文也针对将文件拆分成 2 份的位拆分技术和将文件拆分成 4 份的位拆分技术，分别设计出了合并 2 份文件的位合并技术和合并 4 份文件的位合并技术。位合并过程如图 2 所示，位合并过程可以恢复用户的数据，使得用户可以继续使用自己的数据。

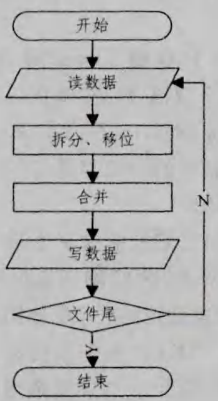

图 2　BC 技术流程图

3.2　BSBC 隐私保护技术实例

本文利用位拆分技术实现了对数据的隐私保护，并利用位合并技术实现了对保护数据的还原过程。本文以把文件拆分成 2 份的 BSBC 隐私保护技术为例，首先对存有英文内容的 TXT 文件进行位拆分，拆分前原文件内容如图 3 所示，本文件中存有英文内容，文件大小为 2.78kB。利用位拆分技术对文件进行隐私保护后，生成两个新的文件，文件内容如图 4 所示，两个文件的大小均为 1.39kB。两个文件的内容都是乱码，即使有攻击者获得这两份文件，也无法知晓原文件的内容，从而起到了数据隐私保护的作用。最后利用位合并的方式，对拆分后的两个文件进行合并。合并后与图 3 中的数据完全一致，文件大小也相同，保证了隐私保护前后数据的一致性。

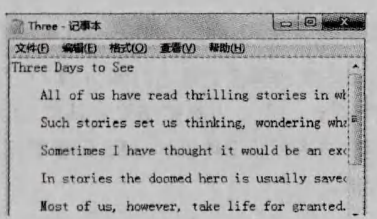

图 3　英文 TXT 文件

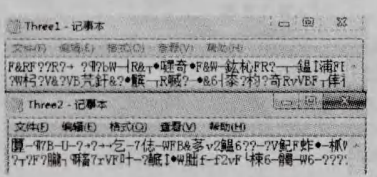

图 4　位拆分成 2 份后文件的内容

本文还通过 BSBC 隐私保护技术将非 TXT 文件拆分成 2 份，以 PDF 文档为例，原 PDF 文档大小为 1.58MB，原文件内容如图 5 所示，进行完位拆分后的文件内容如图 6 所示（拆分成 2 份后的文件大小均为 810kB），拆分后的文件内容无法显示，而合并后文件的内容与图 5 内容一致，且可以完整显示出文件的内容。

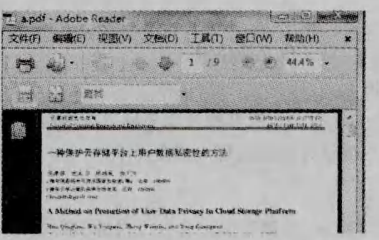

图 5　原 PDF 文档内容

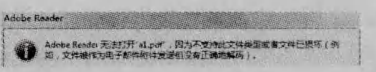

图 6　无法显示的拆分后的内容

用户可以通过位拆分的方式来保护数据的隐私性，可以通过位合并的方式来恢复原来的数据，且恢复后的数据与原数据内容一致，保证了数据的一致性。目前，可以为用户提供两种隐私保护方案，一种是通过位拆分的方式将文件拆分为 2 份，与之相对应的是利用位合并的方式合并 2 份文件；另一种是通过位拆分的方式将文件拆分成 4 份，与之相对应的位合并方式合并 4 份文件。两种隐私方案相比而言，分 4 份的隐私保护方案对文件的隐私保护强度更高。用户可以根据自己的实际需求，选择相应的隐私保护方案，对自己的数据进行隐私保护。

