Computer Science and Application 计算机科学与应用, 2018, 8(5), 709-720
Published Online May 2018 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2018.85079

Hans汉斯

The Optimization of SSL Protocol Based on TCM

Xinglan Zhang, Guanhua Ren

Beijing University of Technology, Beijing
Email: 2461015133@qq.com

Received: May 1st, 2018; accepted: May 16th, 2018; published: May 23rd, 2018

Abstract

Trusted cryptography module TCM is a module which combines the hardware and software of a trusted computing platform. It provides cryptographic operations for trusted computing platforms and has a protected storage space. TCM establishes the source of trust through the system security chip trusted cryptographic module, and then establishes the trust chain in accordance with the relationships of the system startup process through the hardware, firmware, operating system and application program to ensure the credibility of the computing platform and the program. The traditional SSL protocol is basically impossible to defend against man-in-the-middle attacks in the handshake phase, because the identity authentication phase of the handshake protocol does not have information on the underlying hardware platform. The improved scheme of the present invention is based on the new implementation idea of the trusted cryptography module (TCM). Based on the analysis of the existing SSL protocol, an improved SSL protocol based on the trusted cryptography module TCM is proposed and implemented. The protocol involves the communication between the client and the server, and is mainly designed from the following aspects: 1) Using the platform identity certificate to provide hardware and system-level authentication. 2) The protection of the pre-master key by the storage master key of the chip which ensures the transmission of shared keys of both parties and is better guaranteed before data exchange. This method applies the TCM module to the SSL protocol, and establishes hardware-based and system-level authentication for both parties of the communication. It also improves the encryption algorithm so that during the handshake phase, the data transmission between the two parties is better guaranteed, and provides information transmission for the recording protocol part. Stronger security guarantees enhance the ability to resist attacks and improve the security of the agreement.

Keywords

SSL Protocol, Information Security, Trusted Computing

基于TCM的SSL协议的优化

张兴兰，任冠华

北京工业大学，北京

文章引用: 张兴兰, 任冠华. 基于 TCM 的 SSL 协议的优化[J]. 计算机科学与应用, 2018, 8(5): 709-720.
DOI: 10.12677/csa.2018.85079

Email: 2461015133@qq.com

收稿日期：2018年5月1日；录用日期：2018年5月16日；发布日期：2018年5月23日

摘　要

可信密码模块TCM是一个可信计算平台的软硬件相结合的模块，为可信计算平台提供密码运算功能，具有受保护的存储空间。通过系统安全芯片可信密码模块建立信任的源头，然后通过硬件、固件、操作系统和应用程序，按照系统启动过程的前后控制关系建立信任链的方法确保计算平台和程序的可信。传统的SSL协议在握手阶段对于中间人攻击基本上是无法防卫的，这是由于握手协议中身份认证阶段没有底层硬件平台的信息。本发明的改进方案基于可信密码模块(TCM)新的实现思想，在分析现有SSL协议的基础上，提出并实现了一种基于可信密码模块TCM的SSL改进协议。该协议涉及客户端和服务器两方的通信，主要从以下方面进行设计，1)利用平台身份证书提供基于硬件和系统层面的认证。2)通过芯片的存储主密钥对预主密钥的保护，使得数据交换前，双方共享密钥的传输得到了更好的保证。本方法将TCM模块应用到SSL协议中，建立通信双方基于硬件和系统层面的认证，并且提升了加密算法，使得握手阶段，双方数据传输得到了更好的保证，为记录协议部分的信息传输提供更强的安全保障，增强了抵御攻击的能力，提高了协议的安全性。

关键词

SSL协议，信息安全，可信计算

1. 引言

SSL 协议处于 TCP/IP 传输层和应用层之间，为上层应用数据传输提供安全通道，可以提供身份认证、信息加解密等安全功能。主要目标是为两个通信实体提供信息安全性、完整性、安全性的会话连接。

传统的 SSL 协议对于身份攻击和会话密钥攻击的防卫能力比较弱；另外对最后提及的中间人攻击除了加强管理外，基本无法从技术角度防止攻击。

这些安全问题很大程度是由于 SSL 协议中没有对系统平台环境的认证，对硬件设备包括 CPU 中计数器、寄存器等重要的值没有包括在身份认证阶段；在会话密钥传输阶段，预主密钥的生成、保存、传输，以及会话密钥即主密钥的安全存储，这些方面的抗攻击能力不是很强。

可信密码模块 TCM 是一个可信计算平台的软硬件相结合的模块，为可信计算平台提供密码运算功能，具有受保护的存储空间。通过系统安全芯片可信密码模块建立信任的源头，然后通过硬件、固件、操作系统和应用程序，按照系统启动过程的前后控制关系建立信任链的方法确保计算平台和程序的可信[1]。

本文对 SSL 的优化基于可信密码模块(TCM)新的实现思想，在分析现有 SSL 协议的基础上，提出并实现了一种基于可信密码模块 TCM 的 SSL 改进协议。SSL 协议中握手协议作为其中最核心和最复杂的协议，完成了 SSL 的大部分通信交互，优化方案主要对握手协议部分从以下方面进行改进：利用平台身份证书提供基于硬件和系统层面的认证；通过芯片的存储主密钥对预主密钥的保护，使得数据交换前，

