

ICS 35.040

L 80

备案号：



# 中华人民共和国密码行业标准

GM/T 0011—2012

## 可信计算 可信密码支撑平台功能与接口规范

Trusted computing Functionality and interface specification of  
cryptographic support platform

2012-11-22 发布

2012-11-22 实施

国家密码管理局 发布

中华人民共和国密码

行业标准

可信计算

可信密码支撑平台功能与接口规范

GM/T 0011—2012

\*

中国标准出版社出版发行

北京市朝阳区和平里西街甲2号(100013)

北京市西城区三里河北街16号(100045)

网址 [www.spc.net.cn](http://www.spc.net.cn)

总编室:(010)64275323 发行中心:(010)51780235

读者服务部:(010)68523946

中国标准出版社秦皇岛印刷厂印刷

各地新华书店经销

\*

开本 880×1230 1/16 印张 0.00 字数 00 千字

2012年 月第一版 2012年 月第一次印刷

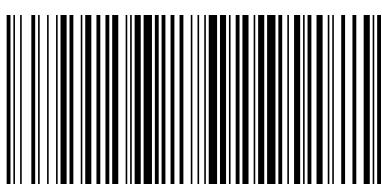
\*

书号: 155066 · 2-@ 定价 00.00 元

如有印装差错 由本社发行中心调换

版权专有 侵权必究

举报电话:(010)68510107



GM/T 0011-2012

## 目 次

前言 .....
引言 .....
1 范围 .....
2 规范性引用文件 .....
3 术语、定义和缩略语 .....
3.1 术语和定义 .....
3.2 缩略语前言 .....
4 可信计算密码支撑平台功能原理 .....
4.1 平台体系结构 .....
4.1.1 平台功能与密码的关系 .....
4.1.2 平台组成结构 .....
4.1.3 可信密码模块 .....
4.1.4 TCM 服务模块 .....
4.2 密码算法要求 .....
4.2.1 SM2 .....
4.2.1.1 系统参数 .....
4.2.1.2 密钥对生成 .....
4.2.1.3 SM2-1 数字签名算法 .....
4.2.1.4 SM2-2 密钥交换协议 .....
4.2.1.5 SM2-3 公钥加密算法 .....
4.2.2 SM3 .....
4.2.3 HMAC .....
4.2.4 SM4 .....
4.2.4.1 算法简介 .....
4.2.4.2 使用模式 .....
4.2.5 随机数发生器 .....
4.3 功能原理 .....
4.3.1 平台完整性 .....
4.3.1.1 概述 .....
4.3.1.2 完整性度量、存储与报告 .....
4.3.1.3 信任链 .....
4.3.1.4 向外部实体证实平台完整性 .....
4.3.2 平台身份可信 .....
4.3.2.1 平台身份标识 .....
4.3.2.2 平台身份建立 .....
4.3.3 平台数据安全保护 .....
4.3.3.1 密钥管理功能 .....

4.3.3.2	数据安全保护方式前言
5	可信计算密码支撑平台功能接口
5.1	概述
5.2	上下文管理
5.2.1	概述
5.2.2	创建上下文
5.2.3	关闭上下文
5.2.4	设置上下文属性(整型参数)
5.2.5	获取上下文属性(整型参数)
5.2.6	设置上下文属性(变长参数)
5.2.7	获取上下文属性(变长参数)
5.2.8	连接上下文
5.2.9	释放上下文
5.2.10	获取上下文默认策略
5.2.11	创建对象
5.2.12	关闭对象
5.2.13	获取平台功能特性
5.2.14	获取 TCM 对象句柄
5.2.15	通过密钥属性加载密钥
5.2.16	通过密钥 ID 加载密钥
5.2.17	注册密钥
5.2.18	销毁密钥
5.2.19	通过密钥 ID 获取密钥
5.2.20	通过公钥获取密钥
5.2.21	通过 ID 获取注册密钥
5.2.22	设置传输会话加密密钥
5.2.23	关闭传输会话
5.3	策略管理
5.3.1	设置策略类属性(整型参数)
5.3.2	获取上下文属性(整型参数)
5.3.3	设置上下文属性(变长参数)
5.3.4	获取上下文属性(变长参数)
5.3.5	设置策略授权
5.3.6	清除策略授权
5.3.7	绑定策略对象
5.4	可信密码模块(TCM)管理
5.4.1	概述
5.4.2	创建平台身份和证书请求
5.4.3	激活平台身份和获取 PIK 证书
5.4.4	创建 PEK 请求
5.4.5	获取 PEK 证书
5.4.6	导入 PEK 密钥

5.4.7	创建不可撤消的密码模块密钥	.....
5.4.8	获取密码模块密钥公钥	.....
5.4.9	创建可撤销的密码模块密钥	.....
5.4.10	撤销密码模块密钥	.....
5.4.11	创建密码模块所有者	.....
5.4.12	清除可信密码模块所有者	.....
5.4.13	设置操作者授权	.....
5.4.14	设置可信密码模块状态	.....
5.4.15	查询设置可信密码模块状态	.....
5.4.16	获取可信密码模块特性	.....
5.4.17	可信密码模块完全自检	.....
5.4.18	获取可信密码模块自检结果	.....
5.4.19	获取可信密码模块产生的随机数	.....
5.4.20	获取可信密码模块单个事件	.....
5.4.21	获取可信密码模块一组事件	.....
5.4.22	获取可信密码模块事件日志	.....
5.4.23	可信密码模块 PCR 扩展	.....
5.4.24	读取可信密码模块 PCR 值	.....
5.4.25	重置可信密码模块 PCR	.....
5.4.26	引证 PCR	.....
5.4.27	读可信密码模块计数器	.....
5.4.28	读可信密码模块当前时钟	.....
5.4.29	获取可信密码模块审计摘要值	.....
5.4.30	设置可信密码模块命令审计状态	.....
5.5	密钥管理	.....
5.5.1	概述	.....
5.5.2	改变实体授权数据	.....
5.5.3	获取策略对象	.....
5.5.4	设置密钥属性(整型参数)	.....
5.5.5	获取密钥属性(整型参数)	.....
5.5.6	设置密钥属性(变长参数)	.....
5.5.7	获取设置密钥属性(变长参数)	.....
5.5.8	加载密钥	.....
5.5.9	卸载密钥	.....
5.5.10	获取密钥公钥	.....
5.5.11	签署密钥	.....
5.5.12	创建密钥	.....
5.5.13	封装密钥	.....
5.5.14	创建迁移授权	.....
5.5.15	创建迁移密钥数据块	.....
5.5.16	导入迁移密钥数据块	.....
5.6	数据加密与解密	.....

5.6.1	改变实体授权	.....
5.6.2	获取策略对象	.....
5.6.3	获取数据属性(整型参数)	.....
5.6.4	设置数据属性(变长参数)	.....
5.6.5	获取数据属性	.....
5.6.6	数据加密	.....
5.6.7	数据解密	.....
5.6.8	数据封装	.....
5.6.9	数据解封	.....
5.6.10	数字信封封装	.....
5.6.11	数字信封解密	.....
5.7	PCR 管理	.....
5.7.1	概述	.....
5.7.2	设置 PCR Locality 属性	.....
5.7.3	获取 PCR Locality 属性	.....
5.7.4	获取 PCR 摘要	.....
5.7.5	设置 PCR 值	.....
5.7.6	获取 PCR 值	.....
5.7.7	选择 PCR 索引	.....
5.7.8	非易失性存储管理	.....
5.7.9	设置非易失性存储区属性(整型参数)	.....
5.7.10	获取非易失性存储区属性(整型参数)	.....
5.7.11	获取非易失性存储区属性(变长参数)	.....
5.7.12	创建非易失性存储区空间	.....
5.7.13	释放非易失性存储区空间	.....
5.7.14	数据写入非易失性存储区	.....
5.7.15	从非易失性存储区读取数据	.....
5.8	杂凑操作	.....
5.8.1	设置杂凑对象属性(整型参数)	.....
5.8.2	获取杂凑对象属性(整型参数)	.....
5.8.3	设置杂凑对象属性(变长参数)	.....
5.8.4	对用户数据进行杂凑操作	.....
5.8.5	设置杂凑值	.....
5.8.6	获取杂凑值	.....
5.8.7	更新杂凑值	.....
5.8.8	对杂凑值签名	.....
5.8.9	验证杂凑值签名	.....
5.8.10	给杂凑类加时间戳	.....
5.9	密钥协商	.....
5.9.1	创建会话	.....
5.9.2	获取会话密钥	.....
5.9.3	释放会话	.....

附录 A (规范性附录) 接口规范数据结构 .....
附录 B (规范性附录) 数字证书格式 .....
附录 C (规范性附录) 主板应用接口 .....
参考文献 .....

## 前　　言

本标准依据 GB/T 1.1—2009 给出的规则起草。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

本标准由国家密码管理局提出并归口。

本标准起草单位：联想控股有限公司、国民技术股份有限公司、同方股份有限公司、中国科学院软件所、北京兆日技术有限责任公司、瑞达信息产业股份有限公司、长春吉大正元信息技术股份有限公司、方正科技集团股份有限公司、中国长城计算机深圳股份有限公司、成都卫士通信息产业股份有限公司、无锡江南信息安全工程技术中心、中国人民解放军国防科学技术大学。

本标准主要起草人：吴秋新、杨贤伟、邹浩、余发江、宁晓魁、王梓、郑必可、林洋、李伟平、尹洪兵、徐震、严飞、刘韧、李丰、许勇、贾兵、王蕾、顾健、何长龙、秦宇、刘鑫、王正鹏、范琴。

## 引　　言

本标准以我国可信计算密码技术要求与应用方案为指导,描述了可信计算密码支撑平台的功能原理与要求,并定义了可信计算密码支撑平台为应用层提供服务的接口规范。用以指导我国相关可信计算产品开发和应用。

本标准凡涉及密码算法相关内容,按照国家有关规定实施。

# 可信计算 可信密码支撑平台功能与接口规范

## 1 范围

本标准描述可信计算密码支撑平台功能原理与要求，并详细定义了可信计算密码支撑平台的密码算法、密钥管理、证书管理、密码协议、密码服务等应用接口规范。

本标准适用于可信计算密码支撑平台相关产品的研制、生产、测评与应用开发。

## 2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T 5271.8 信息系统 词汇 第8部分：安全（eqv ISO/IEC 2382-8—1998）

GM/T 0002 SM4 分组密码算法

GM/T 0003 SM2 椭圆曲线公钥密码算法

GM/T 0004 SM3 密码杂凑算法

GM/T 0005 随机性检测规范

GM/T AAAA SM2 密码算法使用规范

GM/T BBBB 基于 SM2 密码算法的数字证书格式规范

## 3 术语、定义和缩略语

### 3.1 术语和定义

GB/T 5271.8 中界定的以及下列术语和定义适合本文件。

#### 3.1.1 部件 component

计算系统中可被度量的硬件和/或软件模块。

#### 3.1.2 存储主密钥 storage master key

用于保护平台身份密钥和用户密钥的主密钥。

#### 3.1.3 对象 object

可信计算密码支撑平台内可以被实体访问的各类资源，包括密钥数据、运行环境数据、敏感数据等。

#### 3.1.4 可信计算平台 trusted computing platform

构建在计算系统中，用于实现可信计算功能的支撑系统。

#### 3.1.5 可信计算密码支撑平台 cryptographic support platform for trusted computing

可信计算平台的重要组成部分，包括密码算法、密钥管理、证书管理、密码协议、密码服务等内容，为

可信计算平台自身的完整性、身份可信性和数据安全性提供密码支持。其产品形态主要表现为可信密码模块和可信密码服务模块。

3.1.6

**可信度量根 root of trust for measurement**

一个可信的完整性度量单元,是可信计算平台内进行可信度量的基础。

3.1.7

**可信存储根 root of trust for storage**

指存储主密钥,是可信计算平台内进行可信存储的基础。

3.1.8

**可信报告根 root of trust for reporting**

指密码模块密钥,是可信计算平台内进行可信报告的基础。

3.1.9

**可信密码模块 trusted cryptography module**

是可信计算平台的硬件模块,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。

3.1.10

**TCM 服务模块 TCM service module**

可信计算密码支撑平台内部的软件模块,为对平台外部提供访问可信密码模块的软件接口。

3.1.11

**可信方 trusted party**

提供可信证明的机构,包含可信第三方和主管方。

3.1.12

**密码模块密钥 TCM endorsement key**

可信密码模块的初始密钥。

3.1.13

**密钥管理中心 key management centre**

用于密钥生成和管理的软硬件系统。

3.1.14

**平台配置寄存器 platform configuration register**

可信密码模块内部用于存储平台完整性度量值的存储单元。

3.1.15

**平台身份密钥 platform identity key**

可信密码模块的身份密钥。

3.1.16

**平台加密密钥 platform encryption key**

可信密码模块中与平台身份密钥关联的加密密钥。

3.1.17

**双证书 dual certificate**

包括签名证书和加密证书,分别用于签名和数据加密,由可信方一同签发。

3.1.18

**实体 entity**

访问密码支撑平台资源的应用程序和用户。

## 3.1.19

**完整性度量 integrity measurement**

使用密码杂凑算法对被度量对象计算其杂凑值的过程。

## 3.1.20

**完整性度量值 integrity measurement value**

部件被度量后得到的杂凑值。

## 3.1.21

**完整性基准值 predefined integrity value**

部件在可信状态下被度量得到的杂凑值。该值可作为完整性校验的基准。

## 3.1.22

**信任链 trusted chain**

在计算系统启动和运行过程中,使用完整性度量方法在部件之间所建立的信任传递关系。

## 3.2 缩略语

下列缩略语适用于本文件

EK	密码模块密钥	TCM endorsement key
KMC	密钥管理中心	key management centre
HMAC	带密钥的消息验证码算法	the keyed-hash message authentication code
NV	非易失性	non-volatility
PCR	平台配置寄存器	platform configuration register
PEK	平台加密密钥	platform encryption key
PIK	平台身份密钥	platform identity key
SMK	存储主密钥	storage master key
TCM	可信密码模块	trusted cryptography module
TSM	TCM 服务模块	TCM service module

