基于 DES 加解密技术的电子文档访问控制方法①

黄林昊[1], 江　晨[2], 金　彪[2]

[1](福建广播电视大学 电子信息与计算机系, 福州 350007)
[2](福建师范大学 软件学院, 福州 350007)

摘　要: 电子文档的在线阅览已经非常普遍. 付费用户可以下载文档或在线阅读文档的全部内容, 而普通用户则只允许预览有限的内容. 通过这种方式可以在一定程度上实现对电子文档使用范围的控制. 但该方式的不足在于, 付费用户下载文档后, 可以随意将下载的文档转发给他人. 因而, 如何在文档被下载后依然能对其使用范围进行限制, 从而更好地实现版权保护, 值得研究和摸索. 本课题重点研究电子文档的线上线下访问控制方法: 通过对注册用户权限的限制实现电子文档的线上访问控制, 基于 DES 加解密等技术控制下载文档的线下使用范围. 实验结果表明, 本文所提出的基于 DES 加解密技术的电子文档线上线下访问控制方法, 能够更加有效地对电子文档的访问权限进行控制, 具有更强的版权保护力度.

关键词: 电子文档; 线上线下; 访问控制; DES 加解密; 版权保护

Method for Electronic Documents' Access Control Based on DES

HUANG Lin-Hao[1], JIANG Chen[2], JIN Biao[2]

[1](Department of Electronic Information and Computer Science, The Open University of Fujian, Fuzhou 350007, China)
[2](Faculty of Software, Fujian Normal University, Fuzhou 350007, China)

Abstract: Online reading of electronic documents has been very common. Documents could be downloaded or their full content could be read online by those users who have paid for them, while limited content could be previewed by the common users. By this way, documents' owners can realize the control of the use of electronic documents to a certain extent. But there are imperfections in the way. If one user has pay for the documents and downloaded them, he or she could send them to others without any restriction. Therefore, in order to achieve better copyright protection, it is worthy of study and exploring access control method on how to still make some restrictions on documents when they are downloaded. This paper focuses on the method for making some access restrictions on the electronic documents both online and offline. For online documents, access restrictions would be done on registered users, while DES encryption and decryption is used for controlling the downloaded documents. Experimentations show that the proposed method based on DES encryption and decryption technology could control the access to electronic documents more efficiently and give a stronger force on copyright protection.

Key words: electronic documents; online and offline; access control; DES encryption and decryption; copyright protection

随着互联网技术日益发展和计算机的普及, 越来越多的互联网用户愿意在开放平台上发表小说、教程、论文等电子文档并允许其他指定的用户群体进行访问, 即实现一定范围内的信息共享. 然而, 信息共享大背景下, 必然产生访问控制和版权保护问题. 某些特殊文档的制作者或发表者通常仅允许小范围的局部共享而非完全公开. 遗憾的是, 一旦文档所有者将文档上传至各大分享平台, 如百度文库、微盘等, 文档的所有

① 基金项目:省教育厅 A 类资助项目(JA14091,JA14087)
收稿时间:2016-08-16;收到修改稿时间:2016-09-18　[doi:10.15888/j.cnki.csa.005731]

权和控制权即被分离.

为了实现对电子文档的访问控制, 目前许多文库网站主要通过要求用户购买虚拟货币(或其它付费手段)的方式来控制文档的使用范围. 付费用户即可下载相应的文档. 这类做法可以在一定程度上对文档的使用范围进行控制, 然一旦付费用户将文档下载本地后, 该文档即可被随意复制和转发. 以国内大型在线文档分享网站百度文库为例, 其更多的是在线上对用户操作进行限制, 即在提供给用户在线查看的同时, 限制用户线上的阅读页数, 用户使用虚拟货币(上传文档或者其他活动可以赚取货币财富值)购买文档后可以下载文档. 但是其对下载后的文档并没有进行严密的线下文件保护[1].

也有相当一部分的网站已经开始加强版权保护. 国外的大型电商网站亚马逊, 在对待版权问题上相当慎重. 用户购买的书籍必须是官方书店中的正版 modi 或者 PDF 格式, 而且需要将书籍导入到 kindle 中才可以阅览, 无法将书籍复制或者转发给他人看[2].

为了更有效地保护文档所有者的权益, 更有力地实现版权保护, 针对电子文档线上线下访问控制研究就显得尤为重要. 本文结合 DES 加密技术, 设计一种电子文档访问控制方法, 实现了一套电子文档分享及访问控制平台(为了便于后续描述, 将该控制平台简称为 EDSC 平台), 旨在同时实现对电子文档的线上线下访问控制.