4　BSBC 隐私保护技术的实现及优化

4.1　C 语言实现 BSBC 隐私保护技术

本文用 C 语言实现的 BS(位拆分)隐私保护技术，将文件拆分成 2 份的伪代码如图 7 所示。

```
Void BS2C()//C 语言拆分成两份
{ a[i]=fread()//把文件内的数据读入数组 a
  d=a[i]&0F;//取出第 i 个数据的后 4 位
  e=a[i]&F0;//取出第 i 个数据的前 4 位
  f=a[i+1]&0F;//取出第 i+1 个数据的后 4 位
  g=a[i+1]&F0;//取出第 i+1 个数据的后 4 位
  d≪4;
  e≫4;
  f≪4;
  g≫4;//前 4 位后 4 位互换位置
  b[i/2]=d+g;
  c[i/2]=e+f;//重组好的数据分别存入数组 b,c 中
  fwrite(b);
  fwrite(c);//重组好的数据写入两个文件中
}
```

图 7　C 语言实现拆分成 2 份的 BS 技术

用 C 语言实现的 BC(位合并)隐私保护技术，将 2 份文件合并的伪代码如图 8 所示。

```
Void BC2C()//C 语言合并 2 份文件
{ b[i]=fread()//把文件 1 内的数据读入数组 b
  e[i]fread()//把文件 2 内的数据读入数组 c
  d=b[i]&0F;//取出 b 中第 i 个数据的后 4 位
  e=b[i]&F0;//取出 b 中第 i 个数据的前 4 位
  f=c[i]&0F;//取出 c 中第 i 个数据的后 4 位
  g=c[i]&F0;//取出 c 中第 i 个数据的前 4 位
  d≪4;
  e≫4;
```

```
f≪4;
g≫4;//前 4 位后 4 位互换位置
a[i * 2]=d+g;
a[i * 2+1]=e+f;//重组好的数据分别存入数组 a 中
fwrite(a);//重组好的数据写入新文件中
}
```

图 8　C 语言实现合并 2 份的 BC 技术

4.2　用汇编语言优化核心步骤

如果把之前用 C 语言编写的代码进行反汇编，会发现一条 C 语言代码基本上要对应 3 条左右的汇编代码，如图 9 所示，一般情况下汇编代码会比用 C 语言直接写代码的代码量要大一些，这主要是因为在汇编里需要考虑数据在寄存器中是如何操作的。但是本文的核心代码所做的操作基本上都是移位、与、或等较基本的运算，且这些操作使用汇编语言来编写代码，只需要一条代码就可以完成这些基本的运算操作。因此考虑直接采用汇编语言来编写核心代码，这样从代码行数上来比较基本可以缩短为原来代码行数的 1/3。因此本文对原来的 C 语言编码程序进行了优化，改用汇编语言来编写核心的过程。

```
01011875    mov     eax, dword ptr [ebp-200A8h]
0101187B    movzx   ecx, byte ptr [ebp+eax-1005Ch]
01011883    and     ecx, 0Fh
01011886    mov     byte ptr [ebp-20075h], cl
0101188C    movzx   eax, byte ptr [ebp-20075h]
01011893    shl     eax, 4
01011896    mov     byte ptr [ebp-20075h], al
0101189C    mov     eax, dword ptr [ebp-200A8h]
010118A2    movzx   ecx, byte ptr [ebp+eax-1005Ch]
010118AA    and     ecx, 0F0h
010118B0    mov     byte ptr [ebp-20081h], cl
010118B6    movzx   eax,byte ptr [ebp-20081h]
010118BD    sar     eax, 4
010118C0    mov     byte ptr [ebp-20081h], al
010118C6    mov     eax, dword ptr [ebp-200A8h]
010118CC    movzx   eax, byte ptr [ebp+eax-1005Eh]
010118D4    and     ecx,0Fh
010118D7    mov     byte ptr [ebp-2008Dh], cl
010118DD    movzx   eax, byte ptr [ebp-2008Dh]
010118E4    shl     eax, 4
010118E7    mov     byte ptr [ebp-2008D], al
010118ED    mov     eax, dword ptr [ebp-200A8h]
010118F3    movzx   ecx, byte ptr [ebp+eax-1005Eh]
010118FB    and     ecx, 0F0h
01011901    mov     byte ptr [ebp-20099h], cl
01011907    movzx   eax, byte ptr [ebp-20099h]
0101190E    sar     eax,4
01011911    mov     byte ptr [ebp-20099h], al
01011917    movzx   eax, byte ptr [ebp-20075]
0101191E    movzx   ecx,byte ptr [ebp-20099h]
01011925    add     eax, ecx
01011927    mov     edx, dword ptr [ebp-200B4h]
0101192D    mov     byte ptr [ebp+edx-18054h], al
```