## 4 可信计算密码支撑平台功能原理

## 4.1 平台体系结构

## 4.1.1 平台功能与密码的关系

平台功能与密码的关系如下:

## a) 平台完整性

利用密码机制,通过对系统平台组件的完整性度量,确保系统平台完整性,并向外部实体可信地报告平台完整性。

## b) 平台身份可信

利用密码机制,标识系统平台身份,实现系统平台身份管理功能,并向外部实体提供系统平台身份证明。

## c) 平台数据安全保护

利用密码机制,保护系统平台敏感数据。其中数据安全保护包括平台自身敏感数据的保护和用户敏感数据的保护。另外也可为用户数据保护提供服务接口。

密码对平台功能的支撑关系如图 1 所示:

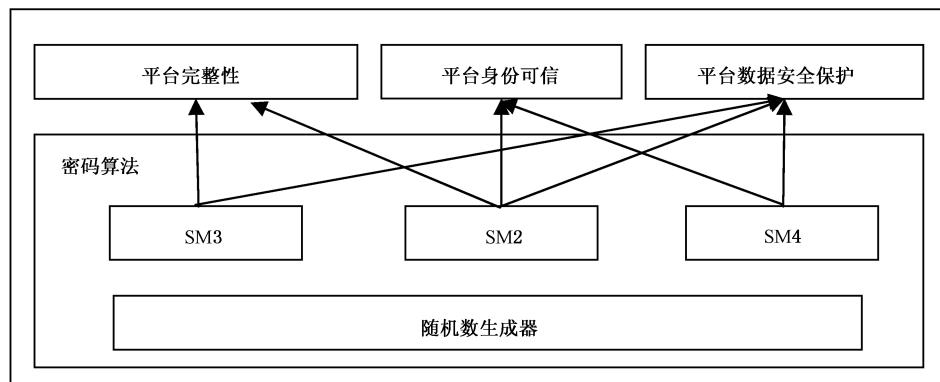


图 1 平台功能与密码的关系

#### 4.1.2 平台组成结构

可信计算平台主要由 TCM 和 TSM 两大部分组成,其功能架构如图 2 所示。

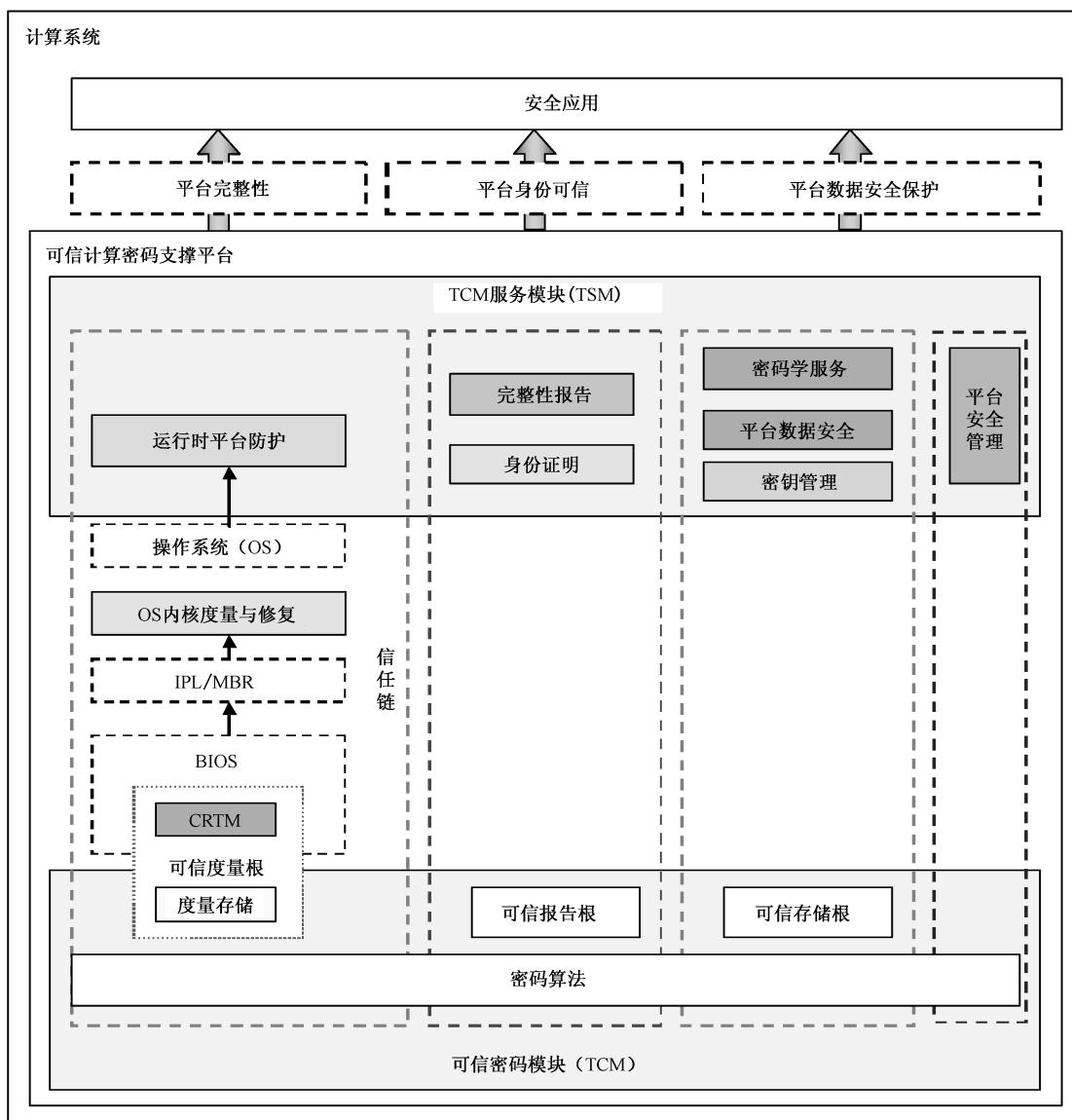


图 2 可信计算密码支撑平台功能架构

可信计算密码支撑平台以可信密码模块为可信根,通过如下三类机制及平台自身安全管理功能,实现平台安全功能:

- 以可信度量根为起点,计算系统平台完整性度量值,建立计算机系统平台信任链,确保系统平台可信。
- 可信报告根标识平台身份的可信性,具有唯一性,以可信报告根为基础,实现平台身份证明和完整性报告。
- 基于可信存储根,实现密钥管理、平台数据安全保护功能,提供相应的密码服务。

#### 4.1.3 可信密码模块

TCM 是可信计算密码支撑平台必备的关键基础部件,提供独立的密码算法支撑。TCM 是硬件和固件的集合,可以采用独立的封装形式,也可以采用 IP 核的方式和其他类型芯片集成在一起,提供 TCM 功能。其基本组成结构如图 3 所示。

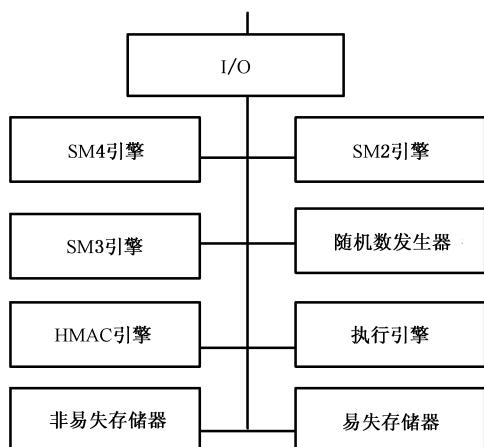


图 3 可信密码模块结构

图 3 中各部件说明如下：

I/O: TCM 的输入输出硬件接口；

SM4 引擎: 执行 SM4 对称密码运算的单元；

SM2 引擎: 产生 SM2 密钥对和执行 SM2 加/解密、签名运算的单元；

SM3 引擎: 执行杂凑运算的单元；

随机数发生器: 生成随机数的单元；

HMAC 引擎: 基于 SM3 引擎的计算消息认证码单元；

执行引擎: TCM 的运算执行单元；

非易失性存储器: 存储永久数据的存储单元；

易失性存储器: TCM 运行时临时数据的存储单元。

#### 4.1.4 TCM 服务模块

可信密码模块定义了一个具有存储保护和执行保护的子系统，该子系统将为计算平台建立信任根基，并且其独立的计算资源将建立严格受限的安全保护机制。为防止 TCM 成为计算平台的性能瓶颈，将子系统中需执行保护的函数与无需执行保护的函数划分开，将无需执行保护的功能函数由计算平台主处理器执行，而这些支持函数构成了 TCM 服务模块，简记为 TSM。

TSM 主要为用户使用 TCM 基础资源提供支持，由多个部分组成，每个部分间的接口定义应具有互操作性。TSM 应提供规范化的函数接口。

TSM 设计目标：

- 为应用程序调用 TCM 安全保护功能提供一个入口点；
- 提供对 TCM 的同步访问；
- 向应用程序隐藏 TCM 所建立的功能命令；
- 管理 TCM 资源。

#### 4.2 密码算法要求

本标准涉及的密码算法包括：SM2 非对称密码算法、SM4 对称密码算法、SM3 密码杂凑算法、HMAC 消息认证码算法、随机数发生器。上述密码算法必须符合国家密码管理局管理要求并在可信密码模块内实现。

#### 4.2.1 SM2

本标准采用的椭圆曲线密码算法为 SM2,其密钥位长为  $m(m=256)$ 。

SM2 算法的详细内容见 GM/T 0003,该算法的使用见 GM/T AAAA。

#### 4.2.2 SM3

本标准采用的密码杂凑算法为 SM3。对于给定的长度为  $k(k < 2^{64})$  的消息,SM3 密码杂凑算法经过填充、迭代压缩和选裁,生成杂凑值。经预处理过的消息分组长度为 512 比特,本标准选用的杂凑值长度为 256 比特,SM3 算法的详细内容见 GM/T 0004。

对分组后的消息迭代压缩:

输入参数:

$M$ :512 比特长度的消息

输出参数:

$D$ :256 比特长度的摘要

#### 4.2.3 HMAC

本标准采用的 HMAC 是利用密码杂凑算法 SM3 对于给定的消息和验证双方共享的秘密信息产生长度为  $t$  个字节的消息验证码,消息验证码产生过程参考 FIPS PUB 198 中的消息验证码产生过程。

输入参数:

$text$ :消息

$Secret\_K$ :双方共享的秘密信息

$K\_len$ :双方共享的秘密信息长度

$t$ :消息验证码的长度,不小于 16 字节,不大于 32 字节

输出参数:

$digest$ : $t$  个字节的消息验证码

算法描述:

步骤 1:如果  $K\_len = Block\_Len$ ,则  $K_0 := Secret\_K$ ,跳转执行步骤 4;

步骤 2:如果  $K\_len > Block\_Len$ ,则对  $K$  进行杂凑运算得到  $L$  个字节,然后拼接上

$Block\_Len - L$  个字节 0 得到  $Block\_Len$  个字节的  $K_0$ ,即  $K_0 := SM3(K) \parallel 00\cdots00$ ,跳转执行步骤 4;

步骤 3:如果  $K\_len < Block\_Len$ ,则对  $Secret\_K$  后面拼接字节 0 得到  $Block\_Len$  个字节的  $K_0$ ,执行步骤 4;

步骤 4: $K_0 \oplus ipad$ ;

步骤 5:在步骤 4 的结果后面拼接  $text$ ,即  $(K_0 \oplus ipad) \parallel text$ ;

步骤 6:对步骤 5 的结果进行杂凑运算,即  $SM3((K_0 \oplus ipad) \parallel text)$ ;

步骤 7: $K_0 \oplus opad$ ;

步骤 8:把步骤 6 的结果拼接在步骤 7 的结果后面,即

$(K_0 \oplus opad) \parallel SM3((K_0 \oplus ipad) \parallel text)$ ;

步骤 9:对步骤 8 的结果进行杂凑运算,即

$SM3((K_0 \oplus opad) \parallel SM3((K_0 \oplus ipad) \parallel text))$ ;

步骤 10:把步骤 9 结果的高  $t$  个字节作为结果输出。

#### 4.2.4 SM4

##### 4.2.4.1 算法简介

本标准采用的对称密码算法为 SM4。该算法是一个分组算法,该算法的分组长度为 128 比特,密钥长度为 128 比特。加密算法与密钥扩展算法都采用 32 轮非线性迭代结构。解密算法与加密算法的结构相同,只是轮密钥的使用顺序相反,解密轮密钥是加密轮密钥的逆序,SM4 算法的详细内容见 GM/T 0002。

###### 加密算法

输入参数:

$M$ :128 比特长度的明文信息

$K$ :128 比特长度的密钥信息

输出参数:

$C$ :128 比特长度的密文信息

###### 解密算法

输入参数:

$C$ :128 比特长度的密文信息

$K$ :128 比特长度的密钥信息

输出参数:

$M$ :128 比特长度的明文信息

##### 4.2.4.2 使用模式

本标准规定采用 CBC 模式,IV 由用户自定义。

对于数据的最后一个分组,数据需要填充,方法如下:

- a) 填充完的数据长度必须是 16 的整数倍;
- b) 如果最后一个分组缺少  $d_1$  个字节( $0 < d_1 < 16$ ),则在该组后面填充  $d_1$  个字节,每个字节内容均是  $d_1$ ;
- c) 如果  $d_1=0$  则填充 16 个字节,每个字节内容均是 16。