1 相关技术介绍

1.1 文件格式转换

目前已有的支持在线阅读的文库网站, 其主要支持的书籍和文档格式为 Word、Txt、PDF 三大类.

三者相比而言, PDF 格式更具优势: 首先, 用户无法随意更改 PDF 格式文件的内容, 因此可以更为有效的保护文档内容; 其次, PDF 文件拥有更好的用户阅读体验; 再者, PDF 文件更加易于拆分, 从而能够更加有效地限定在线预览的范围(页数). 鉴于上述原因, 本系统决定在线上预览采用 PDF 格式, 用户上传的文档无论是 txt 或 doc 格式的文档都将在后台被自动转换为 PDF 格式, 并进行分页拆分.

Word 文件转换成 pdf 文件可以通过调用 Document 对象的 ExportAsFixedFormat 方法实现, 调用方式为 Dispatch.call(doc, "ExportAsFixedFormat", pdfFullPath, wdFormatPDF); txt 文件转换成 pdf 较前者稍显复杂, 核心代码如下.

txt 文件转 pdf 文件的核心代码

```
//设置 pdf 格式排版
Document document = new Document(PageSize.A4, 80, 80, 60, 30);
 PdfWriter.getInstance(document, new FileOutputStream(pdfFilePath));
 document.open();
 //设置输入字体
 BaseFont bfChinese = BaseFont.createFont("STSong-Light", "UniGB-UCS2-H", BaseFont.NOT_ EMBEDDED) ;
 Font fontChinese = new Font(bfChinese, 18, Font. NORMAL);
 //添加段落
 Paragraph t = new Paragraph();        t.setAlignment(Element.ALIGN_CENTER);
 t.setLeading(30.0f);              document.add(t);
 fontChinese = new Font(bfChinese, 11, Font. NORMAL);
 //读取 txt 文本内容
BufferedReader read = null;
read = new BufferedReader(new FileReader (txtFilePath));
……
//写入 PDF 文件中
t = new Paragraph(line, fontChinese);
t.setAlignment(Element.ALIGN_LEFT);
t.setLeading(20.0f);
document.add(t);       ……
```

通过判断用户是否购买进而赋予不同的操作权限, 购买用户可以在线预览整个文件的内容, 而没有未购买的用户则只能预览部分限定的内容.

1.2 基于客户端解密阅读方式的实现分析

DES 加解密[3]是一种对称加密算法, 具有较高的安全性, 其有三个入口参数, 即 Key、Data 和 Mode, 其中 Key 和 Data 均为 64 位, 分别是 DES 算法的工作密钥以及要被加密或被解密的数据, Mode 为 DES 的工作方式(加密或解密). 该算法使用 64 位密钥将 64 位的明文输入块转换成 64 位的密文输出块, 并把输出分为 L0 和 R0 两部分, 每部分各长 32 位.

文献[4]将 DES 算法的实现与文件加密相结合, 通过应用程序选择并且导入文件从而获得文件流, 接着将载入的文件流利用 DES 算法进行加密处理, 最后得到整个文件的加密输出. 在打开文件时, 如果导入的是加密文件流, 则使用 DES 解密算法进行解密得到明文. 文献[5]的研究表明, 以客户端程序的形式对文件进行解密操作是可行的.

本文使用 Java SDK 中封装好的相关库函数以及自行编写的相关用户函数, 实现了 DES 加解密算法, 核心代码如下所示. 电子文档被下载时会利用 DES 加密算法对其进行加密处理, 下载得到的加密文件在客户端打开时, 则相应地使用 DES 解密算法进行解密. 同时, 为了防止用户把解密得到的文档内容复制到他人计算机或者将自己的账号密码借用给他人, 客户端程序将对用户计算机的 Mac 地址进行绑定和校验.