双方共享密钥的传输得到了更好的保证。本方法将 TCM 模块应用到 SSL 协议中，建立通信双方基于硬件和系统层面的认证，并且提升了加密算法，使得握手阶段，双方数据传输得到了更好的保证，为记录协议部分的信息传输提供更强的安全保障，增强了抵御攻击的能力，提高了协议的安全性[2]。

2. 符号及函数说明

一种基于可信密码模块(Trusted Cryptography Module, TCM)的 SSL 协议，用到的符号及函数说明如表 1 所示：

3. 认证优化

改进的 SSL 协议中，信双方的身份认证过程，主要是验证 TCM 的平台身份证书和 SSL 协议证书的哈希值的过程[3]。首先身份验证前提是：通信双方都有 TCM 芯片；通信双方已经申请且拥有了经过认证的平台身份证书[4]。客户端对服务器端认证的过程如图 1 所示：

具体流程描述如下：

1) 服务端的工作流程如下：

a) 服务器端封装 SSL 证书信息和 TCM 平台证书信息，封装后的信息分别记做 m_1、m_2：

Table 1. Symbols and functions directions
表 1. 符号及函数说明

符号/函数	描述
T_C，O_C，H_C	客户端 TCM、客户端 TCM 所有者、客户端 TCM 所在主机
T_S，O_S，H_S	服务器端 TCM、服务器端 TCM 所有者、服务器端 TCM 所在主机
EK_N	某个 TCM 的 EK 公钥
$PubK_S$，$PubKCert_S$	服务器端的 SSL 证书公钥，SSL 证书
PEK_N，PEK_N^-，$PEKCert_N$	PEK 公钥、PEK 私钥、PEK 证书
$KeyPub$，$KeyPri$	密钥对的公开区域、秘密区域
hd_k，shd_Y	密钥 K 的句柄、临时公钥为 Y 的密钥协商会话句柄
$m_1 \Vert m_2$	对 m_1 和 m_2 进行串接
$hash(m)$	m 的哈希值
$hmac(m,k)$	用密钥 k 计算 m 的基于散列的消息认证码
$DS(x)$	对 x 的双签名
K_S，K_P	签名密钥私钥和对应的公钥
$PreMasterSecret$，$MasterSecret$	预主密钥，主密钥
$senc(m,k)$，$sdec(m,k)$	用对称密钥 k 加密 m、解密 m
$aenc(m,k)$，$adec(m,k)$	用公钥 k 加密 m，用私钥 k 解密 m；或者用私钥 k 加密 m，用公钥 k 解密 m
$kdf(f,s)$	以 f 为标志 s 为种子的密钥导出函数
$SM2KE(K_1,K_2,X,Y)$	用发起方的公钥 k1 和临时公钥 X 以及响应方的公钥 K2 和临时密钥公钥 Y，计算 SM2 密钥协商协议的共享会话密钥

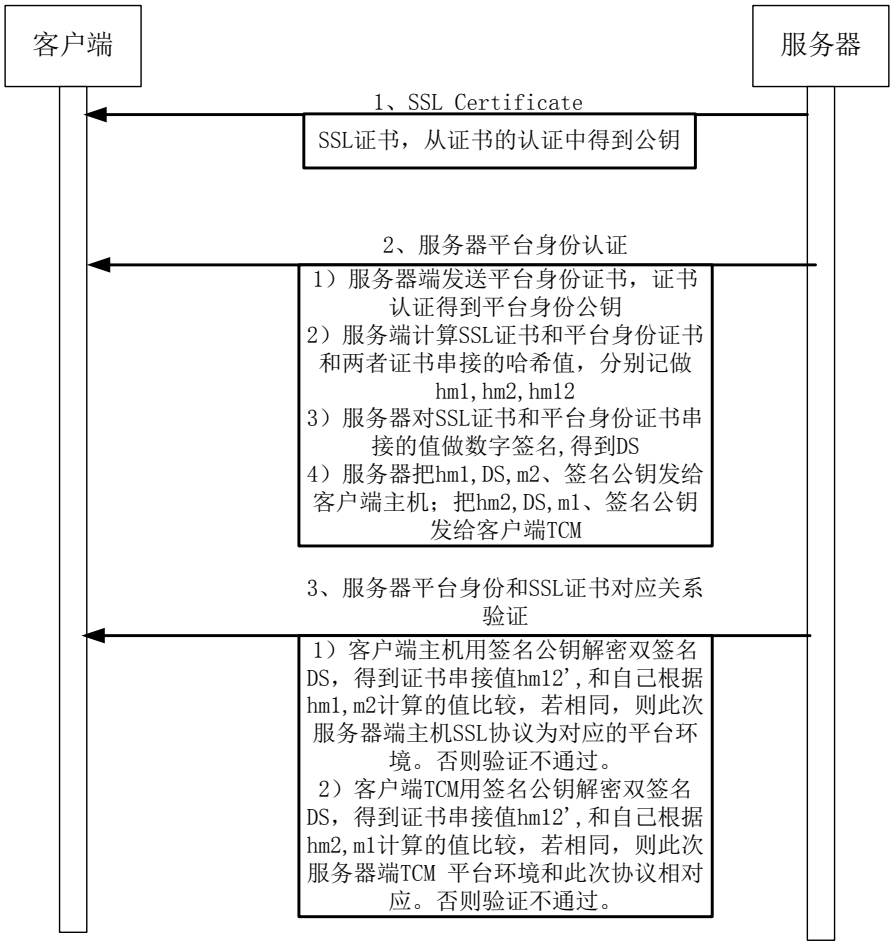

Figure 1. SSL protocol authentication optimization process
图 1. SSL 协议身份认证优化过程

$$m_1 = \left(PubK_S, PubKcert_S\right);\ \ m_2 = \left(PEK_S, PEKCert_S\right).$$