#### 4.2.5 随机数

本标准不规定随机数生成的具体算法,随机数生成算法由可信密码模块制造商设计实现。所生成的随机数必须为真随机数,并符合 GM/T 0005 的要求。

#### 4.3 功能原理

##### 4.3.1 平台完整性

###### 4.3.1.1 概述

可信计算密码支撑平台利用密码机制,通过对系统平台组件的完整性度量、存储与报告,确保平台完整性。

###### 4.3.1.2 完整性度量、存储与报告

完整性度量与存储是指计算部件的度量值,记录该事件到事件日志,并把度量值记入可信密码模块内相应的平台配置寄存器(PCR)中。

图 4 为依次度量两个部件的过程：

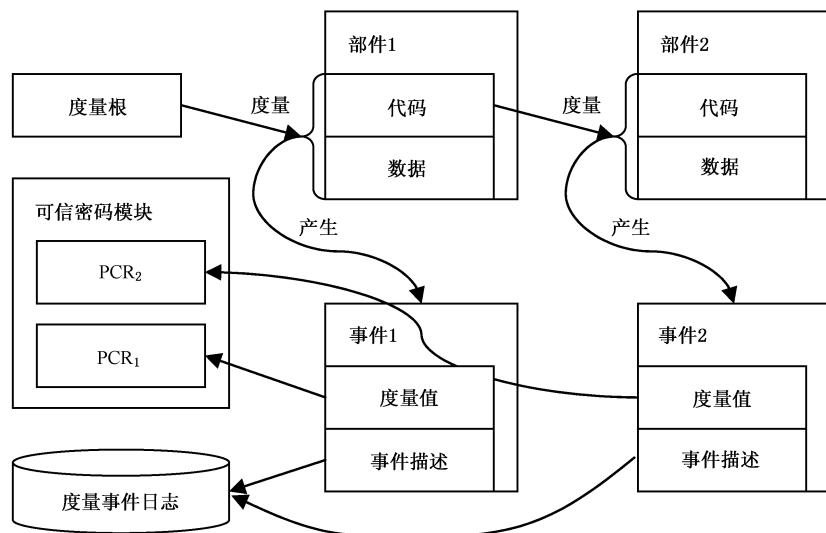


图 4 完整性度量流程

完整性度量与存储应满足如下要求：

- 计算度量值的过程应是执行杂凑运算的过程；
- 杂凑运算输入的数据应为度量者指定的可以表征被度量者特性的数据；
- 杂凑运算输出的杂凑值即为被度量者的完整性度量值；
- 度量者应把度量值记入指定的 PCR 中。记入的办法是：新 PCR 值 = 密码杂凑算法(原 PCR 值 || 度量值)；
- 应把度量过程信息记录到平台事件日志中。至少应记录：度量者信息、被度量者信息、原 PCR 值、度量值、新 PCR 值、完成时间等；
- 如果一个部件序列中的各部件完整性度量值存储在同一个 PCR 中，则采用一种专门的压缩存储方式，即从第一个部件开始，将该部件完整性度量值与目标 PCR 的已有存储值拼接，进行杂凑运算，然后将所得结果再存储于该 PCR 中，依次类推，最后一个部件的完整性度量值存储操作完成后，所得值即为该部件序列存储到 PCR 中的完整性度量值。

完整性报告是指平台向验证者提供平台或部分部件的完整性度量值的过程。

完整性报告应满足如下要求：

- 平台能够向验证者提供指定 PCR 值，无需任何授权；
- 平台能够向验证者提供指定 PCR 值以及对 PCR 值的签名。签名使用平台身份密钥；
- 平台可向验证者提供指定 PCR 的相关事件日志信息；
- 验证者可通过分析完整性度量事件日志信息判断该 PCR 值是否来自正确的度量过程；
- 验证者应使用平台身份密钥验证 PCR 值签名，获得平台完整性报告结果。

#### 4.3.1.3 信任链

信任链可以保障平台完整性。以 PC 为例，平台信任链的建立是以可信度量根为起点，首先对 BIOS 的其它部件进行完整性度量，并将度量值存储于可信密码模块的 PCR 中，按照选择的判断机制判断 BIOS 的完整性，若完整性未被破坏，则运行 BIOS；并度量初始化程序加载器(Initial Program Loader—IPL)/主引导分区(Master Boot Record—MBR)的完整性，基于判断机制判断 IPL/MBR 完整性，若 IPL/MBR 完整性没有被破坏，则运行 MBR；然后由 IPL/MBR 度量 OS 内核的度量与修复部件

的完整性,若该部件完整性未被破坏,则由该部件度量 OS 内核的完整性,若未被篡改,则运行 OS 内核。OS 内核启动后基于同样机制检测 OS 服务完整性,通过信任关系传递,可以确保所启动的系统是可信的。系统运行时可以设置一个平台防护模块,用于检测应用程序或进程的完整性,确保应用程序或进程的可信性。

若上述过程中,发现某一部件的完整性受到破坏,则报告问题并按照指定策略执行相关操作。

#### 4.3.1.4 向外部实体证实平台完整性

平台可向外部实体提供完整性报告,所报告的度量值作为判断平台可信性的依据。

报告完整性度量值时,平台身份密钥应对完整性度量值进行数字签名,接收方通过验证签名有效性以及校验完整性度量值来判断该平台的可信性。

外部实体可以向平台请求验证平台的完整性。一个参考的验证流程如下:

a) 平台按如下方式报告其完整性

- 1) 平台启动后,外部实体向平台发送完整性度量报告的请求;
- 2) 可信密码模块收集 PCR 的值,使用平台身份密钥(PIK)对 PCR 的值进行签名;
- 3) 平台将 PCR 的值,PIK 对 PCR 值的签名和 PIK 证书发送给验证者。

b) 验证者验证平台完整性

- 1) 验证者得到平台发送的 PCR 值,PIK 对 PCR 值的签名和 PIK 证书;
- 2) 验证者验证 PIK 证书;
- 3) 验证者验证 PCR 值的签名;
- 4) 验证者对 PCR 的值与平台的完整性基准值进行比较,若相同,则表明当前平台处于可信状态。

#### 4.3.2 平台身份可信

##### 4.3.2.1 平台身份标识

可信计算密码支撑平台采用 EK 标识其身份,在平台所有者授权下,在 TCM 内部生成一个 SM2 密钥对,作为 PIK,用于对 TCM 内部的信息进行数字签名,实现平台身份认证和平台完整性报告,从而向外部证实平台内部数据的可信性。

密码模块密钥(EK)是唯一的,必须被保存在 TCM 内,仅仅在获取平台所有者操作及申请平台身份证书时使用,且不得被导出 TCM 外部。

密码模块证书在平台使用前由一个可信方签署,确保其可信性,用于建立密码模块密钥与可信密码模块的一一对应关系。

一个可信计算密码支撑平台可以产生多个 PIK,每个 PIK 均与 EK 绑定,对外代表平台身份。

平台身份证书及其对应的平台加密证书由可信方提供,用于验证平台身份密钥(PIK)私钥对 PCR 值的签名。

密码模块证书、平台身份证书及其对应的平台加密证书格式见附录 B。

##### 4.3.2.2 平台身份建立

###### 4.3.2.2.1 获得平台所有权

生成 EK 后,才可以创建平台所有者,取得平台所有权的操作步骤如下:

- a) 所有者输入口令,对输入数据进行长度规一化处理得到平台所有者的授权数据;
- b) 使用密码模块密钥的公钥对平台所有者授权数据进行加密并植入到可信密码模块中,在可信密码模块内部,使用密码模块密钥的私钥,对加密的授权数据进行解密,得到长度规一化的平

- 台所有者授权数据并存储于可信密码模块内部；
- c) 所有者输入存储主密钥口令,对输入数据进行长度规一化处理得到存储主密钥的授权数据；
  - d) 使用密码模块密钥的公钥对存储主密钥授权数据进行加密并植入到可信密码模块中,在可信密码模块内部,使用密码模块密钥的私钥,对加密的授权数据进行解密,得到长度规一化的存储主密钥授权数据并存储于可信密码模块内部；
  - e) 可信密码模块采用对称密码算法 SM4 生成存储主密钥,将其存储于可信密码模块内部；
  - f) 使用随机数发生器产生平台验证信息,存储于可信密码模块内部,不允许外部实体访问,在平台所有权有效期内平台验证信息不能被改变。平台验证信息主要用于数据封装。

#### 4.3.2.2.2 平台身份密钥的生成与激活

平台身份密钥生成及其证书签署与激活的步骤如下：

- a) 平台身份密钥的生成：
  - 1) 可信密码模块验证平台所有者的授权数据和存储主密钥的授权数据；
  - 2) 设置平台身份密钥授权数据,使用密码模块密钥的公钥对授权数据进行加密并植入到可信密码模块中,在可信密码模块内部,使用密码模块密钥的私钥,对加密的授权数据进行解密,得到长度规一化的平台身份密钥授权数据并存储于可信密码模块内部；
  - 3) 可信密码模块采用 SM2 密钥生成算法产生平台身份密钥；
  - 4) 可信密码模块使用存储主密钥加密平台身份密钥的私钥部分；
  - 5) 对可信方的公钥进行杂凑运算,获得可信方公钥杂凑值；
  - 6) 可信密码模块使用平台身份密钥的私钥,采用 SM2 签名算法对可信方公钥杂凑值和平台身份密钥的公钥进行签名,获得平台身份密钥签名；
  - 7) 可信密码模块将平台身份密钥的公钥,密码模块密钥的公钥和平台身份密钥签名发送给可信方。
- b) 可信方接收到平台发送的信息后,签署平台身份证书：
  - 1) 验证平台身份密钥的签名；
  - 2) 验证通过,则使用 SM2 签名算法签署平台身份证书；
  - 3) 随机生成对称加密密钥 SessionKey,并采用 SM4 对称密码算法加密平台身份证书,得到加密的平台身份证书；
  - 4) 采用以下步骤创建平台身份密钥证书与可信密码模块的匹配关系：  
使用密码杂凑算法计算平台身份密钥的公钥的杂凑值。使用密码模块密钥的公钥,对平台身份密钥的公钥的杂凑值和 SessionKey 进行加密,将加密结果和加密的平台身份证书发送给平台。
- c) 激活 PIK 证书的过程表示为：
  - 1) 可信密码模块使用密码模块密钥的私钥,解密得到平台身份密钥的公钥的杂凑值和随机生成的对称加密密钥；
  - 2) 判断所得到的平台身份密钥的公钥的杂凑值是否与该平台的平台身份密钥的公钥的杂凑值相同；
  - 3) 若相同,则可信平台使用所得到的对称加密密钥解密被加密的平台身份证书,获得平台身份证书。

#### 4.3.2.2.3 平台身份密钥的使用

平台身份密钥代表平台身份,用于对可信计算密码支撑平台内部产生的数据进行签名。

#### 4.3.3 平台数据安全保护

数据安全保护是可信计算密码支撑平台的核心功能之一,数据安全保护就是通过密钥对数据采用特定的保护方式进行处理,基本单元包括:密钥、数据、数据保护方式。

- a) 用于数据安全保护的密钥分为对称密钥和非对称密钥。密钥管理包括:密钥生成、密钥加载、密钥销毁、密钥导入、密钥迁移、密钥协商等功能;
- b) 被保护的数据可以是任何数据;
- c) 数据安全保护方式包括:数据加解密、数据封装、数字信封等方式。

##### 4.3.3.1 密钥管理功能

###### 4.3.3.1.1 密钥生成

密钥生成是指由应用层软件设置所需生成密钥的密钥属性、密钥使用授权、密钥迁移授权、密钥的保护操作密钥,并发送给可信密码模块生成指定的密钥。在可信密码模块内,由保护操作密钥加密所生成的密钥私钥部分,然后将生成的密钥数据结构返回给应用层软件。

对于可信密码模块内各类密钥,生成方法包括:

- a) 密码模块密钥由厂商生成,为SM2非对称密钥。密码模块密钥可以定义为不可撤销和可撤销两种类型;
- b) 存储主密钥必须在获取平台所有权时生成,为SM4对称密钥,平台所有者生成存储主密钥时,必须在可信密码模块内部生成;
- c) 平台身份密钥为SM2非对称密钥,必须在可信密码模块内部生成。需向可信方申请相应平台身份证书并激活该密钥;
- d) 平台加密密钥由KMC生成。可信密码模块向可信方发送PEK证书请求,获取加密的PEK证书(包含PEK公私钥信息),然后解密获得PEK及其证书,并激活该PEK;
- e) 用户密钥可以在可信密码模块内部生成,也可在可信密码模块外部生成后导入。用户密钥可以是对称密钥或非对称密钥,其属性设置见附录A。

###### 4.3.3.1.2 密钥加载

密钥生成后,在应用层软件使用该密钥进行数据安全保护操作时,如果需要使用该密钥的私钥,需要将密钥数据(TSM\_KEY\_STRUCT\_KEY)加载到可信密码模块内部,由保护操作密钥解密后才能使用。如果使用该密钥的公钥,则在应用层软件直接使用。

可信密码模块内各类密钥的加载方法包括:

- a) 使用可信密码模块密钥公钥在设置平台所有者之前不需要验证授权,设置平台所有者之后,必须验证所有者授权;密码运算过程必须在可信密码模块内部进行;
- b) 使用存储主密钥必须验证所有者授权、存储主密钥授权;密码运算过程必须在可信密码模块内部进行;
- c) 使用平台身份密钥必须验证平台身份密钥授权、存储主密钥授权;平台身份密钥的私钥必须被加载到可信密码模块内部进行密码运算,公钥的密码运算过程在模块外部进行;
- d) 使用平台加密密钥必须验证平台加密密钥授权、存储主密钥授权;平台加密密钥的私钥必须被加载到可信密码模块内部进行密码运算,公钥的密码运算过程在模块外部进行;
- e) 使用用户密钥必须验证用户密钥授权;用户密钥的私钥必须被加载到可信密码模块内部进行密码运算。

#### 4.3.3.1.3 密钥销毁

密钥生成后,应用层软件可以销毁指定的密钥。对于可信密码模块内各类密钥,销毁方法如下:

- a) 销毁密码模块密钥是通过撤销可撤销的密码模块密钥完成的。对于不可撤销的密码模块密钥,不能销毁;
- b) 销毁存储主密钥通过清除所有者来完成;
- c) 销毁平台身份密钥必须验证平台身份密钥授权、存储主密钥授权后,在可信密码模块内执行;
- d) 销毁平台加密密钥必须验证平台加密密钥授权、存储主密钥授权后,在可信密码模块内执行;
- e) 销毁用户密钥需验证其保护操作密钥授权后,在 TCM 服务模块内执行。

#### 4.3.3.1.4 密钥导入

应用层软件可以采用密钥导入方式,对可信密码模块外部创建的 SM2、SM4 密钥进行保护,并纳入到可信密码模块的密钥体系中。

密钥导入方法:外部密钥需按照指定格式设定密钥属性,并使用指定的保护操作密钥对被导入密钥的私钥部分加密。

#### 4.3.3.1.5 密钥迁移

为保证用户数据安全,需要把相应的用户密钥予以迁移备份。迁移时需要保证被迁移密钥的机密性和完整性。平台内只有用户密钥可以迁移,进行密钥迁移时需确保目标平台是可信计算平台。

假定要将平台 A 的一个可迁移密钥迁移到平台 B 上,其迁移方法描述如下:

- a) 平台 B 将保护操作密钥的公钥发送给平台 A;
- b) 平台 A 将被迁移的密钥加载到可信密码模块中,解密出被迁移的密钥;
- c) 平台 A 的可信密码模块随机产生一个对称加密密钥,使用该密钥加密被迁移的密钥,生成被迁移密钥的迁移数据;
- d) 用平台 B 的保护操作密钥的公钥加密对称加密密钥;
- e) 将迁移数据和已加密的对称加密密钥从平台 A 传递到平台 B;
- f) 平台 B 使用其保护操作密钥的私钥解出对称加密密钥,使用 SM4 对称密码算法,用解密出的对称加密密钥解密迁移数据,得到被迁移密钥;
- g) 平台 B 将被迁移密钥重新加密保护,完成整个迁移过程。

#### 4.3.3.1.6 密钥协商

密钥协商是在两个用户 A 和 B 之间建立一个共享秘密密钥的过程,通过这种方式能够确定一个共享秘密密钥的值,用来提供 A 用户与 B 用户之间数据安全保护。

密钥协商方法见 GM/T AAAA。

#### 4.3.3.2 数据安全保护方式

##### 4.3.3.2.1 数据加解密

数据加解密可以采用对称密码算法和非对称加密算法实现。非对称加密操作可在 TCM 服务模块执行,非对称密钥解密和对称加解密必须在可信密码模块内部执行。

在可信密码模块内部进行加解密操作时,需要先加载密钥。

非对称加解密主要用于短信息绑定,对称加解密可以操作任意长度数据。

#### 4.3.3.2.2 数据封装

将数据与特定的平台状态(PCR 值)及可信密码模块绑定在一起,这种操作称为数据封装。

数据加密使一个数据只能绑定于一个可信密码模块。

数据封装使一个数据不仅绑定于一个可信密码模块,同时通过 PCR 绑定于一种平台状态。

使用对称密码算法的封装表示为:

```
sealedData=SM4_Encrypt (key,(data || PCRValue || TCM_Proof))
```

其中 TCM\_Proof 为 TCM 唯一性标识。

数据的封装表示为:

a) 可信密码模块解密数据:

```
data || PCRValue || TCM_Proof=SM4_Decrypt (key,sealedData);
```

b) 可信密码模块比对当前 PCR 的值是否与解密出的 PCRValue 相同;

c) 可信密码模块比对解密出的 TCM\_Proof 是否与内部存储的数值相同;

d) 若比对相同,输出 data。

使用非对称密码算法的封装表示为:

```
sealedData=SM2_Encrypt (SM2PUBKEY,(data || PCRValue || TCM_Proof))
```

数据的解封表示为:

a) 可信密码模块解密数据:

```
data || PCRValue || TCM_Proof=SM2_Decrypt (SM2PRIKEY,sealedData);
```

b) 可信密码模块比对当前 PCR 的值是否与解密出的 PCRValue 相同;

c) 可信密码模块比对解密出的 TCM\_Proof 是否与内部存储的数值相同;

d) 若比对相同,输出 data。

#### 4.3.3.2.3 数字信封

采用对称密码算法对数据进行加密保护,同时采用非对称密码算法对对称密码算法使用的密钥进行加密保护,这种数据安全保护操作称为数字信封。具体方法如下:

a) 随机产生对称加密密钥 symmetricKey;

b) 使用对称加密密钥,采用对称加密算法对数据进行加密:

```
encData=SM4_Encrypt (symmetricKey,data);
```

c) 使用 SM2 密钥的公钥,采用 SM2 加密算法加密对称密钥:

```
encSymmetricKey=SM2_Encrypt (PUBKEY,symmetricKey);
```

d) 在可信密码模块内部,使用 SM2 密钥的私钥,采用 SM2 解密算法,解密对称密钥,使用对称密钥在平台内解密数据。

### 5 可信计算密码支撑平台功能接口

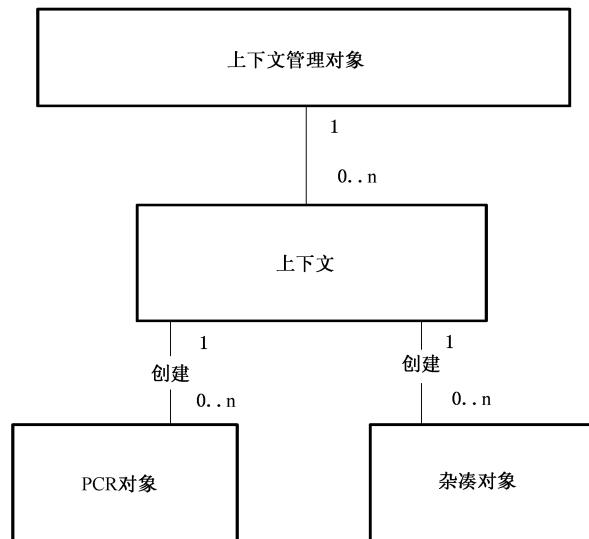
#### 5.1 概述

本章提供了可信计算密码支撑平台功能接口,包括上下文管理、策略管理、可信密码模块管理、密码管理、数据加密与解密、PCR 管理、非易失性存储管理、杂凑操作、密钥协商等九大类接口的定义。

这些接口采用面向对象思想来设计,应用软件在调用时需要创建各个类的工作对象。

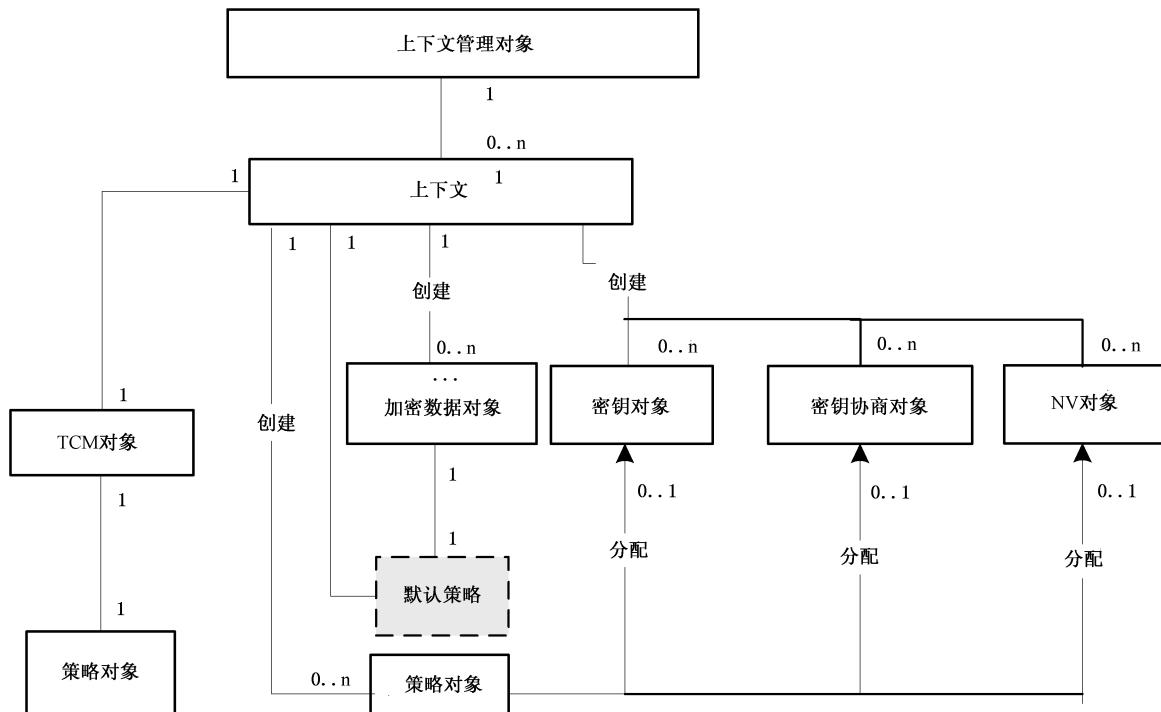
工作对象分为需要授权与无需授权两种,需要授权的工作对象包括 TCM 管理工作对象、密钥对象、加密数据对象、NV 对象、密钥协商对象和策略对象。无需授权的工作对象包括:杂凑对象和 PCR 对象。

对象之间的关系如图 5 和图 6 所示：



- 一个上下文管理对象管理多个上下文对象
- 在一个上下文对象中可创建多个 PCR 对象
- 在一个上下文对象中可创建多个杂凑对象

图 5 无需授权的工作对象之间关系



- 一个策略可分配给多个对象,比如密钥对象,加密数据对象
- 创建上下文时会生成一个默认策略,每个新生成的对象被自动赋予这个默认策略
- 每个工作对象与默认策略绑定,直到有新的策略对象赋予此工作对象
- TCM 对象有一个单独的策略对象代表 TCM 所有者
- 内部的策略函数给一个或多个工作对象分配一个策略对象

图 6 需授权的工作对象之间关系

有关接口函数涉及的数据结构见附录 A。

## 5.2 上下文管理

### 5.2.1 概述

上下文管理类包含关于 TSM 对象的执行环境的信息,如对象的标识和与其他 TSM 软件模块的交互/通信,以及内存管理。

### 5.2.2 创建上下文

功能描述:

创建一个上下文对象,返回该对象句柄。

接口定义:

```

TSM_RESULT Tspi_Context_Create
(
    TSM_HCONTEXT* phContext //out
);
  
```

输入参数描述：

——无。

输出参数描述：

——phContext 创建的上下文对象句柄。

返回参数：

TSM\_SUCCESS

TSM\_E\_INTERNAL\_ERROR

### 5.2.3 关闭上下文

功能描述：

销毁一个上下文对象，并释放所分配的所有资源。

接口定义：

```
TSM_RESULT Tspi_Context_Close
(
    TSM_HCONTEXT hContext //in
);
```

输入参数描述：

——hContext 要关闭的上下文对象的句柄。

输出参数描述：

——无。

返回参数：

TSM\_SUCCESS

TSM\_E\_INVALID\_HANDLE

TSM\_E\_INTERNAL\_ERROR

### 5.2.4 设置上下文属性(整型参数)

功能描述：

本函数设置上下文对象的 32 比特属性。如果请求的数据长度小于 UINT32，必须将数据转换成 32 比特长的数据。

接口定义：

```
TSM_RESULT Tspi_SetAttribUint32
(
    TSM_HOBJECT hObject, //in
    TSM_FLAG attribFlag, //in
    TSM_FLAG subFlag, //in
    UINT32 ulAttrib //in
);
```

输入参数描述：

——hObject 需要设置属性的对象句柄。

——attribFlag 需要设置的属性标记。

——subFlag 需要设置的子属性标记。

——ulAttrib 属性设置的值。

属性定义：

属性	子属性	属性值	描述
TSM_TSPATTRIB_CONTEXT_SILENT_MODE		TSM_TSPATTRIB_CONTEXT_NOT_SILENT	显示协议接口的对话窗口
		TSM_TSPATTRIB_CONTEXT_SILENT	不显示协议接口对话窗口(默认)
TSM_TSPATTRIB_SECRET_HASH_B_MODE	TSM_TSPATTRIB_SECRET_HASH_MODE_POPUP	TSM_TSPATTRIB_HASH_MODE_NOT_NULL	对于不包括其他数据授权密码进行杂凑计算
		TSM_TSPATTRIB_HASH_MODE_NULL	对于包括其他数据授权密码进行杂凑计算

输出参数描述：

——无。

返回参数：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
TSM_E_INVALID_ATTRIB_SUBFLAG
TSM_E_INVALID_ATTRIB_DATA
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR

```

### 5.2.5 获取上下文属性(整型参数)

功能描述：

本函数获取上下文对象的 32 比特属性。

接口定义：

```

TSM_RESULT Tspi_GetAttribUInt32
(
    TSM_HOBJECT hObject, //in
    TSM_FLAG attribFlag, //in
    TSM_FLAG subFlag, //in
    UINT32* pulAttrib //out
);

```

输入参数描述：

- hObject 需要查询属性的对象句柄。
- attribFlag 需要查询的属性标记。
- subFlag 需要查询的子属性标记。

属性定义：

属性	子属性	属性值	描述
TSM_TSPATTRIB_CONTEXT_SILENT_MODE		TSM_TSPATTRIB_CONTEXT_NOT_SI LENT	显示协议接口的对话窗口
		TSM_TSPATTRIB_CONTEXT_SILENT	不显示协议接口对话窗口(默认)
TSM_TSPATTRIB_SECRET_HASH_B_MODE	TSM_TSPATTRIB_SECRET_HASH_MODE_POPUP	TSM_TSPATTRIB_HASH_MODE_NOT_NULL	对于不包括其他数据授权密码进行杂凑计算
		TSM_TSPATTRIB_HASH_MODE_NULL	对于包括其他数据授权密码进行杂凑计算

输出参数描述：

——pulAttrib 指向查询到的属性值。

返回参数：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
TSM_E_INVALID_ATTRIB_SUBFLAG
TSM_E_INVALID_ATTRIB_DATA
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR

```

### 5.2.6 设置上下文属性(变长参数)

功能描述：

本函数设置上下文对象的非 32 比特属性。属性数据的结构和大小依属性而定。

接口定义：

```

TSM_RESULT Tspi_SetAttribData
(
    TSM_HOBJECT hObject,      //in
    TSM_FLAG attribFlag,     //in
    TSM_FLAG subFlag,        //in
    UINT32 ulAttribDataSize, //in
    BYTE* rgbAttribData     //in
);

```

输入参数描述：

——hObject	需要设置属性的对象句柄。
——attribFlag	需要设置的属性标记。
——subFlag	需要设置的子属性标记。
——ulAttribDataSize	prgbAttribData 参数的大小(以字节为单位)
——rgbAttribData	此参数指向一个存放设置属性值的缓冲区。

输出参数描述：

——无。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
TSM_E_INVALID_ATTRIB_SUBFLAG
TSM_E_INVALID_ATTRIB_DATA
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

### 5.2.7 获取上下文属性(变长参数)

功能描述：

本函数获取上下文对象的非 32 比特属性。属性数据的结构和大小依属性而定。为属性数据所分配的内存块必须用 Tspi\_Context\_FreeMemory 来释放。

接口定义：

```
TSM_RESULT Tspi_GetAttribData
(
    TSM_HOBJECT hObject, //in
    TSM_FLAG attribFlag, //in
    TSM_FLAG subFlag, //in
    UINT32 * pulAttribDataSize, //out
    BYTE ** prgbAttribData //out
);
```

输入参数描述：

——hObject 需要获取属性的对象句柄。  
 ——attribFlag 需要获取的属性标记。  
 ——subFlag 需要获取的子属性标记。

输出参数描述：

——pulAttribDataSize 取得的 rgbAttribData 参数的大小(以字节为单位)。若参数 rgbAttribData 为一个 TSM\_UNICODE 字符串，则该大小还包括结束符 NULL。  
 ——prgbAttribData 此参数指向一个存放获取的属性值的缓冲区。

属性定义：

属性	子属性	属性值
TSM_TSPATTRIB_CONTEXT _MACHINE_NAME	0	提供 TSM 运行的机器名，是一个以 NULL 结束的 TSM_UNICODE 字符串。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
TSM_E_INVALID_ATTRIB_SUBFLAG
TSM_E_INVALID_ATTRIB_DATA
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

### 5.2.8 连接上下文

功能描述：

建立与 TCM 的连接, TCM 可以为本地的或远程的。

接口定义：

```
TSM_RESULT Tspi_Context_Connect
(
    TSM_HCONTEXT hContext,           //in
    TSM_UNICODE* wszDestination     //in
);
```

输入参数描述：

——hContext 上下文对象句柄。

——wszDestination 标识连接到的远程服务(以 null 结束),NULL 代表连接本地服务。

输出参数描述：

——无。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_NO_CONNECTION
TSM_E_INTERNAL_ERROR
```

### 5.2.9 释放上下文

功能描述：

释放由 TSM 所分配的上下文内存资源。

接口定义：

```
TSM_RESULT Tspi_Context_FreeMemory
(
    TSM_HCONTEXT hContext,   //in
    BYTE* rgbMemory         //in
);
```

输入参数描述：

——hContext 上下文对象句柄。

——rgbMemory 上下文对象内存指针。若为 NULL, 则与该上下文对象绑定的内存块已经释放。

输出参数描述：

——无。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INTERNAL_ERROR
TSM_E_INVALID_RESOURCE
```

### 5.2.10 获取上下文默认策略

功能描述：

获取上下文的默认策略对象。

接口定义：

```
TSM_RESULT Tspi_Context_GetDefaultPolicy
(
    TSM_HCONTEXT hContext, //in
    TSM_HPOLICY* phPolicy //out
);
```

输入参数描述：

——hContext 上下文对象句柄。

输出参数描述：

——phPolicy 返回上下文对象的策略对象句柄。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INTERNAL_ERROR
```

### 5.2.11 创建对象

功能描述：

创建并初始化一个指定类型的空对象，返回该对象的句柄。

该空对象将与一个已经打开的上下文对象相关联。

接口定义：

```
TSM_RESULT Tspi_Context_CreateObject
(
    TSM_HCONTEXT hContext, //in
    TSM_FLAG objectType, //in
    TSM_FLAG initFlags, //in
    TSM_HOBJECT* phObject //out
);
```

输入参数描述：

——hContext 上下文对象句柄。

——objectType 要创建的对象类型标识。

——initFlags 创建的对象属性的缺省值。

输出参数描述：

——phObject 返回所创建的对象。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_OBJECT_TYPE
```

```

TSM_E_INVALID_OBJECT_INIT_FLAG
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
TSM_E_ENC_INVALID_TYPE
TSM_E_HASH_INVALID_ALG

```

### 5.2.12 关闭对象

功能描述：

销毁一个上下文对象句柄所关联的对象，释放该对象相关的资源。

接口定义：

```

TSM_RESULT Tspi_Context_CloseObject
(
    TSM_HCONTEXT hContext, //in
    TSM_HOBJECT hObject //in
);

```

输入参数描述：

——hContext 上下文对象句柄。

——phObject 要释放的对象句柄。

输出参数描述：

——无。

返回参数：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INTERNAL_ERROR

```

### 5.2.13 获取平台功能特性

功能描述：

获取 TSM 应用服务或核心服务的功能/性能/属性数据。

接口定义：

```

TSM_RESULT Tspi_Context_GetCapability
(
    TSM_HCONTEXT          hContext,           //in
    TSM_FLAG               capArea,            //in
    UINT32                 ulSubCapLength,      //in
    BYTE*                  rgbSubCap,          //in
    UINT32*                pulRespDataLength,   //out
    BYTE**                 prgbRespData       //out
);

```

输入参数描述：

——hContext 上下文对象句柄。

——capArea 查询的属性名称，参见属性说明表。

——ulSubCapLength 子属性的长度参数。

——rgbSubCap 查询的子属性参数。

属性说明表：

属性标记	子属性标记	数据描述
TSM_TCSCAP_ALG	TSM_ALG_XX: 代表支持的算法名称	BOOL 返回: TRUE 代表系统服务支持该算法, FALSE 代表不支持
TSM_TCSCAP_VERSION		从系统服务获取 TSM_VERSION 结构说明数据
TSM_TCSCAP_CACHING	TSM_TCSCAP_PROP_KEYCACHE	BOOL 返回: TRUE 说明系统服务支持密钥缓存; FALSE 说明不支持
TSM_TCSCAP_CACHING	TSM_TCSCAP_PROP_AUTHACHE	BOOL 返回: TRUE 说明系统服务支持授权协议缓存; FALSE 说明不支持
TSM_TCSCAP_PERSSTORAGE		BOOL 返回: TRUE 说明系统服务支持永久存储; FALSE 说明不支持
TSM_TSPCAP_ALG	TSM_ALG_DEFAULT	返回默认算法
TSM_TSPCAP_ALG	TSM_ALG_DEFAULT_SIZE	返回默认密钥长度
TSM_TSPCAP_ALG	TSM_ALG_XX: 代表支持的算法名称	BOOL 返回: TRUE 代表支持该算法, , FALSE 代表不支持
TSM_TSPCAP_VERSION		获取 TSM 版本
TSM_TSPCAP_PERSSTORAGE		BOOL 返回: TRUE 说明支持永久存储; FALSE 说明不支持
TSM_TCSCAP_MANUFACTURER	TSM_TCSCAP_PROP_MANUFACTURER_ID	UINT32 返回: 说明系统服务厂商
	TSM_TCSCAP_PROP_MANUFACTURER_STR	返回系统服务厂商名称
TSM_TSPCAP_MANUFACTURER	TSM_TSPCAP_PROP_MANUFACTURER_ID	UINT32 返回: 说明 TSM 厂商
	TSM_TSPCAP_PROP_MANUFACTURER_STR	返回 TSM 厂商名称
TSM_TSPCAP_RETURNVALUE_INFO	TSM_TSPCAP_PROP_RETURNVALUE_INFO	0 说明采用 ASN.1 编码, 1 说明采用字节流

输出参数描述：

- pulRespDataLength 返回查询的属性参数长度。  
 ——prgbRespData 返回查询的属性数据内存地址。

返回参数：

TSM\_SUCCESS  
 TSM\_E\_INVALID\_HANDLE  
 TSM\_E\_BAD\_PARAMETER  
 TSM\_E\_INTERNAL\_ERROR

#### 5.2.14 获取 TCM 对象句柄

功能描述：

获取 TCM 管理对象的上下文,一个给定的上下文只对应一个 TCM 管理对象实例,代表 TCM 所有者。

接口定义：

```
TSM_RESULT Tspi_Context_GetTcmObject
(
    TSM_HCONTEXT      hContext,          //in
    TSM_HTCM*         phTCM,            //out
);
```

输入参数描述：

——hContext 上下文对象句柄。

输出参数描述：

——phTCM 返回获取的 TCM 类对象句柄。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

### 5.2.15 通过密钥属性加载密钥

功能描述：

根据 key blob 中包含的信息创建密钥对象，并加载到 TCM 中，用 hUnwrappingKey 指向的密钥解密 key blob。hUnwrappingKey 指向的密钥必须先被加载到 TCM 内。

接口定义：

```
TSM_RESULT Tspi_Context_LoadKeyByBlob
(
    TSM_HCONTEXT hContext,          //in
    TSM_HKEY hUnwrappingKey,       //in
    UINT32 ulBlobLength,           //in
    BYTE* rgbBlobData,             //in
    TSM_HKEY* phKey,              //out
);
```

输入参数描述：

——hContext 上下文对象句柄。

——hUnwrappingKey 密钥对象句柄，此密钥用于解密包 rgbBlobData 中的密钥信息。

——ulBlobLength rgbBlobData 指向的密钥数据块的长度（以字节为单位）。

——rgbBlobData 待加载的密钥数据块。

输出参数描述：

——phKey 代表被加载密钥的 Key 对象句柄。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

### 5.2.16 通过密钥 ID 加载密钥

功能描述：

密钥管理器根据 UUID 获取密钥信息，填充密钥对象，并将密钥加载到 TCM 中。加载密钥时所需的保护操作密钥信息可从永久性存储器获得。

使用此命令时，需要考虑保护操作密钥的授权要求：

- (1) 如果该密钥的所有保护操作密钥都不需要授权，应用程序可以直接调用 Tspi\_Context\_LoadKeyByUUID() 来加载当前密钥。
- (2) 如果有一个密钥需要授权，并且应用程序知道密钥缓存信息，则应用程序必须通过调用函数 Tspi\_Context\_GetKeyByUUID() 从密钥库中获得密钥信息，并为其指派策略后，用 Tspi\_Key\_LoadKey() 加载密钥。
- (3) 如果有一个密钥需要授权，并且应用程序不知道密钥缓存信息，则应用程序必须通过调用 Tspi\_Context\_GetRegisteredKeysByUUID() 从密钥库中获取密钥缓存信息。然后为其指派策略，用 Tspi\_Key\_LoadKey() 加载密钥。

接口定义：

```
TSM_RESULT Tspi_Context_LoadKeyByUUID
(
    TSM_HCONTEXT hContext, //in
    TSM_FLAG persistentStorageType, //in
    TSM_UUID uuidData, //in
    TSM_HKEY* phKey //out
);
```

输入参数描述：

——hContext	上下文对象句柄。
——persistentStorageType	标志，指明要加载的密钥注册在系统库还是用户库中。
——uuidData	密钥的 UUID 值。

输出参数描述：

——phKey 返回密钥对象句柄。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_PS_KEY_NOTFOUND
TSM_E_INTERNAL_ERROR
```

### 5.2.17 注册密钥

功能描述：

将密钥注册到 TSM 永久性存储数据库中。

接口定义：

```
TSM_RESULT Tspi_Context_RegisterKey
(
    TSM_HCONTEXT hContext,           //in
    TSM_HKEY hKey,                  //in
```

```

TSM_FLAG persistentStorageType,           //in
TSM_UUID uuidKey,                      //in
TSM_FLAG persistentStorageTypeParent,    //in
TSM_UUID uuidParentKey                 //in
);

```

输入参数描述：

——hContext	上下文对象句柄。
——hKey	待注册的密钥对象句柄。在调用本方法之前应调用 Tspi_SetAttribData()方法填充密钥对象。
——persistentStorageType	永久存储类型,用来指明要把密钥注册到系统库还是用户库中。
——uuidKey	分配给密钥的 UUID。
——persistentStorageTypeParent	用来指明保护操作密钥注册在系统库还是用户库中的标志。

输出参数描述：

——uuidParentKey 保护操作密钥的 UUID

返回参数：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_PS_KEY_NOTFOUND
TSM_E_INTERNAL_ERROR

```

### 5.2.18 销毁密钥

功能描述：

将密钥从密钥数据库中注销。

接口定义：

```

TSM_RESULT Tspi_Context_UnregisterKey
(
    TSM_HCONTEXT hContext,           //in
    TSM_FLAG persistentStorageType, //in
    TSM_UUID uuidKey,              //in
    TSM_HKEY* phkey,                //out
);

```

输入参数描述：

——hContext	上下文对象句柄。
——persistentStorageType	指明密钥注册位置(系统库或用户库)的标志。
——uuidKey	密钥的 UUID。

输出参数描述：

——phKey 返回密钥对象句柄。

返回参数：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_PS_KEY_NOTFOUND
TSM_E_INTERNAL_ERROR

```

### 5.2.19 通过密钥 ID 获取密钥

功能描述：

利用 UUID 在密钥库中查找注册的密钥信息，创建并初始化密钥对象，返回新创建的密钥对象的句柄。

接口定义：

```
TSM_RESULT Tspi_Context_GetKeyByUUID
(
    TSM_HCONTEXT hContext, //in
    TSM_FLAG persistentStorageType, //in
    TSM_UUID uuidData, //in
    TSM_HKEY* phKey //out
);
```