图 9　C 语言位拆分的核心代码反汇编结果

4.3　通过指令调度优化性能

在用汇编编写核心代码的时候，由于对寄存器的操作存在写入数据后马上就需要读取寄存器内容的情况，根据流水线的运行原理，写后读会造成流水线停顿一个时钟周期，如果要对大量的数据进行位拆分或位合并过程，每次拆分、合并的时候都需要停顿一个时钟周期，这一个时钟周期将带来大量时间的浪费。因此本文通过调整代码顺序的方式来解决目前存在的数据冲突问题，并且在代码中尽量采用立即数进行寻址，而不转换成寄存器寻址。

4.4　性能测试

测试环境为：一台联想 ThinkPad SL300 笔记本电脑，CPU 为因特尔酷睿 2 双核 T5870 2.00GHz，内存为 2.00GB。本文分别使用 1M，9.98M，100M，293M，503M，763M 的数据进行测试。本文所显示的时间都是每一个测试数据测试 10 次后取平均时间的结果。

4.4.1　C 语言实现 BSBC 隐私保护技术

本文用 C 语言实现 BSBC 隐私保护技术，与现在普遍认为安全性较高、加密速度较快且广泛使用的 AES 加密算法进行加密时间的对比，AES 的密钥长度取 128 比特。分别用 AES 加密算法和 BS(位拆分)技术对数据进行隐私保护，结果如表 1 所列。从图 10 中可以看出，与传统的加密算法相比，BS 隐私保护技术的性能平均提升了 9～14 倍。

表 1　AES 加密时间与 BS 技术拆分时间(单位 s)

	AES 加密时间	C 语言分 2 份时间	C 语言分 4 份时间
1M	0.173	0.02	0.034
9.98M	1.829	0.098	0.159
100M	18.588	0.841	1.561
293M	50.42	2.755	4.628
503M	82.299	4.474	8.217
763M	128.595	6.974	13.623

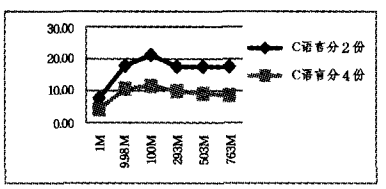

图 10　相比 AES 的加密时间，性能提升图

本文用 AES 加密算法的解密时间与 BC(位合并)技术的合并时间进行对比。表 2 为测试结果。从图 11 可以看出使用 BC(位合并)技术相较于传统的 AES 解密时间性能平均提升了 20～35 倍。

表 2　AES 解密时间与 BC(位合并)技术拆分时间(单位 s)

	AES 解密时间	C 语言合 2 份时间	C 语言合 4 份时间
1M	0.381	0.02	0.031
9.98M	3.974	0.099	0.169
100M	39.955	0.818	1.558
293M	107.019	2.631	4.583
503M	179.189	6.457	8.064
763M	274.266	9.694	14.819