DES 加密核心代码

```
    // 根据参数生成 Key
    public void getKey(String strKey){
    ……
     KeyGenerator _generator = KeyGenerator.getInstance("DES");
     _generator.init(new SecureRandom(strKey.getBytes()));
    this.key = _generator.generateKey();
     _generator = null;
     ……
}
//对文件进行加密并保存目标文件 destFile 中
public void encrypt(String file,String desFile)throws Exception{
    //将加密码转换成 UTF-8 格式
    DESKeySpec desKeySpec = new DESKeySpec (passKey.getBytes("UTF-8"));
    //DES 加密字节位数选择
     Cipher cipher = Cipher.getInstance ("DES/CBC/PKCS5Padding");
     SecretKeyFactory keyFactory = SecretKeyFactory.getInstance("DES");
    SecretKey secretKey = keyFactory.generateSecret (desKeySpec);
    IvParameterSpec iv = new IvParameterSpec(passKey.getBytes());
    //选择加密模式
    cipher.init(Cipher.ENCRYPT_MODE, secretKey, iv);
     InputStream is = new FileInputStream(file);
     OutputStream out = new FileOutputStream(desFile);
     CipherInputStream cis = new CipherInputStream(is, cipher);
     byte[] buffer = new byte[1024];       int r;
     //对文件流进行加密处理
     while ((r=cis.read(buffer))>0) out.write(buffer,0,r);}
     ……
 }
```

客户端 DES 解密核心代码

```
public static void DesDecrypt(string m_InFilePath, string m_OutFilePath, string sDecrKey)
    {
      DESCryptoServiceProvider des = new DESCryptoServiceProvider();
      //转换编码格式
      des.Key = ASCIIEncoding.ASCII.GetBytes (sDecrKey);
      des.IV = ASCIIEncoding.ASCII.GetBytes (sDecrKey);
      FileStream fin = new FileStream(m_InFilePath, FileMode.Open, FileAccess.Read);
    FileStream fout = new FileStream(m_OutFilePath, FileMode.OpenOrCreate, FileAccess.Write);
    fout.SetLength(0);
    byte[] bin = new byte[1024];     long rdlen = 0;          long totlen = fin.Length;          int len;
    //解密模式选择
    CryptoStream encStream=new CryptoStream (fout,des.CreateDecryptor(), CryptoStreamMode.Write);
    while (rdlen < totlen) {//读入文件信息
    len = fin.Read(bin, 0, 1000);   encStream.Write (bin, 0, len);     rdlen = rdlen + len;
    }
     ……
}
```

2 EDSC平台概述

平台设计与实现过程中主要涉及到对 Java SDK、MyEclipse、Visual Studio 2012、HTML+CSS[6-9]等软件或技术的使用.

EDSC 平台主要由用于线上访问控制的 web 平台以及用于线下访问控制的客户端软件构成. 线上文件保护的 Web 平台主要功能包括: 允许文档拥有者以对上传文档设置是否免费或标价; 对上传的文档进行文档预览范围的限制; 对上传文件进行格式转换(word 格式转为 PDF), 进而提供在线预览; 对被下载的文件进行加密处理; 用户可以收藏和评论电子文档. 线下文件保护的客户端软件主要实现功能包括: 对用户账号密码进行校验; 向服务端提供用户的 MAC 地址, 保证阅读文档机器的唯一性, 防止复制和传输文档导致的侵权问题; 对从 web 平台下载的文档进行解密处理, 并且在客户端上显示. 图 1 为 EDSC 平台的体系结构图.

值得说明的是, 若考虑硬件设备投入成本的限制, 方案中的加密服务器以及解密服务器所执行的操作, 可以分别集成到 web 服务端与线下客户端.

对于注册且已付费用户而言, 其可以选择在线阅读文档的全部内容, 亦可以选择下载文档; 对于未注册或已注册但未付费的用户而言, 其只被允许在线预览文档所有者事先设定的部分内容. 表 1 简述了 EDSC 平台的核心功能模块, 图 2 为 EDSC 平台操作的整体数据流图.

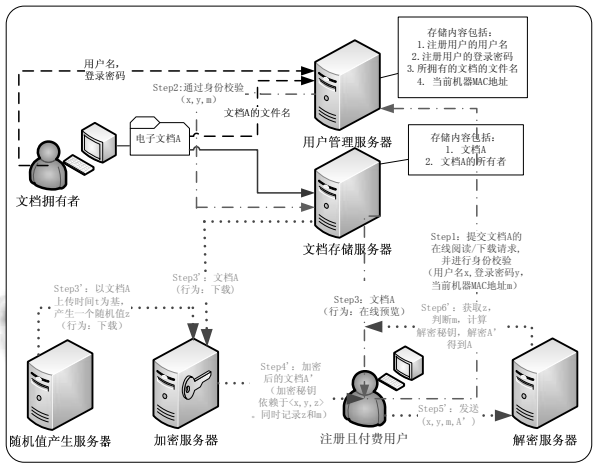

图 1 EDSC 平台体系结构图

表 1 EDSC 平台核心功能模块简述

序号	模块名称	功能简述
1	电子文档上传	文档拥有者将文档上传至线上 web 平台, 并对文档进行描述、标价以及预览范围限定等
2	下载加密	对用户下载的文件进行加密处理, 加密秘钥唯一且与用户账号相关
3	线下客户端解密	对从 web 端下载的加密文档进行解密处理(只有同时通过身份校验和 MAC 校验后才可以进行正常解密)

