云南大学学报（自然科学版），2010，32（2）：140～146　　CN 53－1045/N　ISSN 0258－7971
Journal of Yunnan University

OpenSSL 在安全电子商务系统中的应用[*1]

张　璇，李成功，黄勤龙，彭　朋
（云南大学 软件学院，云南 昆明　650200）

摘要：OpenSSL 是一个功能强大的安全通信开放源码库，具备优秀的跨平台性能. SET 协议在电子商务安全电子支付服务中可以详细而准确地反映交易各方之间的各种关系，用来保证在 Internet 上实现银行卡支付交易的安全性. 基于 OpenSSL 在 Windows 环境下搭建了一个安全电子商务系统所必须的 CA 认证中心，并将其应用到 SET 协议中，实现了 SET 协议中 CA 认证中心的功能.

关键词：OpenSSL；SET；CA
中图分类号：TP 309　　**文献标识码**：A　　**文章编号**：0258－7971(2010)02－0140－07

电子商务作为网络技术发展的必然，已经对传统的经济贸易方式产生了冲击. 世界范围内的政府部门、公众服务机构、电信企业、银行等金融服务机构、众多 ISP 以及各类型企业和数以亿计的个人用户都开始广泛地参与电子商务活动. 现在，电子商务已经成为国际上各个国家制定经济政策的主要依据之一. 电子商务的蓬勃发展虽然为人们的生活提供了许多便利，但是由于 Internet 具有开放性、灵活性、共享性等特点，也为参与电子商务活动的实体带来了巨大的安全威胁. 设计并实现一个通用性强、安全性高的网络协议是实现电子商务安全的关键技术之一，它会对电子商务整体性产生很大的影响[1]. 目前国内外广泛采用的安全电子商务协议有 2 种，即 SSL（Secure Socket Layer，安全套接层协议）和 SET（Secure Electronic Transaction，安全电子交易协议），SSL 协议虽然简单，但不能实现多方的认证和访问控制，而 SET 协议则可以对所有参与电子交易的成员进行认证并对访问的内容进行控制，因而可以很好地满足电子商务安全服务标准的要求.

1　OpenSSL 及 SET 协议中的 CA 认证中心简介

1.1　OpenSSL　OpenSSL 是一个功能强大的安全通信开放源码库，它采用 C 语言作为开发语言，具有优秀的跨平台性能，支持 Linux、UNIX、Windows、Mac 等平台，它提供的主要功能有：SSL 协议实现（包括 SSLv2、SSLv3 和 TLSv1）、大量软算法（对称/非对称/哈希）、大数运算、非对称算法密钥生成、ASN. 1 编解码库、证书请求（PKCS10）编解码、数字证书编解码、CRL 编解码、OCSP 协议、数字证书验证、PKCS#7 标准实现和 PKCS#12 个人数字证书格式实现等功能[2－3].

OpenSSL 目前最新的版本是 0.9.8h，我们使用的是 0.9.8g，其中，我们主要使用了它的密钥和证书管理功能. 密钥和证书管理是 SET 协议的一个重要组成部分，OpenSSL 为之提供了丰富的功能，支持多种标准. 首先，OpenSSL 实现了 ASN. 1 的证书和密钥相关标准，提供了对证书、公钥、私钥、证书请求以及 CRL 等数据对象的 DER、PEM 和 BASE64 的编解码功能. 在此基础上，OpenSSL 实现了对证书的 X. 509 标准编解码、PKCS#12 格式的编解码以及 PKCS#7 的编解码功能.

---

* **收稿日期**：2008－12－07
基金项目：云南省教育厅青年教师科学研究基金资助项目（K1050513）；云南大学理（工）科校级科研资助项目（KL070032）.
作者简介：张　璇（1978－），女，江苏人，讲师，主要从事安全电子支付、数据隐私保护方面的研究.

1.2　SET 协议中的 CA 认证中心　SET 协议是由 Visa 和 MasterCard 两大信用卡组织联合开发的电子商务安全协议. 它是一种基于消息流的协议,它可以详细而准确地反映交易各方之间的各种关系,用来保证在公共网络上实现银行卡支付交易的安全性,因而成为 Internet 上进行在线交易的电子支付系统规范. 目前 SET 协议已在国际上被大量实验性地使用并经受了考验,成为了事实上的工业标准[4-7].

在 SET 协议中,参与交易的各方都需要在交易过程中证明自己的身份,为了满足这个要求,SET 协议要求所有交易参与方在 CA 认证中心进行身份认证并使用 CA 为其制作的数字证书在交易中证明自己的身份,如图 1 所示.