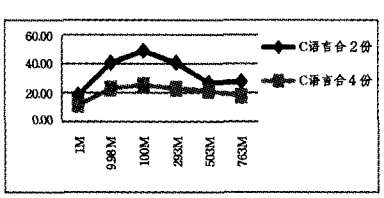

图 11　相比 AES 的解密时间，性能提升图

从C语言实现的BSBC隐私保护技术的BS(位拆分)技术、BC(位合并)技术时间可以看出,与现在使用最多的主流AES加密算法的加密、解密时间相比,BSBC隐私保护技术具有明显优势。因此如果用户需要对大量的数据进行加密,且需要快速地对数据进行加密的时候,使用BSBC数据隐私保护技术来对数据进行拆分和合并,从而达到数据隐私保护的目的是一个不错的选择。

4.4.2 用汇编语言优化核心步骤

直接使用汇编语言所编写的代码,比用C语言编写的代码反汇编后的结果缩短了2/3的代码量。针对这一优化策略,本文对用汇编语言编写后的拆分时间与原来用C语言编写的程序的拆分时间进行了对比,通过拆分的时间来看使用汇编语言进行编写程序后是否使得程序的拆分速度有所提升。测试结果如表3所列。从图12中可以看出,使用汇编语言编写核心代码后拆分时间较之前C语言程序平均提升了0.34～1.2倍。

表3 C语言位拆分时间与汇编位拆分时间对比(单位s)

	C语言分2份时间	汇编分2份时间	C语言分4份时间	汇编分4份时间
1M	0.02	0.015	0.034	0.017
9.98M	0.098	0.074	0.159	0.087
100M	0.841	0.606	1.561	0.665
293M	2.755	1.842	4.628	1.851
503M	4.474	3.228	8.217	3.479
763M	6.974	5.43	13.623	5.642

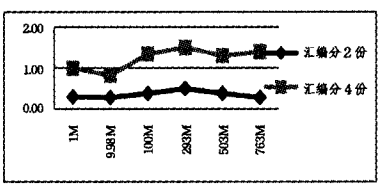

图12 C语言位拆分时间与汇编位拆分时间对比图

同理我们对位合并程序也用汇编语言进行编写,如表4所列。从图13中可以看出,使用汇编语言后合并时间平均提升了0.21～0.58倍。

表4 C语言位合并时间与汇编位合并时间(单位s)

	C语言合2份时间	汇编合2份时间	C语言合4份时间	汇编合4份时间
1M	0.02	0.015	0.031	0.02
9.98M	0.099	0.075	0.169	0.1
100M	0.818	0.628	1.558	0.735
293M	2.631	2.483	4.583	2.671
503M	6.457	5.554	8.064	6.82
763M	9.694	9.055	14.819	11.472

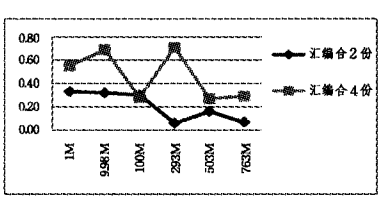

图13 C语言位合并时间与汇编位合并时间对比图

4.4.3 通过指令调度优化性能

本文将通过指令调度优化的位拆分时间和为进行指令调度优化的位拆分时间进行对比,如表5所列。从图14中可以看出,进行指令调度优化后性能平均提升了0.05～0.4倍。

表5 调整代码顺序前后拆分时间(单位s)

	汇编分2份时间	优化后分2份时间	汇编分4份时间	优化后分4份时间
1M	0.02	0.012	0.015	0.014
9.98M	0.098	0.07	0.074	0.07
100M	0.841	0.559	0.606	0.569
293M	2.755	1.744	1.842	1.757
503M	4.474	3.179	3.228	3.205
763M	6.974	4.746	5.43	4.939

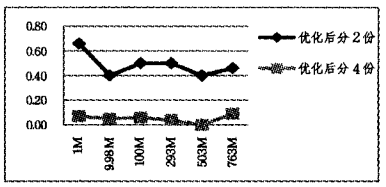

图14 指令调度优化后位拆分时间提升图

本文用同样的方法对解码程序的代码顺序进行了调整,调整后解码时间如表6所列。从图15中可以看出,进行指令调度优化后合并时间平均提升了0.17～0.22倍。

表6 指令调度优化前后合并时间(单位s)