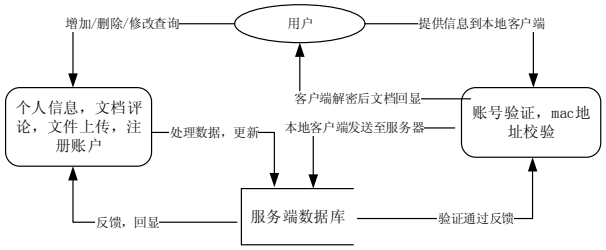

图 2　EDSC 平台操作数据流图

文档被下载至本地计算机时, 当前下载用户的相关信息(用户名、计算机 MAC 地址等)以及文档加密时所采用的随机值均会被记录. 其目的在于, 便于以后的解密和身份校验. 当解密服务器集成到线下客户端时, 可以保证被下载的文档只能由本平台的客户端进行解密和显示.

3　EDSC平台核心功能设计

3.1 保证秘钥唯一性

对于任何一种加解密算法而言, 秘钥都是十分重要的. 如图 1 所示, 本文在使用 DES 算法时, 以 (x,y,z) 的组合键来生成秘钥. 其中, x 表示用户名(本平台不允许同一个用户名重复注册), y 为用户密码, z 则是一个以文档被下载时间 t 为基数产生的随机值.

用户名的唯一性, 再加上随机值 z 的参与, 不仅使得不同用户的秘钥不一样, 而且还能确保同一个用户的不同文档的操作秘钥也不相同, 进而可以更加有效地保证了加(解)密秘钥的唯一性. 获取时间 t 并以此为基数产生随机数的核心代码如下所示.

```
Calendar c = Calendar.getInstance();
        int y = c.get(Calendar.YEAR);
        int m = c.get(Calendar.MONTH);
        int d = c.get(Calendar.DATE);
        int hr = c.get(Calendar.HOUR_OF_DAY);
        int mi = c.get(Calendar.MINUTE);
        int se = c.get(Calendar.SECOND);
        intz=(int)(Math.random()*(y+m+d)*hr*mi*se);
```

此后, (x,y,z) 将被组合成一个字符串, 传递给自定义函数 getKey(), 最终产生加(解)密秘钥.

3.2 电子文档上传

文档拥有者可以将自己的作品上传到 web 平台, 并对自己的作品进行描述和价格标定. Web 平台对上传的文件格式进行限制, 目前只支持 txt、word 和 pdf 格式. 图 3 为该功能模块的时序图.

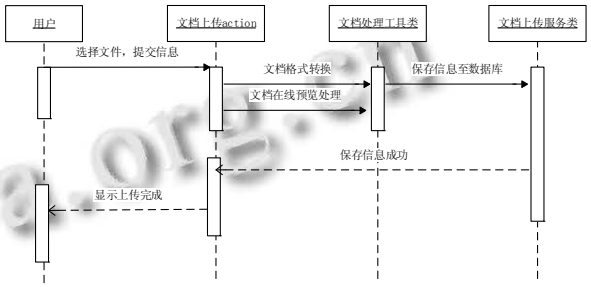

图 3　电子文档上传时序图

用户进入上传界面, 选择需要上传的文件并填写文件相关信息后, 进行提交. 此后, 前台数据会通过 struts2 的文件上传模块传入电子文档信息处理类, 由处理类负责把用户的输入信息利用 hibernate 进行封装后, 通过数据库接口将文档信息存入 mysql 数据库, 同时将上传文件保存在服务端的指定位置. 如前文所述, 文件上传成功后, 平台会自动调用文件格式转换函数, 将用户所上传的非 PDF 格式文档转换成相应的 PDF 格式[10].