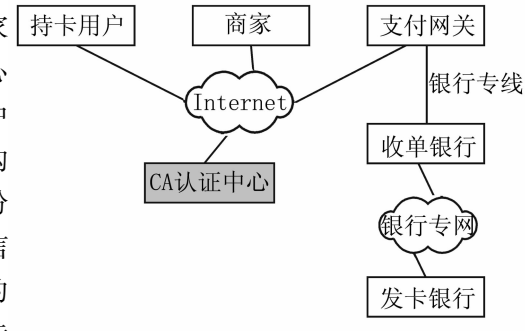

图 1　SET 安全支付参与方及应用系统框架

Fig. 1　Entities in SET and application system framework

SET 协议规定的交易过程包括:持卡用户注册,商家注册,持卡用户发送购买请求,商家请银行进行支付验证,商家获得支付. 其中,持卡用户和商家的注册就是在 CA 认证中心申请数字证书,支付网关在参加交易前也必须到 CA 认证中心申请数字证书,在交易过程中,持卡用户将数字证书作为购买请求信息的一部分首先发送给商家,用于证明自己的身份(见图 2),商家收到此购买请求,确认持卡用户身份和订单信息无误后,以类似图 2 所示的方式生成包含商家数字证书的支付验证请求发送给银行的支付网关,银行支付网关验证持卡用户身份、商家身份以及支付信息无误后发送一个支付代码给商家,待商家发货并且得到持卡用户的确认后商家就可以凭支付代码到支付网关获得支付. 在整个交易过程中,CA 作为电子交易安全的关键环节,主要负责产生、分配并管理所有参与网上交易的实体所需的身份认证数字证书. 电子交易的各方都必须拥有合法的身份,即由 CA 签发的数字证书,在交易的各个环节,交易的各方都需检验对方数字证书的有效性,从而解决了用户信任的问题[8]. 在接下来的内容中我们将介绍如何利用 OpenSSL 搭建 SET 协议的 CA 认证中心.

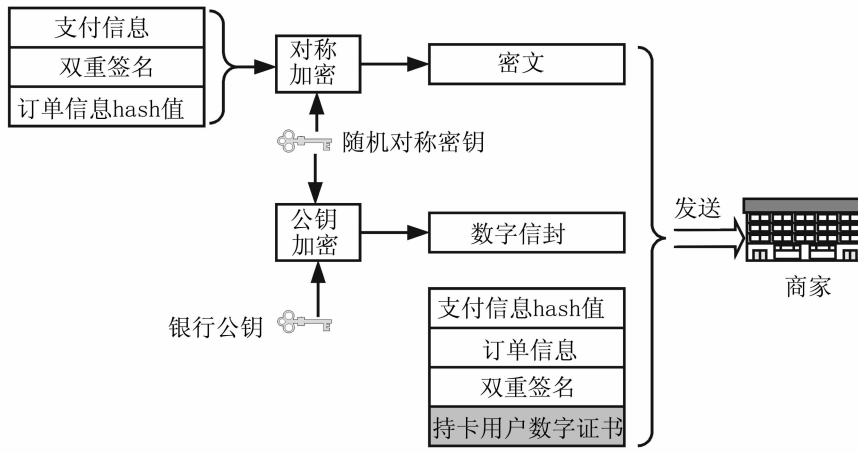

图 2　持卡用户发送的 SET 购买请求

Fig. 2　Cardholder sends purchase request in SET

2　基于 OpenSSL 开发 SET 协议的 CA 认证中心

CA 作为权威的、公正的、可信赖的第三方,其作用是至关重要的,它主要负责发放和管理数字证书,相

应的需求有[9-10]:

(1) 接受并验证用户数字证书的申请;

(2) 向申请者颁发数字证书,即证书的发放;

(3) 接受用户数字证书的查询;

(4) 数字证书的更新、作废和归档;

(5) 历史数据的归档.

在SET协议中我们实现的CA主要由以下3部分组成:

(1) *注册服务器* 通过Web服务器为用户提供数字证书申请服务;

(2) *证书申请受理机构* 负责处理用户提出的数字证书申请;

(3) *认证中心服务器* 是数字证书生成、发放的运作实体,同时提供数字证书的管理服务.

下面以完成一个用户证书申请,生成证书请求到制作证书的过程为例,说明如何在Windows环境下使用OpenSSL实现CA认证中心.

2.1 准备工作 在Windows下编译OpenSSL,我们使用的是Visual Studio2005环境下Vc8的cl编译器[11].