b) 服务器端分别计算 SSL 证书、TCM 平台身份证书、两类封装证书串接的哈希值，计算后的值分别记做计算 hm_1、hm_2、hm_{12}：

$$hm_1 = hmac\left(m_1\right);\ \ hm_2 = hmac\left(m_2\right);\ \ hm_{12} = hmac\left(m_1 \| m_2\right).$$

c) 服务器端生成签名密钥对 K_S、K_P，服务器端进行数字双签名得到 DS_S：$DS_S = aenc\left(m_{12}, K_{CS}\right)$，此方程使用私钥 K_{CS} 加密 m_{12}，利用数字双签名将 TCM 平台身份证书和 SSL 证书关联在一起，表名此次 SSL 协议的特定平台环境为 TCM 中的信息。

d) 服务器端把 K_S，m_1，DS_S 及 m_1 的哈希值 hm_1 发送给客户端 TCM 即 TCM_C，这样 TCM 只能得到 m_2 的杂凑值，但得不到 m_2；把 K_P，m_2，DS_S 及 m_2 的哈希值 hm_2 发送给客户端主机 H_C，这样 H_C 只能得到 m_1 的杂凑值而得不到 m_1。

2) 客户端的验证流程如下：

a) 客户端对服务器端的 SSL 证书和平台身份证书分别进行认证。

b) 客户端对 SSL 证书和平台身份证书的对应关系进行认证，验证此次 SSL 协议的特定平台环境是否是 TCM 中的系统环境，具体验证如下：

计算 $hm'_{12}=adec\left(DS_S,K_P\right)$；计算 $hm_2=hmac\left(m_2\right)$；验证方程 $hm_{12}=hmac\left(m_1\|m_2\right)$是否成立，此方程用于计算 m_1 和 m_2 求并后的散列值，若成立则签名有效，继续以下步骤，否则终止此次协议[5] [6]。

4. 会话密钥交换过程优化

主密钥作为会话密钥，是由预主密钥、客户端和服务器端在 Hello 阶段交互的随机数经过加密产生，优化后的密钥交换过程加入 TCM 模块的参与，加密密钥的生成根据有随机性和机密性，同时对会话密钥的存储保护也得到加强[7] [8]。密钥交换过程如图 2 所示：