输入参数描述：

——hContext	上下文对象句柄。
——persistentStorageType	指明密钥注册位置(系统库或用户库)的标志。
——uuidData	密钥的 UUID。

输出参数描述：

——phKey	函数执行成功返回密钥对象句柄。
---------	-----------------

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_PS_KEY_NOTFOUND
TSM_E_INTERNAL_ERROR
```

### 5.2.20 通过公钥获取密钥

功能描述：

利用给定的公钥信息在密钥库中查找到所对应的密钥信息，创建并初始化密钥对象。返回新创建的密钥对象的句柄。

接口定义：

```
TSM_RESULT Tspi_Context_GetKeyByPublicInfo
(
    TSM_HCONTEXT hContext, //in
    TSM_FLAG persistentStorageType, //in
    TSM_ALGORITHM_ID algID, //in
    UINT32 ulPublicInfoLength, //in
    BYTE* rgbPublicInfo, //in
    TSM_HKEY* phKey //out
);
```

输入参数描述：

——hContext	上下文对象句柄。
——persistentStorageType	指明密钥注册位置的标志。

——ulAlgId 所查找的密钥的算法标识。  
 ——ulPublicInfoLength rgbPublicInfo 参数的长度(以字节为单位)。  
 ——rgbPublicInfo 公钥数据块。

输出参数描述：

——phKey 函数执行成功返回密钥对象的接口指针。

返回参数：

```
TSM_SUCCESS  
TSM_E_INVALID_HANDLE  
TSM_E_BAD_PARAMETER  
TSM_E_PS_KEY_NOTFOUND  
TSM_E_INTERNAL_ERROR
```

### 5.2.21 通过 ID 获取注册密钥

功能描述：

获取一个 TSM\_KM\_KEYINFO 结构的数组,该数组反映注册密钥的层次信息。

注：调用方需要调用 Tspi\_Context\_FreeMemory 方法来释放本函数分配的内存。

接口定义：

```
TSM_RESULT Tspi_Context_GetRegisteredKeysByUUID  
(  
    TSM_HCONTEXT hContext,           //in  
    TSM_FLAG persistentStorageType, //in  
    TSM_UUID* pUuidData,           //in  
    UINT32* pulKeyHierarchySize,    //out  
    TSM_KM_KEYINFO** ppKeyHierarchy //out  
) ;
```

输入参数描述：

——hContext 上下文对象句柄。  
 ——persistentStorageType 指明密钥注册位置(系统库或用户库)的标志。  
 ——pUuidData 密钥注册后分配的 UUID。如果此参数为 NULL,返回的数据包括从根密钥开始的所有密钥的层次结构。如果此参数指定了 UUID,返回的数据包括从指定密钥开始到 SMK 的路径的层次结构,返回数据的第一个元素是指定密钥的信息,依次是其保护操作密钥的信息,最后一个为 SMK 的信息。

输出参数描述：

——pulKeyHierarchySize ppKeyHierarchy 参数的大小(以字节为单位)。  
 ——ppKeyHierarchy 成功执行后,ppKeyHierarchy 指向一个存放密钥层次结构数据的缓冲区。

返回参数：

```
TSM_SUCCESS  
TSM_E_INVALID_HANDLE  
TSM_E_BAD_PARAMETER  
TSM_E_INTERNAL_ERROR
```

### 5.2.22 设置传输会话加密密钥

功能描述：

为上下文对象设置具体的传输会话加密密钥。

接口定义：

```
TSM_RESULT Tspi_Context_SetTransEncryptionKey
(
    TSM_HCONTEXT      hContext,          //in
    TSM_HKEY          hKey,             //in
);
```

输入参数描述：

——hContext 上下文对象句柄

——hKey 指定进行会话加密使用的密钥柄

输出参数描述：

——无

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

### 5.2.23 关闭传输会话

功能描述：

该命令结束传输会话，释放传输会话加密密钥。

接口定义：

```
TSM_RESULT Tspi_Context_CloseTransport
(
    TSM_HCONTEXT      hContext,          //in
);
```

输入参数描述：

——hContext 上下文对象句柄。

输出参数描述：

——无

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

## 5.3 策略管理

策略管理类用于为不同用户应用程序配置相应的安全策略与行为。

应用程序通过策略管理为授权机制提供专门的授权秘密信息的处理操作。

### 5.3.1 设置策略类属性(整型参数)

功能描述：

本函数设置对象的 32 比特属性。如果请求的数据长度小于 UINT32，必须将数据转换成正确的大小。

接口定义：

```
TSM_RESULT Tspi_SetAttribUint32
(
    TSM_HOBJECT    hObject,      //in
    TSM_FLAG       attribFlag,   //in
    TSM_FLAG       subFlag,     //in
    UINT32         ulAttrib     //in
);
```

输入参数描述：

——hObject 需要设置属性的对象句柄。

——attribFlag 需要设置的属性标记。

——subFlag 需要设置的子属性标记。

属性说明表：

属性标识	子属性标识	数据描述
TSM_TSPATTRIB_P_OICY_CALLBACK_HMAC	应用程序提供	
TSM_TSPATTRIB_P_OICY_CALLBACK_TAKEOWNERSHIP	应用程序提供	
TSM_TSPATTRIB_P_OICY_SECRET_LIFETIME	TSM_TSPATTRIB_P_OLSECRET_LIFETIME_ALWAYS	授权数据将一直有效
	TSM_TSPATTRIB_P_OLSECRET_LIFETIME_COUNTER	授权数据能够被使用的次数
	TSM_TSPATTRIB_P_OLSECRET_LIFETIME_TIMER	授权数据能够被使用秒数
TSM_TSPATTRIB_S_ECRET_HASH_MODE	TSM_TSPATTRIB_SECRET_HASH_MODE_POPUP	TSM_TSPATTRIB_HASH_MODE_NULL(默认)
		TSM_TSPATTRIB_HASH_MODE_NOT_NULL

——ulAttrib 属性设置的值。

输出参数描述：

——无

返回参数

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
TSM_E_INVALID_ATTRIB_SUBFLAG
```

```

TSM_E_INVALID_ATTRIB_DATA
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR

```

### 5.3.2 获取上下文属性(整型参数)

功能描述：

本函数获取对象的 32 比特属性。如果请求的数据长度小于 UINT32，必须将数据转换成正确的大小。

接口定义：

```

TSM_RESULT Tspi_GetAttribUint32
(
    TSM_HOBJECT    hObject,      // in
    TSM_FLAG       attribFlag,   // in
    TSM_FLAG       subFlag,     // in
    UINT32*        pulAttrib   // out
);

```

输入参数描述：

——hObject 需要获取属性的对象句柄。

——attribFlag 需要获取的属性标记。

——subFlag 需要获取的子属性标记。

属性说明表：

属性标识	子属性标识	数据描述
TSM_TSPATTRIB_POLICY_CALLBACK_HMAC	应用程序提供	
TSM_TSPATTRIB_POLICY_CALLBACK_TAKEOWNERSHIP	应用程序提供	
TSM_TSPATTRIB_POLICY_SECRET_LIFETIME	TSM_TSPATTRIB_POLSECRET_LIFETIME_ALWAYS	授权数据将一直有效
	TSM_TSPATTRIB_POLSECRET_LIFETIME_ME_COUNTER	授权数据能够被使用的次数
	TSM_TSPATTRIB_POLSECRET_LIFETIME_TIMER	授权数据能够被使用秒数
TSM_TSPATTRIB_SECRET_HASH_MODE	TSM_TSPATTRIB_SECRET_HASH_MODE_POPUP	TSM_TSPATTRIB_HASH_MODE_NULL(默认)
		TSM_TSPATTRIB_HASH_MODE_NOT_NULL

输出参数描述：

——pulAttrib 接收属性设置的值。

返回参数

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
TSM_E_INVALID_ATTRIB_SUBFLAG
TSM_E_INVALID_ATTRIB_DATA
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

### 5.3.3 设置上下文属性(变长参数)

功能描述：

本函数设置对象的非 32 比特属性。属性数据的结构和大小依属性而定。

接口定义：

```
TSM_RESULT Tspi_SetAttribData
(
    TSM_HOBJECT    hObject,           //in
    TSM_FLAG       attribFlag,        //in
    TSM_FLAG       subFlag,          //in
    UINT32         ulAttribDataSize, //in
    BYTE*          rgbAttribData   //in
);
```

输入参数描述：

- hObject 需要设置属性的对象句柄。
- attribFlag 指示要设置的属性的标志。
- subFlags 指示要设置的属性的子标志。

属性说明表：

属性标识	子属性标识	数据描述
TSM_TSPATTRIB_POLICY_CALLBACK_HMAC	应用程序提供	
TSM_TSPATTRIB_POLICY_CALLBACK_TAKEOWNERSHIP	应用程序提供	
TSM_TSPATTRIB_POLICY_POPUPSTRING	0	POPUP 窗口的字符串

——ulAttribDataSize rgbAttribData 参数的大小(以字节为单位)。

——rgbAttribData 此参数指向一个存放设置属性值的缓冲区。

输出参数描述：

——无。

返回参数

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
```

```

TSM_E_INVALID_ATTRIB_SUBFLAG
TSM_E_INVALID_ATTRIB_DATA
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR

```

#### 5.3.4 获取上下文属性(变长参数)

功能描述：

本函数获取对象的非 32 比特属性。属性数据的结构和大小依属性而定。对于开辟的内存空间，需要在 Tspi\_Context\_FreeMemory 中释放。

接口定义：

```

TSM_RESULT Tspi_GetAttribData
(
    TSM_HOBJECT hObject,           //in
    TSM_FLAG attribFlag,          //in
    TSM_FLAG subFlag,             //in
    UINT32* pulAttribDataSize,    //out
    BYTE** prgbAttribData        //out
);

```

输入参数描述：

——hObject 需要获取属性的对象句柄。

——attribFlag 需要获取的属性的标志。

——subFlags 需要获取的属性的子标志。

属性说明表：

属性标识	子属性标识	数据描述
TSM_TSPATTRIB_POLICY_CALLBACK_K_HMAC	应用程序提供	
TSM_TSPATTRIB_POLICY_CALLBACK_K_TAKEOWNERSHIP	应用程序提供	
TSM_TSPATTRIB_POLICY_POPUPSTRING	0	POPUP 窗口的字符串

输出参数描述：

——pulAttribDataSize prgbAttribData 参数的大小(以字节为单位)。

——prgbAttribData 此参数指向一个存放指定属性值的缓冲区。

返回参数

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
TSM_E_INVALID_ATTRIB_SUBFLAG
TSM_E_INVALID_ATTRIB_DATA
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR

```

### 5.3.5 设置策略授权

功能描述：

本函数设置策略对象的授权数据和句柄。

如果 secret mode 不要求任何授权数据,参数 ulSecretLength 被设为 0,rgbSecret 为 NULL。

secret mode 包括：

- a) TSM\_SECRET\_MODE\_NONE:无授权要求。
- b) TSM\_SECRET\_MODE\_SCH:对 rgbSecret 指向一个散列后的 32 字节秘密信息。
- c) TSM\_SECRET\_MODE\_PLAIN:rgbSecret 指向一段明文,ulSecretLength 为 rgbSecret 的串长。
- d) TSM\_SECRET\_MODE\_POPUP:TSM 提示用户输入一串口令,其代表了一串 TSM\_UNICODE 字符串.该字符串必须为杂凑计算值。

接口定义：

```
TSM_RESULT Tspi_Policy_SetSecret
(
    TSM_HPOLICY hPolicy,           //in
    TSM_FLAG secretMode,          //in
    UINT32 ulSecretLength,         //in
    BYTE* rgbSecret               //in
);
```

输入参数描述：

——hPolicy	策略对象句柄。
——secretMode	策略所采用的安全模式。
——ulSecretLength	参数的字节长度。
——rgbSecret	与策略相关的秘密数据 blob。

输出参数描述：

——无。

返回参数

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

### 5.3.6 清除策略授权

功能描述：

用来清除缓存的策略授权信息。

接口定义：

```
TSM_RESULT Tspi_Policy_FlushSecret
(
    TSM_HPOLICY hPolicy           //in
);
```

输入参数描述：

——hPolicy 策略对象句柄。

输出参数描述:

——无

返回参数:

TSM\_SUCCESS  
TSM\_E\_INVALID\_HANDLE  
TSM\_E\_INTERNAL\_ERROR

### 5.3.7 绑定策略对象

功能描述:

该函数分配策略给一个工作对象(TCM, key, encdata),每一个工作对象利用这个策略去发起一个授权的 TCM 命令。每一个工作对象在创建时都有一个默认的策略,可以由该函数给工作对象绑定一个新策略,同时要把该策略加到该工作对象的策略列表里。

接口定义:

```
TSM_RESULT Tspi_Policy_AssignToObject  
(  
    TSM_HPOLICY    hPolicy,           //in  
    TSM_HOBJECT    hObject           //in  
)
```

输入参数描述:

——hPolicy 策略对象句柄。

——hObject 工作对象句柄。

输出参数描述:

——无

返回参数:

TSM\_SUCCESS  
TSM\_E\_INVALID\_HANDLE  
TSM\_E\_INTERNAL\_ERROR

## 5.4 可信密码模块(TCM)管理

### 5.4.1 概述

TCM 管理类用于表示 TCM 的所有者。TCM 的所有者可以看成是 PC 环境中的系统管理员。因此,每个上下文仅有一个 TCM 管理类的实例。这个对象自动与一个策略对象相关联,用于处理 TCM 所有者的授权数据。此外,它还提供一些基本控制和报告的功能。

### 5.4.2 创建平台身份和证书请求

功能描述:

本函数创建平台身份密钥(PIK),绑定用户身份标记信息,返回一个证书请求包。该证书供可信方来验证这个身份密钥。

只有 TCM Owner 有权创建平台身份密钥。

接口定义:

```
TSM_RESULT Tspi_TCM_CollateIdentityRequest  
(
```

```

TSM_HTCM hTCM, //in
TSM_HKEY hKeySMK, //in
TSM_HKEY hCAPubKey, //in
UINT32 ulIdentityLabelLength, //in
BYTE* rgbIdentityLabelData, //in
TSM_HKEY hIdentityKey, //in
TSM_ALGORITHM_ID algID, //in
UINT32* pulTCMIdentityReqLength, //out
BYTE** prgbTCMIdentityReq //out
);

```

输入参数描述：

——hTCM	TCM 对象句柄。
——hKeySMK	SMK 对象句柄。
——hCAPubKey	含有可信方公钥信息的密钥对象句柄。
——ulIdentityLabelLength	rgbIdentityLabelData 参数的字节数。
——rgbIdentityLabelData	指向由用户设定的代表平台身份的字符串,为 TSM_UNICODE 字符串。
——hIdentityKey	身份密钥对象句柄。
——algID	对称密钥算法类型,用于标识加密 PIK 证书请求信息的对称密钥算法。

输出参数描述：

——pulTCMIdentityReqLength	接收 prgbTCMIdentityReq 的缓冲区字节大小。
——prgbTCMIdentityReq	指向一个发送给可信方的请求数据。

返回参数

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR

```

#### 5.4.3 激活平台身份和获取 PIK 证书

功能描述：

本函数验证 PIK 证书的真实性,并返回解密的证书。

接口定义：

```

TSM_RESULT Tspi_TCM_ActivateIdentity
(
    TSM_HTCM hTCM, //in
    TSM_HKEY hIdentKey, //in
    UINT32 ulAsymCAContentsBlobLength, //in
    BYTE* rgbAsymCAContentsBlob, //in
    UINT32 ulSymCAAttestationBlobLength, //in
    BYTE* rgbSymCAAttestationBlob, //in
    UINT32* pulCredentialLength, //out
    BYTE** prgbCredential //out
);

```

输入参数描述:

——hTCM	指向 TCM 对象的句柄。
——hIdentityKey	身份密钥对象句柄。
——ulAsymCAContentsBlobLength	rgbAsymCAContentsBlob 参数的大小(单位:字节)。
——rgbAsymCAContentsBlob	指向从可信方获得的加密数据,包含用于解密被加密 PIK 证书的对称密钥和 PIK 公钥的杂凑值。
——ulSymCAAttestationBlobLength	rgbSymCAAttestationBlob 参数的大小(单位:字节)。
——rgbSymCAAttestationBlob	指向从可信方获得的被加密的 PIK 证书。

输出参数描述:

——pulCredentialLength	prgbCredential 参数的大小(单位:字节)。
——prgbCredential	指向被解密的 PIK 证书数据。

返回参数

TSM_SUCCESS	
TSM_E_INVALID_HANDLE	
TSM_E_BAD_PARAMETER	
TSM_E_INTERNAL_ERROR	

#### 5.4.4 创建 PEK 请求

功能描述:

创建 PEK 证书请求。

接口定义:

```
TSM_RESULT Tspi_TCM_CollatePekRequest
(
    TSM_HTCM           hTCM,                //in
    TSM_HKEY            hCAPubKey,           //in
    UINT32              ulPekLabelLength,     //in
    BYTE*               rgbPekLabelData,      //in
    TSM_ALGORITHM_ID   algID,               //in
    UINT32              ulPekParamsLength,    //in
    BYTE*               rgbPekParams,         //in
    UINT32*             pulTCMPekReqLength,   //out
    BYTE**              prgbTCMPekReq        //out
);
```

输入参数描述:

——hTCM	TCM 对象句柄。
——hCAPubKey	可信方的公钥对象句柄。
——ulPekLabelLength	rgbPekLabelData 参数的字节数。
——rgbPekLabelData	指向身份标示的内存区指针,其指向的内容 TSM_UNICODE 类型的字符串。
——algID	对称密钥算法类型,用于标识加密 PEK 及其证书的请求信息的对称密钥算法。
——ulPekParamsLength	rgbPekParams 数据长度(单位:字节)。
——rgbPekParams	PEK 密钥参数。

输出参数描述：

- pulTCMPekReqLength 接收 prgbTCMPekReq 的缓冲区字节大小。
- prgbTCMPekReq 指向用于请求 PEK 及其证书的结构数据。

返回参数

```
TSM_SUCCESS  
TSM_E_INVALID_HANDLE  
TSM_E_BAD_PARAMETER  
TSM_E_INTERNAL_ERROR
```

#### 5.4.5 获取 PEK 证书

功能描述：

本函数验证 PEK 证书的真实性，并返回解密的证书。

接口定义：

```
TSM_RESULT Tspi_TCM_ActivatePEKCert  
(  
    TSM_HTCM           hTCM,                      //in  
    UINT32             ulAsymCAContentsBlobLength, //in  
    BYTE*              rgbAsymCAContentsBlob,      //in  
    UINT32             ulSymCAAttestationBlobLength, //in  
    BYTE*              rgbSymCAAttestationBlob,     //in  
    UINT32*            pulCredentialLength,        //out  
    BYTE**             prgbCredential            //out  
) ;
```

输入参数描述：

- hTCM 指向 TCM 对象的句柄。
- ulAsymCAContentsBlobLength rgbAsymCAContentsBlob 参数的大小(单位:字节)。
- rgbAsymCAContentsBlob 指向被密码模块密钥公钥加密的对称密钥，其从可信方处获得。
- ulSymCAAttestationBlobLength rgbSymCAAttestationBlob 参数的大小(单位:字节)。
- rgbSymCAAttestationBlob 指针，指向被加密的 PEK 证书，其从可信方处获得。

输出参数描述：

- pulCredentialLength prgbCredential 参数的大小(单位:字节)。
- prgbCredential 指向被解密的 PEK 证书数据。

返回参数

```
TSM_SUCCESS  
TSM_E_INVALID_HANDLE  
TSM_E_BAD_PARAMETER  
TSM_E_INTERNAL_ERROR
```

#### 5.4.6 导入 PEK 密钥

功能描述：

导入 PEK 密钥。

接口定义：

```
TSM_RESULT Tspi_TCM_ActivatePEK
(
    TSM_HTCM      hTCM,           //in
    TSM_HKEY      hSMKKey,        //in
    TSM_HKEY      hPEKKey,        //in,out
    TSM_HPCRS     hPEKPcr,        //in
    UINT32         ulAsymCAContentsBlobLength, //in
    BYTE*          rgbAsymCAContentsBlob,   //in
    UINT32         ulSymCAAttestationBlobLength, //in
    BYTE*          rgbSymCAAttestationBlob,   //in
);

```

输入参数描述：

——hTCM	指向 TCM 对象的句柄。
——hSMKKey	SMK 对象句柄。
——hPEKKey	用于承载解密之后 PEK 密钥对象句柄。
——hPEKPcr	用于指定 PEK 平台 PCR 绑定属性,可以为空。
——ulAsymCAContentsBlobLength	rgbAsymCAContentsBlob 参数的大小(单位:字节)。
——rgbAsymCAContentsBlob	指向被 EK 公钥加密的对称密钥,其从可信方处获得。
——ulSymCAAttestationBlobLength	rgbSymCAAttestationBlob 参数的大小(单位:字节)。
——rgbSymCAAttestationBlob	指向被对称密钥加密的 PEK 密钥结构。

输出参数描述：

——hPEKKey 指向解密之后 PEK 密钥对象句柄。

返回参数

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

#### 5.4.7 创建不可撤消的密码模块密钥

功能描述：

创建不可撤消的密码模块密钥。hKey 对象的属性由 Tspi\_SetAttribData() 设置

接口定义：

```
TSM_RESULT Tspi_TCM_CreateEndorsementKey
(
    TSM_HTCM hTCM,           //in
    TSM_HKEY hKey,           //in
    TSM_VALIDATION* pValidationData //in,out
);

```

输入参数描述：

——hTCM	指向 TCM 对象的句柄。
——hKey	指向 EK 的密钥对象句柄,该对象描述了 EK 的创建属性。
——pValidationData	提供用于计算签名的外部数据。

输出参数描述：

——pValidationData 该命令执行成功后,该缓冲区包含计算得到的验证数据。

返回参数

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

#### 5.4.8 获取密码模块密钥公钥

功能描述：

本函数获取密码模块的公钥。

接口定义：

```
TSM_RESULT Tspi_TCM_GetPubEndorsementKey
(
    TSM_HTCM hTCM,                      //in
    TSM_BOOL fOwnerAuthorized,           //in
    TSM_VALIDATION* pValidationData,     //in,out
    TSM_HKEY* phEndorsementPubKey      //out
);
```

输入参数描述：

——hTCM 指向 TCM 对象的句柄。

——fOwnerAuthorized 如果为真,若要获取 EK 的公钥,TCM 所有者的秘密必须提供,否则不需要。

——pValidationData 提供用于计算签名的外部数据。

输出参数描述：

——pValidationData 该命令执行成功后,pValidationData 包含计算得到的验证数据。

——phEndorsementPubKey 获取到指向 EK 公钥的密钥对象句柄。

返回参数

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

#### 5.4.9 创建可撤销的密码模块密钥

功能描述：

创建一个可撤销的密码模块密钥,输入输出参数描述了撤销 EK 的授权是由 TCM 实现还是应用程序实现。撤销 EK 的授权数据应由调用者来保护和管理。

接口定义：

```
TSM_RESULT Tspi_TCM_CreateRevocableEndorsementKey
(
    TSM_HTCM hTCM,                      //in
    TSM_HKEY hKey,                       //in
    TSM_VALIDATION* pValidationData,     //in,out
);
```

```

    UINT32* pulEkResetDataLength,           //in,out
    BYTE** prgbEkResetData                //in,out
);

```

输入参数描述：

——hTCM	TCM 对象句柄。
——hKey	EK 句柄。
——pValidationData	提供用于计算签名的外部数据。
——pulEkResetDataLength	如果这一值为 0, TSM 使用 TCM 来产生撤销 EK 的授权数据,之后,该参数包含撤销 EK 的授权数据包的长度。如果是其他值,则表示了由外部产生的撤销 EK 的授权数据包长度。
——prgbEkResetData	若 pulEkResetDataLength 不为 0, 该数据为外部创建的撤销 EK 的授权数据。

输出参数描述：

——pValidationData	该命令执行成功后,该缓冲区包含计算得到的验证数据。
——pulEkResetDataLength	如果这一值为 0, TSM 使用 TCM 来产生撤销 EK 的授权数据,之后,该参数包含复位数据包的长度。如果是其他值,则表示了由外部产生的复位数据包长度。
——prgbEkResetData	若输入的 * pulEKResetDataLength 为 0, 则此变量的输出表示由 TCM 产生的撤销 EK 授权数据。

返回参数

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR

```

#### 5.4.10 撤销密码模块密钥

功能描述：

该命令清除 TCM 的可撤销 EK, 并执行清除所有者的命令。此命令相当于清除所有的计数器(除了基准计数器)、EK、SMK、所有者的授权和及其相关的 NV。该命令不涉及其他的 NV。

接口定义：

```

TSM_RESULT Tspi_TCM_RevokeEndorsementKey
(
    TSM_HTCM hTCM,                      //in
    UINT32 ulEkResetDataLength,          //in
    BYTE* rgbEkResetData               //in
);

```

输入参数描述：

——hTCM	TCM 对象句柄。
——ulEkResetDataLength	撤销 EK 的授权数据大小。
——rgbEkResetData	撤销 EK 的授权数据。

输出参数描述：

——无

返回参数

```
TSM_SUCCESS
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

#### 5.4.11 创建密码模块所有者

功能描述：

该方法取得 TCM 的所有权。这一过程是 TCM 所有者插入一段共享秘密到 TCM 中, TCM 的所有者有权限执行一些特殊操作。当该命令执行时, 隐含地执行了调用 Tspi\_Context\_RegisterKey 对 SMK 进行注册的过程。

接口定义：

```
TSM_RESULT Tspi_TCM_TakeOwnership
(
    TSM_HTCM hTCM,           //in
    TSM_HKEY hKeySMK,        //in
    TSM_HKEY hEndorsementPubKey //in
);
```

输入参数描述：

——hTCM	TCM 对象句柄。
——hKeySMK	SMK 的密钥对象句柄。
——hEndorsementPubKey	EK 公钥的密钥对象句柄, 该密钥用于加密 SMK 秘密和 TCM 秘密。

输出参数描述：

——无

返回参数

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INTERNAL_ERROR
```

#### 5.4.12 清除可信密码模块所有者

功能描述：

本函数清除 TCM 的所有权。

接口定义：

```
TSM_RESULT Tspi_TCM_ClearOwner
(
    TSM_HTCM hTCM,           //in
    TSM_BOOL fForcedClear    //in
);
```

输入参数描述：

——hTCM	TCM 对象句柄。
——fForcedClear	若为 FALSE, 将执行需要 TCM 所有者授权的清除命令。否则, 将执行需要提供物理访问授权的清除命令。

输出参数描述：

——无

返回参数

TSM\_SUCCESS  
TSM\_E\_INVALID\_HANDLE  
TSM\_E\_INTERNAL\_ERROR

#### 5.4.13 设置操作者授权

功能描述：

本函数设置 TCM 的操作者授权值。操作者可以是平台管理员或用户，操作者必须物理的操作平台。如果执行成功，hOperatorPolicy 就会赋给 hTCM 对象作为操作者授权策略。此命令中，操作者的授权是明文形式。

接口定义：

```
TSM_RESULT Tspi_TCM_SetOperatorAuth
(
    TSM_HTCM hTCM,           //in
    TSM_HPOLICY hOperatorPolicy //in
);
```

输入参数描述：

——hTCM TCM 对象句柄。  
——hOperatorPolicy 策略对象句柄，该策略对象包含新授权信息。

输出参数描述：

——无。

返回参数：

TSM\_SUCCESS  
TSM\_E\_INVALID\_HANDLE  
TSM\_E\_BAD\_PARAMETER  
TSM\_E\_INTERNAL\_ERROR

#### 5.4.14 设置可信密码模块状态

功能描述：

本函数用于修改 TCM 的状态。

接口定义：

```
TSM_RESULT Tspi_TCM_SetStatus
(
    TSM_HTCM hTCM,           //in
    TSM_FLAG statusFlag,     //in
    TSM_BOOL fTcmState       //in
);
```