此外, 为了赋予付费用户与未付费用户不同的在线阅读范围, 转换得到的 PDF 文件还会被进行裁剪分割操作, 得到两份不同的文件, 一份用于全部内容的在线预览(针对付费用户), 另一份则用于免费预览限定的内容(针对未付费用户, 具体页数可有文档拥有者在上传文档的同时加以指定). PDF 文件拆分的核心代码如下所示, 图 4 则为文档在线预览的效果图.

```
public static void splitPDF(InputStream inputStream, OutputStream outputStream, int sPage, int ePage) {
        Document document = new Document();
        try {
        PdfReader inputPDF = new PdfReader (inputStream);
        int totalPages = inputPDF.getNumberOfPages();
        ……
        PdfWriter writer = PdfWriter.getInstance (document, outputStream);
          document.open();
```

```
        PdfContentByte cb = writer.getDirectContent();
        PdfImportedPage page;
        while(sPage <= ePage)
       {
        document.newPage();
        page = writer.getImportedPage(inputPDF, sPage);
        cb.addTemplate(page, 0, 0);
        sPage++;
        }
        outputStream.flush();
        document.close();
        outputStream.close();
     }
     catch (Exception e) {……      }
      finally { ……      }
   }
```

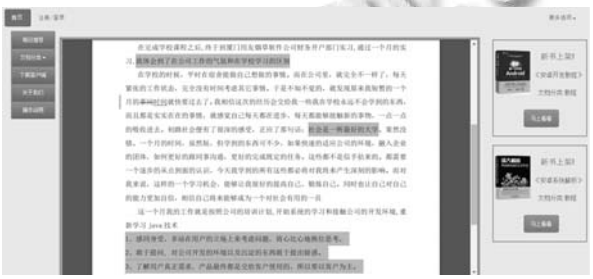

图 4 在线预览界面

3.3 下载加密

对于已经付费的用户, Web 平台开放下载权限, 其可以在电子文档信息界面中进行下载. 被下载文件将会被执行加密操作. 加密秘钥产生过程如 3.1 节所述, 图 5 为文件下载加密的时序图.

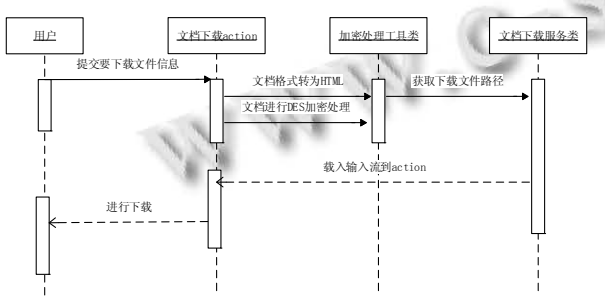

图 5 下载加密时序图

值得说明的是: 1)在文档被下载的同时, 用户当前计算机的 MAC 地址将会被记录; 2)被下载的加密文档只有使用 EDSC 平台的线下客户端并具备的一定的条件(将在 3.4 节进行细述)才可以被正常打开和显示.

3.4 线下客户端解密

为了防止付费用户把下载文档解密后发送或者复制到他人的计算机, 本文采用线下客户端校验的方法进行相应处理: 1)权限校验---用户打开客户端后, 需要输入用户名和密码并通过服务端的校验后, 客户端方可执行解密操作; 2)Mac 地址校验---用户在打开客户端后, 客户端会自动发送用户的 mac 地址到服务端, 与之前(第一次)存储的客户端 mac 地址进行匹配, 若不匹配, 则解密失败.

鉴于上述处理, 只有同时满足权限校验和 mac 地址校验的用户才可以正常打开从 web 端下载的加密文档, 从而保证下载的文档只能由一个用户的某一台设备查看. 图 6 为客户端解密的时序图, 图 7 和图 8 则分别为正常解密和解密失败时客户端的显示效果.

此外, 为了进一步保证下载解密并在客户端显示的文档内容不被外传, 本文还在客户端操作界面上禁用了所有的鼠标右键操作, 禁用了 Ctrl+C 、PrintScreen、Alt+ PrintScreen 等按键或组合键的操作, 并利用 Hook 技术对用户可能的截屏操作进行了监听, 例如 QQ 截图常用的 Ctrl+Alt+A 组合键等.