	汇编合2份时间	优化后合2份时间	汇编合4份时间	优化后合4份时间
1M	0.015	0.015	0.02	0.018
9.98M	0.075	0.073	0.1	0.09
100M	0.628	0.599	0.935	0.725
293M	2.483	1.683	2.671	2.021
503M	5.554	3.008	6.32	4.703
763M	9.055	7.847	11.472	9.772

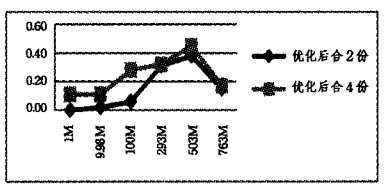

图15 指令调度优化后位合并时间提升图

经过优化后BSBC数据隐私技术位拆分时间与AES加密算法的加密时间如表7所列,可以看出,优化后的位拆分时间较AES加密算法的加密时间有了很大的优化,如图16所示,BSBC数据隐私保护技术拆分数据的时间是AES加密算法加密时间的25倍。如果要处理大量数据,BSBC数据隐私保护技术的位拆分方式具有明显优势。

表7 优化后拆分时间与AES加密时间(单位s)

	AES加密时间	优化后分2份时间	优化后分4份时间
1M	0.173	0.012	0.014
9.98M	1.829	0.07	0.07
100M	18.588	0.559	0.569
293M	50.42	1.744	1.757
503M	82.299	3.179	3.205
763M	128.595	4.746	4.939

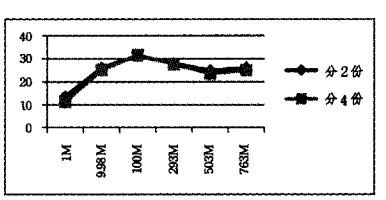

图16 优化后拆分时间与AES加密时间对比提升图

经过优化后 BSBC 数据隐私技术位合并时间与 AES 加密算法的解密时间如表 8 所列。

表 8　优化后位合并时间与 AES 解密时间(单位 s)

	AES 加密时间	优化后合 2 份时间	优化后合 4 份时间
1M	0.381	0.015	0.018
9.98M	3.974	0.073	0.09
100M	39.955	0.599	0.725
293M	107.019	1.883	2.021
503M	179.189	4.008	4.703
763M	274.266	7.847	9.772

从表 8 中可以看出,优化后的位合并时间较 AES 加密算法的加密时间有了很大的优化,如图 17 所示。图 17 所表示的是 BSBC 相较于传统的 AES 加密算法其加密性能提升了多少倍,BSBC 数据隐私保护技术合并数据的时间是 AES 加密算法加密时间的 35 倍。如果要处理大量数据,BSBC 数据隐私保护技术的位合并方式具有明显优势。

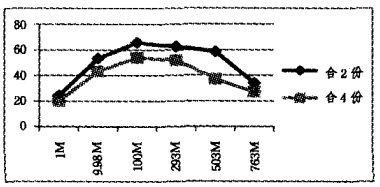

图 17　优化后位合并时间与 AES 解密时间对比提升图

结束语　本文中描述了一种不依赖于原有加密算法的数据隐私保护方式对用户的数据进行隐私保护工作。传统的加密算法对数据进行加密时主要依赖于密钥,一旦密钥丢失,用户的数据就不能得到恢复。而且当用户需要对大量的数据进行隐私保护,或用户数据的版本要经常修改的时候,传统的加密算法会因此产生大量的密钥,如何有效地、高效地对这些密钥进行管理是用户必须考虑的问题。针对以上问题,本文提出利用位拆分方式实现对数据的重新编码,把数据分成多份,再把分离好的数据分别传到云存储平台上,从而实现对用户数据的隐私保护。用户想要查看原数据时,只需从云存储平台分别下载分离后的数据,再通过相应的位合并方式,实现对数据的解码,从而恢复出原数据。