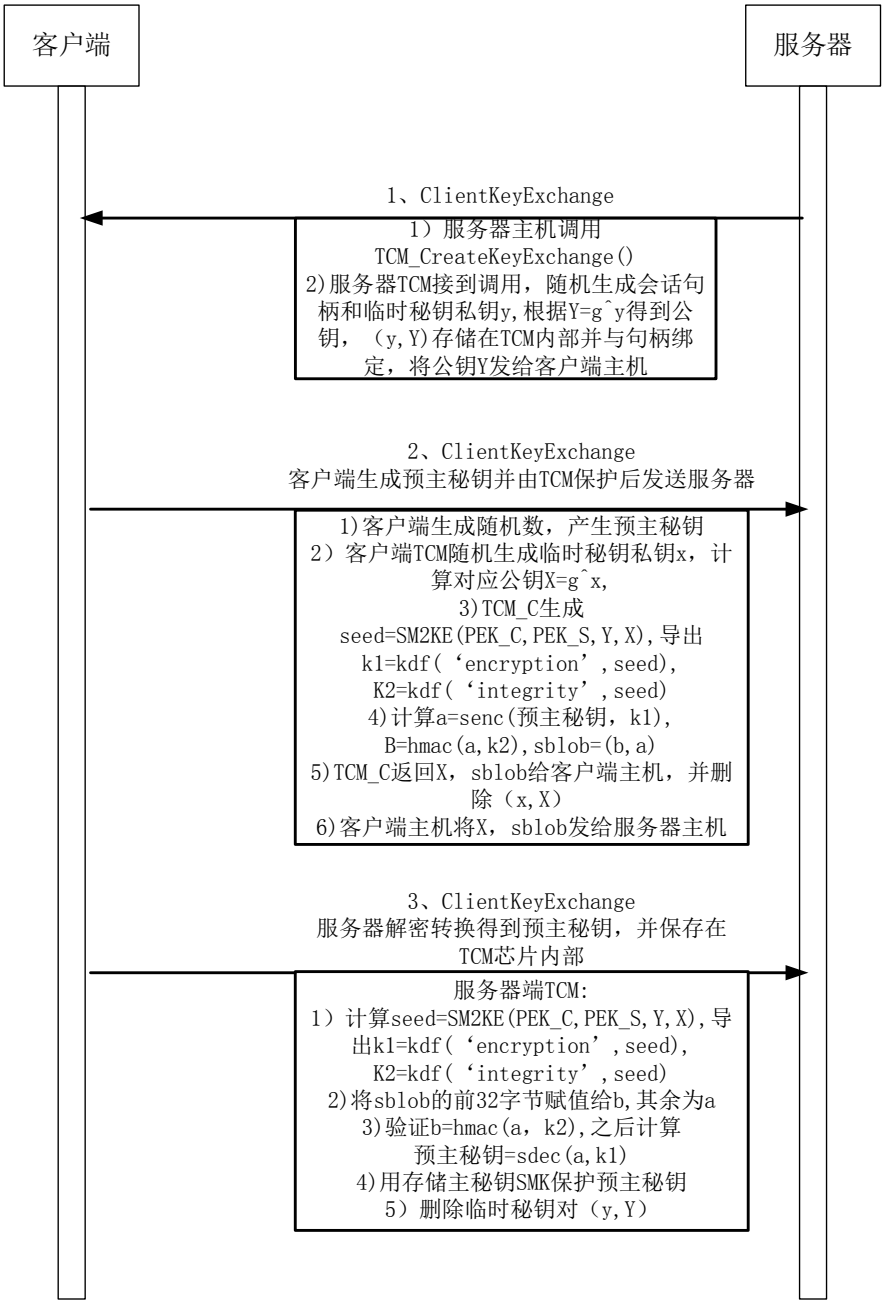

Figure 2. Flowchart for optimizing key exchange in the SSL protocol
图 2.SSL 协议密钥交换部分优化流程图

具体流程描述如下：

1) 双方 TCM 模块生成公私密钥对，私钥句柄发送给自己所在主机，公钥发送给通信的另一方。

2) 客户端计算用于加密预主密钥的加密密钥：

首先客户端主机随机生成预主密钥；TCM 模块随机生成临时密钥私钥 x，计算临时密钥公钥 $X = g^x$；

接着 TCM 客户端调用 TCM 模块 SM2 加密算法，用服务器端和客户端的 SSL 证书公钥和双方 TCM 模块生成的密钥对中的公钥，计算 SM2 密钥协商协议的共享会话密钥种子；再根据种子导出预主密钥的加密密钥。此处的 $SM2KE$ 函数为调用 TCM 模块的 $SM2$ 公钥加密算法，用客户端和服务器端的平台身份公钥和临时公钥，计算用于会话协商中的种子。具体计算流程如下：

1) $seed = SM2KE\left(PEK_C, PEK_S, Y, X\right)$，$k_1 = kdf\left('encryption', seed\right)$

2) 计算 $a = senc\left(PreMasterSecret, K_1\right)$，$b = hmac\left(a, K_2\right)$，$sblob = \left(a, b\right)$

最后，客户端 TCM 模块返回 X 和 $Sblob$ 给主机，主机将 X 和 $Sblob$ 传给服务器主机，客户端主机同时删除临时密钥对 $\left(x, X\right)$。

服务器端 TCM 模块首先使用与客户端相同的方法生成预主密钥的加密密钥种子；接着 TCM 模块根据客户端传送的相关信息计算预主密钥，具体计算流程如下：

1) 将 $Sblob$ 前 32 字节赋值给 b，其余字节赋值给 a；

2) 验证 $b = hmac\left(a, K_2\right)$；

3) 计算 $PreMasterSecret = sdec\left(a, K_1\right)$；

最后用 TCM 模块使用存储主密钥保护预主密钥，同时删除随机生成的临时密钥对。

5. 优化后的 SSL 握手过程

优化后的握手协议部分通信过程如图 3 所示[9] [10] [11]：

1) SSL 客户端通过 Client Hello 消息将它支持的 SSL 版本、加密算法、密钥交换算法、 $hmac$ 算法等信息发送给 SSL 服务器。

2) SSL 服务器确定本次通信采用的 SSL 版本和加密套件，并通过 Server Hello 消息通知给 SSL 客户端。如果 SSL 服务器允许 SSL 客户端在以后的通信中重用本次会话，则 SSL 服务器会为本次会话分配会话 ID，并通过 Server Hello 消息发送给 SSL 客户端。

3) SSL 服务器将携带自己公钥信息的数字证书 $PubKCert_S$ 和服务器公钥 $PubK_S$ 发送给 SSL 客户端。

4) SSL 服务器将自身的平台身份证书 $PEKCert_S$ 和对应的平台身份公钥 PEK_S 发送给 SSL 客户端。

5) 服务器端封装封装 SSL 证书信息为 m_1：$m_1 = \left(PubK_S, PubKCert_S\right)$；TCM 证书信息为 m_2：$m_2 = \left(PEK_S, PEKCert_S\right)$。

6) 服务器端计算 m_1 的哈希值 hm_1：$hm_1 = hmac\left(m_1\right)$；计算 m_2 的哈希值 hm_2：$hm_2 = hmac\left(m_2\right)$；计算 m_1、m_2 的哈希值 hm_{12}：$hm_{12} = hmac\left(m_1 \| m_2\right)$。

7) 服务器端生成签名密钥对 K_S、K_P，服务器端进行数字双签名得到 DS_S：$DS_S = aenc\left(m_{12}, K_{CS}\right)$，利用数字双签名将TCM平台身份证书和SSL证书关联在一起，表名此次SSL协议的特定平台环境为TCM中的信息。