(1) 安装ActivePerl(初始化的时候需要用到Perl解释器);

(2) 使用VS2005下的Visual Studio 2005 Command Prompt进入控制台模式(这个模式会自动设置各种环境变量);

(3) 解压缩OpenSSL的包,进入OpenSSL的目录;

(4) 在OpenSSL的目录下运行Perl Configure VC-WIN32(必须在这个目录下执行该命令,否则找不到Configure文件,或者指定完整的Configure文件路径);

(5) 在解压目录下执行ms\do_ms命令;

(6) 执行nmake -f ms\ntdll.mak编译后,如果编译成功,最后的输出都在out32dll目录下,包括可执行文件、2个dll和2个lib文件:libeay32.dll,libeay32.lib,ssleay32.dll,ssleay32.lib;将libeay32.dll和SSLeay32.dll复制到system32目录下,将libeay32.lib和SSLeay32.lib复制到当前目录并且导入工程中,将OpenSSL里inc32下的OpenSSL复制到VS2005的include文件夹下(或者用option->directories操作).

2.2 产生密钥对 在SET协议中,将用户和用户的公钥绑定在一起存放在数字证书中,并且与相关联的私钥联合使用,可以完成身份认证、数据加解密以及签名验证等功能,为了保证这些功能的安全性,形成一对安全的密钥对是至关重要的.RSA算法是现在比较流行的公开密钥算法,在OpenSSL里,产生RSA密钥对的文件为genrsa.c[11].

在genrsa.c文件中,主要用函数RSA_generate_key(int bits,unsigned long e_value,void (*callback)(int,int,void *),void *cb_arg)负责产生密钥对,其中,参数bits是密钥的长度(比如:1 024位),参数e_value是RSA的公钥,callback和cb_arg两个参数为产生私钥过程中的回调函数,在一般应用中将它们置为NULL即可.运行该函数将返回一个EVP_PKEY的RSA结构.

为安全起见,我们使用自己编写的1 024位RSA算法来生成一对安全的密钥对[12],并将密钥对通过[char BN_E[20],公钥e]、[char BN_N[20],模数n]放入OpenSSL中的EVP_PKEY的RSA结构中,用来生成我们自己需要的证书.具体实现代码如下:

```
rsa = RSA_generate_key(bits,RSA_F4,NULL,NULL);
bnn = BN_new();//重新设置密钥对
bne = BN_new();
BN_hex2bn(&bnn,BN_N);// 存放我们自己生成的公钥e
BN_hex2bn(&bne,BN_E );// 存放我们自己生成的模数n
```

```
rsa->n = bnn;
rsa->e = bne;
if (! EVP_PKEY_assign_RSA(pk,rsa))
        goto err;
rsa = NULL;
X509_REQ_set_pubkey(x,pk);//设置证书公钥
```

在目前开源的证书加密库中,普遍都不允许我们将自己生成的公钥导入证书,而 OpenSSL 可以使用 OpenSSL API 导入我们自己生成的公钥对来制作证书,这样可以让我们根据系统的需要设计不同强度的密码算法来实现不同安全等级的系统.

2.3　生成证书请求文件　CA 认证中心在生成数字证书前需要获取用户提交的证书请求文件,此文件包括客户端产生的公钥和用户输入的证书信息,CA 通过此证书请求文件产生证书文件. 在 OpenSSL 中,证书请求文件的生成由 req. c 实现[11].

证书请求包括 3 个部分:证书请求信息,签名算法标识符和证书请求信息的数字签名. 证书的请求信息包括用户实体的别名、公钥和一系列有关的其他信息. 构造一个证书请求包含如下几个步骤:

(1) 构造用户实体信息,包含实体别名,实体公钥和一组可选属性. 在生成 req 证书请求之前需要申请一个 X509_REQ 对象,为加入实体信息还必须申请一个 X509_NAME 对象,然后将信息加到如下的结构中,最后把这个结构赋给 req.

```
X509_REQ *x; X509_NAME *name = NULL;
X509_NAME_add_entry_by_NID(name, NID_countryName, MBSTRING_UTF8, (unsigned char *)
CoutryName, -1, -1,0);//添加国家信息
X509_NAME_add_entry_by_NID(name, NID_stateOrProvinceName, MBSTRING_UTF8, (unsigned char
*)ProvinceName, -1, -1,0) ;//添加省份信息
X509_NAME_add_entry_by_NID(name, NID_localityName, MBSTRING_UTF8, (unsigned char *)Loca-
tionName, -1, -1,0) ;//添加市县信息
X509_NAME_add_entry_by_NID(name, NID_commonName, MBSTRING_UTF8, (unsigned char *)Com-
monName, -1, -1,0) ;//添加申请者名字信息
X509_NAME_add_entry_by_NID(name, NID_organizationName, MBSTRING_UTF8, (unsigned char *)
OrganizationName, -1, -1,0) ;//添加公司信息
X509_NAME_add_entry_by_NID(name, NID_organizationalUnitName, MBSTRING_UTF8, (unsigned char
*)OrganizationNameUnit, -1, -1,0) ;//添加部门信息
```