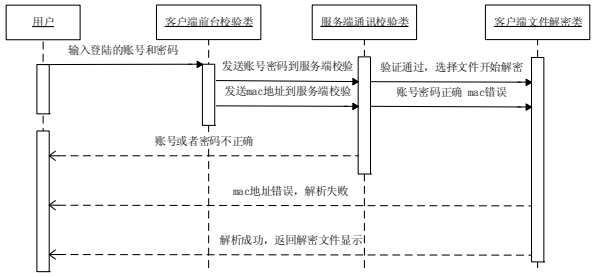

图 6 客户端解密时序图

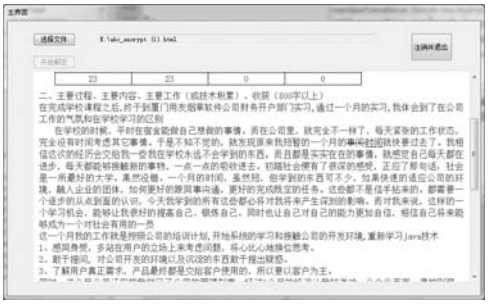

图 7　客户端成功解密

图 8　客户端解密失败

4　总结与展望

为了从线上和线下两个阶段对电子文档(共享文档)的访问权限进行控制, 本文利用 DES 加解密等技术设计实现了一个 web 和桌面应用项结合的版权保护方案 EDSC 平台. 线上访问控制采用与同类作品相类似的做法, 即通过要求用户进行注册并付费来限制文档的使用范围, 不同之处在于, 线上 web 平台会利用 DES 加密技术对被下载的文档的加密处理; 线下访问控制则基于 DES 解密技术、Hook 技术、Scoket 通信技术[11-15]等开发了特定的客户端, 用于完成用户身份校验以及对下载的加密文档进行解密和显示等操作. 为了更好地控制文档的使用范围, 本文还尝试了对所有可能的传播方式进行的禁用.

平台的主要局限在于: 1)只允许付费用户在第一次下载并成功解密并显示的计算机上线下阅读文档, 如果其想在个人不同的计算机上进行阅读, 需要向服务器申请解除 MAC 地址绑定, 为了防止频繁更换计算机, EDSC 平台也对解绑申请的次数进行了限定; 2)平台的线下客户端目前仅针对 PC 端进行开发, 尚不支持在移动平台对从 web 下载的加密文档进行阅读; 3)虽然本文已从最大可能上做到了访问控制, 阻止文档的非法外传, 但是仍不能做到百分百的阻止, 例如用户可以对解密显示的内容进行拍照后发给他人等.

鉴于上述分析, 本文后续工作将主要考虑新的身份校验方案, 针对移动平台开发响应客户端以及思考关于其他文档非法扩散途径的应对办法等.

参考文献

1 龚倍伦.论电子书之版权保护与限制—兼议亚马逊电子书删除事件.电子知识产权,2010,(1):74–77.

2 田燕.高校图书馆在数字化文库建设中的版权角色与版权管理—结合“百度文库”相关版权纠纷案件的分析.图书馆,2015,(10):84–87.

3 常峰,于良玉.DES 数据加密和解密技术.信息与电脑:理论版,2015,(11):106–107.

4 周文婷,马凤伟,孔庆.基于 DES 算法的文件加密系统的设计与实现.计算机安全,2012,(7):13–16.

5 冯黎明.文件自适应加密解密系统设计与实现.技术与市场,2016,(1):90.

6 胡岘,易晓东,戴华东.一种 HTML5 云文件系统.网络安全技术与应用,2012,(11):65–68.

7 易昌华.HTML5 发展趋势的研究和探索.价值工程,2012,31(36):314–315.

8 张俊杰.浅谈 HTML5 的技术革新.科技视界,2012,(18):185–186.

9 Frain B. Responsive web design with HTML5 and CSS3. Packt Publishing Limited, 2012.

10 扈小燕,刘培洵,刘力强.将 Word 文档自动转换成 PDF 格式的编程实现.计算机与现代化,2012,(2):187–189.

11 Pizano E, Rohatgi R. System and Method for Biometrically Secured, Transparent Encryption and Decryption. US, US8627106. 2014.

12 Mccarty RJ. Computer Program Products and Systems for Transparent Data Encryption and Decryption. US, US 7743403 B2. 2010.

13 Black W, Price K. Systems and Methods for Transparent Per-File Encryption and Decryption Via Metadata Identification. US20140258720. 2014.

14 Winslow RN. Assent to Conditions for Network Access. US, US 8826384 B2. 2014.

15 任小强,陈金鹰,李文彬,胡波.网络通信之 Java Socket 多线程通信.信息通信,2015,(6):206–207.