8) 服务器端把 K_S，m_1、DS_S 及 m_1 的哈希值 hm_1 发送给客户端 TCM 即 TCM_C，这样 TCM 只能得到 m_2 的杂凑值，但得不到 m_2；把 K_P，m_2，DS_S 及 m_2 的哈希值 hm_2 发送给客户端主机 H_c，这样 H_C 只能得到 m_1 的杂凑值而得不到 m_1。

9) 客户端验证服务器端 SSL 证书 $PubKCert_S$。

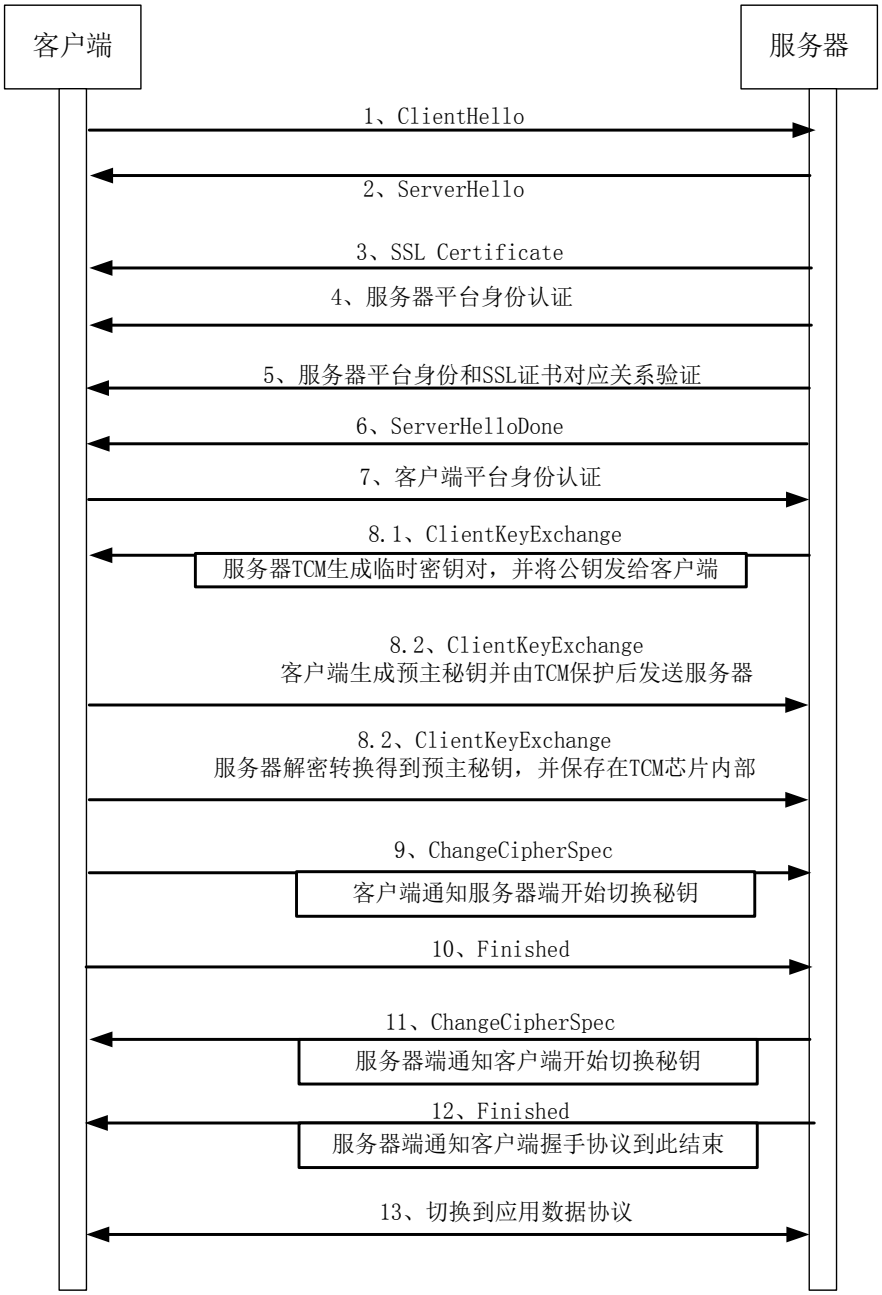

Figure 3. Flowchart after optimizing the SSL protocol
图 3. SSL 协议优化后的整体流程图

10) 客户端验证 TCM 平台身份证书 $PEKCert_S$。

11) 客户端进行 SSL 证书和 TCM 平台身份证书对应关系的认证：

计算 $hm'_{12} = adec(DS_S, K_P)$；计算 $hm_2 = hmac(m_2)$；验证方程 $hm_{12} = hmac(m_1 \| m_2)$ 是否成立，若成立则签名有效，继续以下步骤，否则终止此次协议。