还需要加入一个实体公钥(上小节生成的 pk)

```
X509_REQ_set_pubkey(x,pk);//设置证书公钥
```

加入一组可选的扩展属性:

```
STACK_OF(X509_EXTENSION) *exts = sk_X509_EXTENSION_new_null();
add_ext_req(exts, NID_subject_alt_name, EmailAddress);
add_ext_req(exts, NID_netscape_cert_type, "client,email");
```

生成扩展对象:

```
X509_REQ_add_extensions(x,exts);
```

加入扩展项目:

```
sk_X509_EXTENSION_pop_free(exts, X509_EXTENSION_free);
```

(2) 用主体的私钥对上面的 req 进行签名,在签名之前需要选择哈希算法.

```
if (! X509_REQ_sign(x,pk,EVP_md5()))////EVP_sha1() 选择 MD5 算法或 SHA - 1 算法
    goto err;
```

生成文件:

```
bp = BIO_new(BIO_s_file());
if (BIO_write_filename(bp,szCertFile ) < = 0)
{
        return NULL;
}
i2d_X509_REQ_bio(bp,x);//生成 der 格式的文件
PEM_write_bio_X509_REQ(bp,x);//生成 PEM 格式的文件
```

这样一份证书请求文件就形成了.

2.4 生成证书文件 CA 认证中心将根据证书请求文件生成用户实体的数字证书. 根据前面有关数字证书的介绍我们已经知道了证书的构成项,下面我们就只需要一项一项添加即可.

(1) 生成证书对象

```
X509 * x;//要产生的证书
X509 * m_pCACert;//导入 CA 根证书
X509_REQ * req = NULL;//导入证书请求文件
if ((x = X509_new()) = = NULL)
    goto err;
if ((m_pCACert = X509_new()) = = NULL)
    return -1;
```

(2) 读入 CA 文件和证书请求文件

```
bp = BIO_new(BIO_s_file());
BIO_read_filename(bp,zsCACertFile);//CA 证书格式为 pem
m_pCACert = PEM_read_bio_X509(bp,NULL,NULL,NULL);//读入 CA 证书文件
BIO_read_filename(bp,zsPrivateKeyFile);//CA 的私钥文件一般保存为 PEM 格式
m_pCAKey = PEM_read_bio_PrivateKey(bp,NULL,NULL,NULL);//读入 CA 的私钥文件
if (BIO_read_filename(bp,zsCertReqFile ) < = 0)
{
  return NULL;
}
req = d2i_X509_REQ_bio(bp,NULL);//读入证书请求文件 req
req = PEM_read_bio_X509_REQ(bp,NULL,NULL,NULL);
```

(3) 设置版本号

```
X509_set_version(x,2);//设置版本号
```

(4) 设置证书序列号

```
ASN1_INTEGER_set(X509_get_serialNumber(x),serial);
```

(5) 设置证书开始时间

```
X509_gmtime_adj(X509_get_notBefore(x),0);
```

(6) 设置证书结束时间

```
X509_gmtime_adj(X509_get_notAfter(x),(long)60 * 60 * 24 * days);
```

(7) 设置证书的主体名称,req 是刚刚生成的证书请求

```
X509_set_subject_name(x,X509_REQ_get_subject_name(req));
```

(8) 设置证书的公钥信息

```
pk = X509_PUBKEY_get((*req).req_info->pubkey);//从 req 中得到公钥信息
X509_set_pubkey(x,pk);//设定公钥信息
```

(9) 设置证书的签发者信息

```
X509_set_issuer_name(x,X509_get_subject_name(m_pCACert));// m_pCACert 是 CA 证书
```

(10) 设置扩展项目

```
add_ext_X509(x,NID_basic_constraints,"critical,CA:FALSE");
```

(注意:这里的 FALSE 改为 TRUE 的话,证书安装时,会认为证书为 CA 根证书文件,若制作的为个人证书,用 FALSE.)

```
add_ext_X509(x,NID_key_usage,"critical,keyCertSign,cRLSign");
add_ext_X509(x,NID_subject_key_identifier,"hash");
/* Netscape 相关扩展 */
add_ext_X509(x,NID_netscape_cert_type,"MyOpenSSLCA");
add_ext_X509(x,NID_netscape_comment,"The comment extension for SET");
```