用 C 语言实现了 BSBC 隐私保护技术后,又针对 BSBC 隐私保护技术主要进行移位和与、或操作的特点,在原有 C 语言实现的 BSBC 隐私保护技术的基础上,本文核心代码利用汇编语言进行编程,加快了数据拆分和合并的速度。随后根据计算机指令调度的原理对汇编语言的指令顺序做了调整,从而优化了原有的 BSBC 隐私保护技术。从实验结果中可以看出,BSBC 隐私保护技术相对于传统的加密算法在加密时间和解密时间上有大幅度的提升。

今后,还可以通过多线程的方式进一步提升位拆分、位合并的速度,并把 BSBC 数据隐私保护技术应用到移动终端,为用户移动终端上的数据提供隐私保护。

参 考 文 献

[1] Cloud storage [EB/OL]. http://en.wikipedia.org/wiki/Cloud_stroage,2012-5-10

[2] 傅颖勋,罗圣美,舒继武.安全云存储系统与关键技术综述[J].计算机研究与发展,2013,50(1)

[3] Cloud computing [EB/OL]. http://en.wikipedia.org/wiki/Cloud_computing,2012-5-10

[4] Twinstrara [EB/OL]. http://twinstrara.com,2012-05-10

[5] 侯清铧,武永卫,郑纬民,等.一种保护云存储平台上用户数据私密性的方法[J].计算机研究与发展,2011,48(7)

[6] Amazon simple storage service [EB/OL]. http://aws.amazon.com/s3,2012-05-10

[7] Using Data Encryption [EB/OL]. http://docs.aws.amazon.com/AmazonS3/latest/dev/UsingEncryption.html,2006-3-1

[8] Shraer A,Cachin C,Cidon A,et al. Venus: Verification for untrusted cloud storage [C]// Proc of the 2010 ACM Workshop on Cloud Computing Security Workshop. New York: ACM,2010: 19-30

[9] iCloud:iCloud 安全性与隐私政策概览[EB/OL]. http://support.apple.com/kb/HT4865? viewlocale=zh_CN&locale=zh_CN,2013-2-11

[10] Alani D M M. DES96-Improved DES Security [C]// 2010 7th International Multi-Conference on Systems Signals and Devices (SSD). Amman,2010:1-4

[11] Shao Jun-xiang, He Zhi-min. High-speed implementation of 3DES encryption algorithm based on FPGA[C]// Modern electronic technology. 2004

[12] Kelsey J,Schneier B,Wagner D. Key Schedule Cryptanalysis of IDEA,G-DES,Gost,Softer and Triple DES[M]. Springer Verlag,1997

[13] NIST Advanced Encryption Standard (AES)[OL]. Development Effort web site http://csrs.nist.gov/encryption/aes/aes-home.htm

[14] Daemen J, Rijmen V. AES Proposal: Rijndael Version 2[EB/OL]. http://www.east.kuleuven.ac.be/~ rijmen/rijndael, 1999-10-05

[15] Rivest R,Shamir A,Aldeman L. A Methoed for Obtaining DigitalSignatures and Public-key Cryptosystems[J]. J. Communications of the ACM,1978,21(2):120-126

[16] Shimizu Y, Nuno F. Performance Evaluation of Novel DSA Scheme that combines Polling Method with Random Access Method[C]//PIMRC' 06. Helsinki,Finland,Sept. 2006

(上接第 132 页)

[11] Ma G,Pei Q,Wang Y,et al. A General Sharing Model Based on Proxy Re-encryption[C]// 2011 Seventh International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP). IEEE,2011:248-251

[12] Feng X,Tang Z,Yu Y Y. An efficient contents sharing method for DRM [C] // Consumer Communications and Networking Conference,2009(CCNC 2009),6th IEEE. IEEE,2009:1-5

[13] Lee S,Kim J,Hong S J. Redistributing time-based rights between consumer devices for content sharing in DRM system[J]. International Journal of Information Security,2009,8(4):263-273