12) 客户端发送 PEK_C 和 $PEKCert_C$ 给服务器，服务器验证 PEK_C 和 $PEKCert_C$。

13) 服务器发送 Server Hello Done 消息，通知 SSL 客户端版本和加密套件协商结束，开始进行密钥交换。

14) 服务器端 H_S 调用 $TCM\_CreateKeeyExchange()$；

服务器 T_S 接到调用后:

a) 创建密钥协商会话，随机生成会话句柄 shd_Y

b) 随机生成临时密钥私钥 y，计算临时密钥公钥 $Y=g^y$

c) 将临时密钥对 (y,Y) 存储在 TCM 中并与句柄 shd_Y 绑定；

T_S 返回 shd_Y， shd_Y 给 H_S， H_S 将 Y 发送给客户端 H_C。

15) 客户端 H_C 随机生成一个预主密钥 $PreMasterSecret$；

客户端 TCM 即 T_C： a) 随机生成临时密钥私钥 x，计算临时密钥公钥 $X=g^x$

b) 计算 $seed=SM2KE(PEK_C,PEK_S,Y,X)$，

$$K_1=kdf('encryption',seed),$$

$$K_2=kdf('intergrity',seed)$$

c) 计算 $a=senc(PreMasterSecret,K_1)$，

$$b=hmac(a,K_2),\ \ sblob=(a,b)$$

T_C 返回 X 和 $Sblob$ 给 H_C， H_C 将 X 和 $sblob$ 传给 H_S， T_C 同时删除临时密钥对 (x,X)。

16) 服务器端 T_S： 1)计算 $seed=SM2KE(PEK_C,PEK_S,Y,X)$，

$$K_1=kdf('encryption',seed),$$

$$K2=kdf('intergrity',seed)$$

将 $sblob$ 前 32 字节赋值给 b，其余字节赋值给 a

验证 $b=hmac(a,K_2)$，

计算 $PreMasterSecret=sdec(a,K_1)$，

用 SMK 存储主密钥保护 $PreMasterSecret=sdec(a,K_1)$：

$$keyBlob=senc(PreMasterSecret,SMK).$$

17) O_S 调用 $TCM\_ReleaseExchangeSession(shd_Y)$， T_S 接到调用后，删除 shd_Y 指向的临时密钥对 (y,Y)。

18) H_S 利用协商好的算法及参数 Ver_c、Ver_s 计算得到主密钥 $MasterSecret$，接着 H_S 计算 $Finished$ 消息的内容，及哈希值：

$$MD5(MasterSecret+Pad_2+MD5(HandshakeMessages+MasterSecret+Pad_1))$$
$$+SHA(MasterSecret+Pad_2+SHA(HandshakeMessages+MasterSecret+Pad_1))$$

公式中 $HandshakeMessages$ 表示握手消息指从客户端问候消息开始的内不包括此消息在内的到这个步骤的信息， Pad_1 和 Pad_2 为填充字节。

服务器 H_S 传送 $ChangeCiperSpec$、$Finished$ 给客户端 H_C。客户端 H_C 计算主密钥 $MasterSecret=hmac(PreMasterSecret)$， H_C 计算 $Finished$ 消息的内容：

$$Finished'=MD5(MasterSecret+Pad_2+MD5(HandshakeMessages+MasterSecret+Pad_1))$$
$$+SHA(MasterSecret+Pad_2+SHA(HandshakeMessages+MasterSecret+Pad_1))$$

比较 $Finished'$ 和 $Finished$，相同表示消息未被篡改，最后客户端也将 $Finished'$ 发给服务器表示整个

握手过程的结束。

服务器验证客户端发送的 *Finished'* 消息，如果成立，则可以分块导出 *MasterSecret* 。

6. 优化应用

SSL 协议在电子商务、网上支付、机密文件的网络传输上发挥重要作用，用来保证信息的可靠、机密、完整性的传输，优化后的 SSL 协议对上述应用的安全性的到加强。下面详细描述优化后的 SSL 协议在电子商务中的一个应用实例，在网络交易中，客户下单后必须将自己的订单信息完整地传输给商家，才可以保证交易的正常进行，在双方都安装有 TCM 芯片的前提下，在网络间传输信息时采用优化后的 SSL 协议，其具体通信过程如下：

T_C：客户端安装的 TCM 芯片；T_S：商家端安装的 TCM 芯片。

1) 客户 → 商家：$ClientHello(C, Ver\_c, Ran\_c, Suit\_c)$ 客户通过 *ClientHello* 消息将它支持的 SSL 版本、加密算法、密钥交换算法、*hmac* 算法等信息发送给商家。

2) 商家 → 客户：$ServerHello(S, Ver\_s, Ran\_s, Suit\_s)$，与 1)相对应，商家通过 *ServerHello* 消息将 Server 端支持的 SSL 版本、加密算法、密钥交换算法、*hmac* 算法等信息发送给客户。

3) 商家 → 客户：商家进行 SSL 证书认证，将携带自己公钥信息的数字证书 $PubKCert_S$ 和商家公钥 $PubK_S$ 发送给客户。

4) 商家 → 客户：商家进行平台认证，将自身的平台身份证书 $PEKCert_S$ 和对应的平台身份公钥 PEK_S 发送给客户。

5) 商家：商家封装 TCM 证书信息为 m_1：$m_1 = (PEK_S, PEKCert_S)$；封装 SSL 证书信息为 m_2：$m_2 = (PubK_S, PubKCert_S)$。

6) 商家：商家计算 m_1 的哈希值 hm_1：$hm_1 = hmac(m_1)$；计算 m_2 的哈希值 hm_2：$hm_2 = hmac(m_2)$；计算 m_1、m_2 的哈希值 hm_{12}：$hm_{12} = hmac(m_1 \| m_2)$。