(11) 设置签名值

```
if (! X509_sign(x,m_pCAKey,EVP_md5()))//EVP_sha1()
    goto err;
```

这样一份符合 X.509 标准的证书就生成了,最后需要对它进行编码保存.

```
if (BIO_write_filename(bp,zsCertFile ) <= 0) //写成证书文件
{
        return NULL;
}
//i2d_X509_bio(bp,x);//DER 格式
PEM_write_bio_X509(bp,x);//PEM 格式
```

通过上述方法,我们生成了符合 X.509 标准的数字证书,并且这个数字证书可以放在 Windows 的 IE 浏览器中进行统一的管理.

3　OpenSSL 在 SET 协议中的应用

我们将基于 OpenSSL 实现的 CA 认证中心系统应用到 SET 协议中,并对 SET 协议进行改进,实现了一个完整的安全电子商务系统,在这个系统中我们的公私钥是在客户端产生的,这样可以保证私钥只是由用户保存,而将公钥和其他证书必要的信息发送给 CA 服务器,由 CA 服务器生成数字证书,证书生成后用户可以下载到自己的电脑上安装证书文件. 整个系统的所有加解密函数和证书生成函数都是用 C 写的动态链接库,然后用 C#来调用,这样可以有效地利用 C 语言的高效和 C#语言的便利.

4　结束语

SSL 协议已经在电子商务安全中被广泛地使用,但 SSL 协议不能确保交易的非否认性,而 SET 协议弥补了 SSL 协议的缺陷,能够更好地保证交易的安全性,我们将基于 OpenSSL 实现的 CA 认证中心应用到 SET 协议中不仅可以简化 SET 协议的实现,而且生成的证书可以放在 IE 浏览器中进行统一管理,完全可以满足 SET 协议对 CA 认证中心的要求.

参考文献:

[1] 陈国辉,施伟. OpenSSL在电子商务安全中的应用[J]. 微计算机信息(测控自动化),2004,20(5):118-120.

[2] The OpenSSL project[EB/OL]. [2008-08-10]. http://www.openssl.org/.

[3] 霍英. OpenSSL在网络安全传输中的应用研究[J]. 企业技术开发,2006,25(8):9-11.

[4] MasterCard and Visa Corporations. Secure electronic transaction (SET) specification - book1: business description Version 1.0, May 1997[EB/OL]. [2008-7-28]. Http://www.setco.org.

[5] MasterCard and Visa Corporations. Secure electronic transaction (SET) specification - book2: programmer's guide Version 1.0, May 1997[EB/OL]. [2008-7-28]. Http://www.setco.org.

[6] MasterCard and Visa Corporations. Secure electronic transaction (SET) specification - book3: formal protocol definition Version 1.0, May 1997[EB/OL]. [2008-8-10]. Http://www.setco.org.

[7] William Stallings. Cryptography and network security[M]. 4th ed. Prentice Hall, 2005.

[8] Donal O'Mahony, Michael Peirce, Hitesh Teswari. Electronic payment systems for e-commerce[M]. 2nd ed. Boston and London: Artech House, INC. 2001.

[9] ITU-T recommendation X.509 (1997E): information technology - open systems interconnection - the directory: authentication framework, June 1997[EB/OL]. [2008-3-22]. Http://www.itut.org.

[10] Carlisle Adams. Understanding PKI: concepts, standards, and deployment considerations[M]. 2nd ed. Addison-Wesley Professional, 2002.

[11] 赵春平. OpenSSL编程[EB/OL]. [2008-7-28]. http://download.csdn.net.

[12] Bruce Schneier. Applied cryptography: protocols, algorithms, and source code in C[M]. 2nd ed. Wiley, 1996.

The use of OpenSSL in the secure electronic commerce system

ZHANG Xuan, LI Cheng-gong, HUANG Qing-long, PENG Peng

(School of Software, Yunnan University, Kunming 650200, China)

Abstract: OpenSSL is a powerful secure open library for secure communication, and it has excellent cross-platform performance. SET protocol is used to guarantee the security of the credit card-based payment system over Internet; it can show the detailed relationship among the parties in secure electronic payment exactly. In this paper, we mainly described how to use OpenSSL in windows to develop CA, where CA is a main part of a secure electronic commerce system, and then we apply it into SET protocol to realize the function of CA.

Key words: OpenSSL; SET; CA