7) 商家：商家生成签名密钥对 K_S、K_P，商家进行数字双签名得到 DS_S：$DS_S = aenc(m_{12}, K_{CS})$，利用数字双签名将 TCM 平台身份证书和 SSL 证书关联在一起，表名此次 SSL 协议的特定平台环境为 TCM 中的信息。

8) 商家 → 客户：商家把 K_S，m_1，DS_S 及 m_1 的哈希值 hm_1 发送给客户端 TCM 即 TCM_C，这样 TCM 只能得到 m_2 的杂凑值，但得不到 m_2；把 m_2，DS_S 及 m_2 的哈希值 hm_2 发送给客户端主机 H_C，这样 H_C 只能得到 m_1 的杂凑值而得不到 m_1。

9) 客户：客户端 TCM 进行平台身份和 SSL 证书对应关系验证：

计算 $hm'_{12} = adec(DS_S, K_P)$；计算 $hm_1 = hamc(m_1)$;验证方程 $hm_{12} = hamc(m_1 \| m_2)$ 是否成立，若成立则签名有效，继续以下步骤，否则终止此次协议。

10) 客户：客户端主机进行平台身份和 SSL 证书对应关系验证：

a) 计算 $hm'_{12} = adec(DS_S, K_P)$；计算 $hm_2 = hmac(m_2)$；

b) 验证方程 $hm_{12} = hamc(m_1 \| m_2)$ 是否成立，若成立则签名有效，继续以下步骤，否则终止此次协议。

11) 客户：客户端 TCM 验证 PEK_S 和 $PEKCert_S$。

12) 客户 → 商家：客户端发送 PEK_C 和 $PEKCert_C$ 给商家，商家验证 PEK_C 和 $PEKCert_C$。

13) 商家 → 客户：商家发送 Server Hello Done 消息，通知客户版本和加密套件协商结束，开始进行密钥交换。

14) 商家 → 客户：商家主机 H_S 调用 $TCM\_CreateKeyExchange()$；

商家 T_S 接到调用后:

a) 创建秘钥协商会话，随机生成会话句柄 shd_Y

随机生成临时秘钥私钥 y，计算临时秘钥公钥 $Y=g^y$

将临时秘钥对 (y,Y) 存储在 TCM 中并与句柄 shd_Y 绑定；

T_S 返回 Y， shd_Y 给 H_S， H_S 将 Y 发送给客户端 H_C。

15) 客户→商家：客户端主机 H_C 随机生成一个预主秘钥 $PreMasterSecret$；

客户端 TCM 即 T_C：a) 随机生成临时秘钥私钥 x，计算临时秘钥公钥 $X=g^x$；

计算 $seed=SM2KE(PEK_C,PEK_S,Y,X)$，

$$K_1=kdf('encryption',seed),$$

$$K_2=kdf('intergrity',seed);$$

计算 $a=senc(PreMasterSecret,K_1)$，

$$b=hmac(a,K_2),\ \ sblob=(a,b).$$

T_C 返回 X 和 $sblob$ 给 H_C， H_C 将 X 和 $sblob$ 传给 H_S， T_C 同时删除临时秘钥对 (x,X)。

16) 商家：商家端 TCM 即 T_S：a) 计算 $seed=SM2KE(PEK_C,PEK_S,Y,X)$，

$$K_1=kdf('encryption',seed),$$

$$K_2=kdf('intergrity',seed);$$

将 $sblob$ 前 32 字节赋值给 b，其余字节赋值给 a；

验证 $b=hamc(a,K_2)$，

计算 $PreMasterSecret=sdec(a,K_1)$，

用 SMK 存储主秘钥保护 $PreMasterSecret=sdec(a,K_1)$：

$keyBlob=senc(PreMasterSecret,SMK)$。

商家：商家端 TCM 所有者调用 $TCM\_ReleaseExchangeSession(shd_Y)$，$T_S$ 接到调用后，删除 shd_Y 指向的临时秘钥对 (y,Y)。

商家：商家主机利用协商好的算法及参数 Ver_c、Ver_s 计算得到主密钥 $MasterSecret$，接着计算 $Finished$ 消息的内容，及哈希值：

$$Finished=MD5(MasterSecret+Pad_2+MD5(HandshakeMessages+MasterSecret+Pad_1))$$
$$+SHA(MasterSecret+Pad_2+SHA(HandshakeMessages+MasterSecret+Pad_1))$$

公式中 $HandshakeMessages$ 表示握手消息指从客户端问候消息开始的内容，不包括此消息在内的到这个步骤的信息， Pad_1 和 Pad_2 为填充字节。

商家→客户：商家 H_S 传送 $ChangeCiperSpec$、$Finished$ 给客户端 H_C。客户端 H_C 计算主密钥：$MasterSecret=hmac(PreMasterSecret)$， H_C 计算 $Finished$ 消息的内容：

$$Finished'=MD5(MasterSecret+Pad_2+MD5(HandshakeMessages+MasterSecret+Pad_1))$$
$$+SHA(MasterSecret+Pad_2+SHA(HandshakeMessages+MasterSecret+Pad_1))$$

比较 $Finished'$ 和 $Finished$，相同表示消息未被篡改，最后客户端也将 $Finished'$ 发给商家表示整个握手过程的结束。

商家：商家验证客户端发送的 *Finished'* 消息，如果成立，则可以分块导出 *MasterSecret* 。

客户：客户端可以放心地使用主密钥 *MasterSecret* 加密订单信息发送给商家，商家可以用主密钥加密发送请求付款消息。

7. 优化后 SSL 协议分析

7.1. 安全性分析

安全性分析主要从身份认证、会话密钥的安全传输两方面进行分析：

1) SSL 协议身份认证安全性分析：

当有攻击者冒充服务器身份时，由于服务器端发送的身份认证信息包含：平台身份证书 $PEKCert_S$ 和平台身份公钥 PEK_S。平台身份密钥对 TCM 内部数据如寄存器、计数器值等进行数字签名，以此证明身份。攻击者不能获取到 TCM 芯片内部的上述数据，也就不能伪造服务器身份证书，从而防范身份攻击。

2) SSL 协议会话密钥安全传输分析：

密钥交互阶段，由于第一方面客户端和服务器的 TCM 模块生成临时密钥对 (x,X) 和 (y,Y)，将公钥 X 和 Y、私钥 x 、 y 的句柄 sed_X 、 sed_Y 返回给主机，将公钥 X、Y 同时发给另一方，接着以临时公钥 X、Y 和 TCM 平台身份公钥 PEK_C 、 PEK_S 为参数，采用 TCM 的 SM2 密钥协商协议 $SM2KE$ 生成会话密钥 *PreMasterSecret* 的种子 *seed* ，即 $seed = SM2KE(PRK_C, PEK_S, X, Y)$。

当攻击者试图伪造会话密钥时：

由于攻击者可能没有 TCM 芯片，很难支持 SM2 密钥协商协议，不能进行 $SM2KE(PRK_C, PEK_S, X, Y)$ 这步运算；

即使支持 SM2 密钥协商协议，由于没法截取 TCM 内部生成的临时密钥对 (x,X) 和 (y,Y)、平台身份公钥 PEK_C 和 PEK_S，同时临时密钥对用完即删除，所以不可能生成会话密钥的种子 *seed* ，也就无法伪造会话密钥。

由于散列函数单向性，获取会话密钥摘要值也不能得到会话密钥。

上述三点说明优化后的协议可以防止会话密钥阶段的攻击。

7.2. 效率分析

优化后的 SSL 协议在身份认证和会话密钥生成、存储部分都应用了可信密码模块 TCM，利用 TCM 在信息保护方面的平台身份认证、密钥生成、密钥管理方面的基于硬件的安全可信特性，虽然安全性得到很大提升，但是由于平台身份认证、密钥的生成与管理都需要时间，所以效率方面不如传统的 SSL 协议，在某些机密性要求高、时效性要求低的应用领域优化后的基于 TCM 的 SSL 协议更适用[12]。

8. 本文总结

本文对基于 TCM 的 SSL 优化分四部分做了介绍：第一部分明确了 SSL 的优化目的。第二部分主要对 SSL 协议中的握手部分的优化从以下两方面做了描述：身份认证优化、会话密钥交互阶段的优化；之后对优化后的 SSL 协议做了整体描述；第三部分介绍了优化后的 SSL 协议的一个应用实例。第四部分对优化的 SSL 协议从安全性和效率两方面做了分析。

参考文献

[1] Noorman, J., Agten, P., Daniels, W., *et al.* (2013) Sancus: Low-Cost Trustworthy Extensible Networked Devices with

a Zero-Software Trusted Computing Base. *Usenix Conference on Security*, 479-494.

[2] Liu, J., Sun, J. and Li, T. (2005) An Enhanced Remote Login Authentication with Smart Card. *IEEE Workshop on Signal Processing Systems Design and Implementation*, 2005. IEEE, 229-232.

[3] 王群, 李馥娟. 可信计算技术及其进展研究[J]. 信息安全研究, 2016, 2(9): 834-843.

[4] 徐震, 陈路, 于爱民. 可信增强 TLS 协议的设计与实现[J]. 华中科技大学学报(自然科学版), 2016, 44(3): 44-48.

[5] 李刚. 创新驱动 构筑网络强国安全保障——沈昌祥院士谈技术可信计算的创新与发展[J]. 中国信息安全, 2015(2): 46-51.

[6] 李海威, 范博, 李文锋. 一种可信虚拟平台构建方法的研究和改进[J]. 信息网络安全, 2015(1): 1-5.

[7] 王佳慧, 刘川意, 王国峰, 等. 基于可验证计算的可信云计算研究[J]. 计算机学报, 2016, 39(2): 286-304.

[8] 吴欢, 詹静, 赵勇, 等. 一种高效虚拟化多级网络安全互联机制[J]. 山东大学学报(理学版), 2016, 51(3): 98-103.

[9] 沈昌祥, 公备. 基于国产密码体系的可信计算体系框架[J]. 密码学报, 2015, 2(5): 381-389.

[10] 沈昌祥, 张焕国, 王怀民, 等. 可信计算的研究与发展[J]. 中国科学:信息科学, 2010(2): 139-166.

[11] 孙瑜, 王溢, 洪宇, 等. 可信软件基技术研究及应用[J]. 信息安全研究, 2017, 3(4): 316-322.

[12] 潘柱廷. 从失效和成本两视角看可信计算的应用[J]. 中国信息安全, 2015(2): 57-60.