输入参数描述：

——hTCM TCM 管理对象句柄。  
——statusFlag 表示待设置的状态或属性。  
——fTcmState 表示待设置的状态或属性值。

属性定义：

属性	FTcmState 状态值	描述
TSM_TCMSTATUS_DISABLE_OWNERCLEAR	忽略	永久禁止 TCM 所有者进行 ClearOwner 操作。此时,方法 ClearOwner() 中的 fForcedClear 参数将不再允许取 FALSE 值。这个设置需要所有者授权。
TSM_TCMSTATUS_DISABLE_FORCECLEAR	忽略	临时禁止 TCM 所有者的强制清除操作(这种禁止只在本次系统运行时有效,在下一次系统重新启动时将被取消)。此时,方法 ClearOwner() 中的 fForcedClear 参数将暂时不允许取 TRUE 值(直到下次系统重新启动为止)。
TSM_TCMSTATUS OWNERS_ETDISABLE	TSM_BOOL	fTCMState=TRUE: 表示设置 TCM 的状态为 Disabled。该命令需要 TCM 所有者的授权。
TSM_TCMSTATUS_PHYSICALDISABLE	TSM_BOOL	fTCMState=TRUE: 表示设置 TCM 的状态为 Disabled。该命令必须是物理现场的。
TSM_TCMSTATUS_PHYSICALSETDEACTIVATED	TSM_BOOL	fTCMState=TRUE: 表示设置 TCM 的状态为 Deactivated。该命令必须是物理现场的。
TSM_TCMSTATUS_SETTEMPDEACTIVATED	忽略	暂时将 TCM 的状态设置为 Deactivated(直到下次系统重新启动为止)。
TSM_TCMSTATUS_SETOWNERRINSTALL	TSM_BOOL	fTCMState=TRUE: 表示允许使用 TakeOwnership() 方法来取得 TCM 的所有者关系。这个操作需要物理现场。
TSM_TCMSTATUS_DISABLE_PUBEKREAD	TCMs: TSM_BOOL	永久禁止在没有 TCM 所有者授权的情况下读取 EK 公钥信息的操作,即设置该属性后,必须有 TCM 所有者授权才能进行读取 EK 公钥信息的操作。设置了这个属性后,GetPubEndorsementKey() 方法中的 fOwnerAuthorized 参数取 FALSE 值不可能再有效了。设置这个属性值需要所有者授权。
TSM_TCMSTATUS_DISABLE	TSM_BOOL	将 TCM 设置为可用或不可用状态。
TSM_TCMSTATUS_DEACTIVATED	TSM_BOOL	将 TCM 设置为激活或非激活状态。

输出参数描述：

——无。

返回参数：

TSM\_SUCCESS  
 TSM\_E\_INVALID\_HANDLE  
 TSM\_E\_BAD\_PARAMETER  
 TSM\_E\_INTERNAL\_ERROR

#### 5.4.15 查询设置可信密码模块状态

功能描述：

查询 TCM 状态。

读取 TCM 状态标记需要所有者授权。

接口定义：

```
TSM_RESULT Tspi_TCM_GetStatus
(
    TSM_HTCM hTCM,           //in
    TSM_FLAG statusFlag,     //in
    TSM_BOOL* pfTcmState     //out
);
```

输入参数描述：

——hTCM TCM 管理对象句柄。  
——statusFlag 表示待获取的状态或属性。

输出参数描述：

——fTcmState 表示待获取的状态或属性值。

属性定义：

属性	描述
TSM_TCMSTATUS_DISABLEOWNERCLEAR	* pfTCMState=TRUE:表示 TCM 所有者授权清除所有者关系的操作已经被永久禁止了。ClearOwner()方法中输入参数 fForcedClear 取值为 FALSE 时将永远无效。
TSM_TCMSTATUS_DISABLEFORCECLEAR	* pfTCMState=TRUE:表示强制清除 TCM 的操作被暂时禁止了(直到系统下一次重新启动时为止)。ClearOwner()的输入参数 fForcedClear 取值为 TRUE 也将暂时无意义了。
TSM_TCMSTATUS_DISABLED	* pfTCMState=TRUE:表示 TCM 的属性标志位 disabledfTCMState=TRUE。
TSM_TCMSTATUS_SETTEMPDEACTIVATED	* pfTCMState=TRUE:表示 TCM 暂时地被 Deactivated 了。
TSM_TCMSTATUS_SETOwnerInstall	* pfTCMState=TRUE:表示可以使用 TakeOwnership()方法来获取所有者关系。
TSM_TCMSTATUS_DISABLEPUBEKREAD	* pfTCMState=TRUE:表示永远禁止没有 TCM 所有者授权就可以进行读取 EK 公钥的操作,即读取 EK 公钥时必须进行 TCM 所有者授权。 GetPubEndorsementKey()方法中的输入参数 fOwnerAuthorized 取 FALSE 值将不再有效。需要所有者授权。
TSM_TCMSTATUS_PHYSPRES_LIFETIMELOCK	* pfTCMState=TRUE:表示在 TCM 生存期内,TCM 的 physicalPresenceHWEable 和 physicalPresenceCMDEnable 标志都不允许修改。
TSM_TCMSTATUS_PHYSPRES_HWENABLE	* pfTCMState=TRUE:表示 TCM 物理在线的硬件信号被允许可以用做物理在线的标志。
TSM_TCMSTATUS_PHYSPRES_CMDENABLE	* pfTCMState=TRUE:表示允许使用 TCM 命令 TSC_PhysicalPresence 来表明物理在线。

表 (续)

属性	描述
TSM_TCMSTATUS_CEKP_USED	* pfTCMState=TRUE: 表明 EK 密钥对是使用 CreateEndorsementKey()方法来生成的。 * pfTCMState= FALSE: 表明 EK 密钥对是由厂家创建的。
TSM_TCMSTATUS_PHYSPRESENCE	* pfTCMState= TRUE: 表示物理在线的软件标志。
TSM_TCMSTATUS_PHYSPRES_LOCK	* pfTCMState= TRUE: 表示改变物理在线的标志的操作是不允许的。
TSM_TCMSTATUS_ENABLE_REVKEEK	表明是否允许 EK 被重新设置
TSM_TCMSTATUS_NV_LOCK	* pfTCMState= TRUE: 表示 NV 授权访问是必需的;

返回参数：

TSM\_SUCCESS  
TSM\_E\_INVALID\_HANDLE  
TSM\_E\_BAD\_PARAMETER  
TSM\_E\_INTERNAL\_ERROR

#### 5.4.16 获取可信密码模块特性

功能描述：

获取 TCM 功能特性。有些功能特性信息必须提供所有者授权信息才能获得。

接口定义：

```
TSM_RESULT Tspi_TCM_GetCapability
(
    TSM_HTCM hTCM,                //in
    TSM_FLAG capArea,             //in
    UINT32 ulSubCapLength,        //in
    BYTE* rgbSubCap,              //in
    UINT32* pulRespDataLength,    //out
    BYTE** prgbRespData          //out
);
```

输入参数描述：

——hTCM TCM 对象句柄。  
——capArea 指示查询属性的标志。  
——ulSubCapLength rgSubCap 参数的字节长度。  
——rgbSubCap 指示要查询的属性数据。

输出参数描述：

——pulRespDataLength prgbRespData 参数的字节长度。  
——prgbRespData 返回的指定属性实际数据。

## 属性定义：

属性	子属性	描述
TSM_TCMCAP_ORD	命令码	返回布尔值。TRUE 表示 TCM 支持该命令，FALSE 表示 TCM 不支持该命令。
TSM_TCMCAP_FLAG	忽略	永久性和易失性的比特标志位。
TSM_TCMCAP_ALG	TSM_ALG_XX:	返回布尔值(TSM 算法 ID 的值)。TRUE 表示 TCM 支持该算法, FALSE 表示 TCM 不支持该算法。
TSM_TCMCAP_PROPERTY	TSM_TCMCAP_PROP_PCR	UINT32 值。返回 TCM 支持的 PCR 寄存器个数。
	TSM_TCMCAP_PROP_PCR_MAP	返回 TCM_PCR_ATTRIBUTES 的比特标志位
	TSM_TCMCAP_PROP_MANUFACTURER	UINT32 值。返回 TCM 厂商的标识。
	TSM_TCMCAP_PROP_SLOTS 或者 TSM_TCMCAP_PROP_KEYS	UINT32 值。返回 TCM 可以加载的 256 位 SM2 密钥的最大个数。可随时间和情况而改变。
	TSM_TCMCAP_PROP_OWNER	布尔值。返回 TRUE 表示 TCM 成功地创建了一个所有者。
	TSM_TCMCAP_PROP_MAXKEYS	UINT32 值。返回 TCM 所支持的 256 位 SM2 密钥的最大个数,不含 EK。
	TSM_TCMCAP_PROP_AUTHSESSIONS	UINT32 值。可用的授权会话的个数,可随时间和情况而改变。
	TSM_TCMCAP_PROP_MAXAUTHSESSIONS	UINT32 值。返回 TCM 支持的可加载授权会话的最大个数,可随时间和情况而改变。
	TSM_TCMCAP_PROP_TRANSACTIONS	UINT32 值。返回可用传输会话的个数,可随时间和情况而改变。
	TSM_TCMCAP_PROP_MAXTRANSACTIONS	UINT32 值。返回 TCM 支持的可加载传输会话的最大个数。
	TSM_TCMCAP_PROP_SESSIONS	UINT32 值。返回会话池中可用会话的个数。会话池中的会话包括授权会话和传输会话,可随时间和情况而改变。
	TSM_TCMCAP_PROP_MAXSESSIONS	UINT32 值。返回 TCM 支持的最大会话个数,包括授权会话和传输会话。
	TSM_TCMCAP_PROP_CONTEXTS	UINT32 值。返回可保存的会话个数,可随时间和情况而改变。
	TSM_TCMCAP_PROP_MAXCONTEXTS	UINT32 值。返回保存的会话的最大个数。
	TSM_TCMCAP_PROP_COUNTERS	UINT32 值。返回可用单调计数器的个数,可随时间和情况而改变。
	TSM_TCMCAP_PROP_MAXCOUNTERS	UINT32 值。返回 TCM_CreateCounter 控制的单调计数器的最大个数。

表 (续)

属性	子属性	描述
	TSM_TCMCAP_PROP_ACT IVECOUNTER	TCM_COUNT_ID。返回当前计数器的 ID。 若没有活动的计数器，则返回 0xff..ff。
	TSM_TCMCAP_PROP_MIN COUNTERINCTIME	UINT32 值。表示单调计数器递增的最长时间 间隔，该间隔以 1/10 秒为度量单位。
	TSM_TCMCAP_PROP_TIST IMEOUTS	UINT32 值四元数组，每个元素表示以毫秒计 的超时值，顺序如下：TIMEOUT_A， TIMEOUT_B，TIMEOUT_C，TIMEOUT_D。 由平台特定的接口规范决定在何处使用这些 超时值。
	TSM_TCMCAP_PROP_STA RTUPEFFECTS	返回 TCM_STARTUP_EFFECTS 结构。
	TSM_TCMCAP_PROP_MA XCONTEXTCOUNTDIST	UINT32 值。返回上下文计数值的最大间距， 至少必须为 2 <sup>16</sup> -1。
	TSM_TCMCAP_PROP_DUR ATION	返回 UINT32 值三元数组，每个元素依次表示 如下三类命令以毫秒计的周期值： SMALL_DURATION， MEDIUM_DURATION，LONG_DURATION
	TSM_TCMCAP_PROP_MA XNVAVAILABLE	UINT32 值。返回可分配的 NV 区域的最大个 数，可随时间和情况而改变。
	TSM_TCMCAP_PROP_INP UTBUFFERSIZE	UINT32 值。返回 TCM 输入缓冲区的大小 (字节)。
	TSM_TCMCAP_PROP_MA XNVWRITE	UINT32 值。未建立 TCM 所有者前发生 的 NV 写操作次数。
	TSM_TCMCAP_PROP_REV ISION	字节：这是标准结构中的 TCM 版本号。
	TSM_TCMCAP_PROP_ORD _AUDITED	表格：正被审计的命令码
	TSM_TCMCAP_PROP_LOC ALITIES_AVAIL	TCM 中可用的 localities 个数。
TSM_TCMCAP_VERSION	忽略	返回一个标识 TCM 版本的 TSM_VERSION 结构。
TSM_TCMCAP_VERSION_VA L	忽略	查询 TCM 的版本号
TSM_TCMCAP_NV_LIST	忽略	获取用来定义 NV 存储区的索引列表。
TSM_TCMCAP_NV_INDEX	UINT32(TCM_NV_INDEX)	获取 TCM_NV_DATA_PUBLIC 结构，此结构 表示指定 NV 存储区的值。
TSM_TCMCAP_MFR	忽略	获取厂商特定的 TCM 和 TCM 状态信息。

表 (续)

属性	子属性	描述
TSM_TCMCAP_SYM_MODE	TCM_SYM_MODE_* type 类型的一种加密	布尔值。查询 TCM 是否支持特定类型的对称加密。
TSM_TCMCAP_HANDLE	TCM_RT_* values 值之一,说明是否应当返回密钥、授权会话、传输会话列表。	UINT32 值数组 (TCM 句柄), 返回 TCM 中当前已加载对象的句柄列表
TSM_TCMCAP_TRANS_ES	TSM_ES_* values 值	布尔值。查询 TCM 是否在传输会话中支持特定的加密方案。
TSM_TCMCAP_AUTH_ENCRYPTION	TSM_ALGORITHM_ID values 值	布尔值。查询 TCM 是否在授权会话的 AuthData 加密中支持特定的加密方案。

返回参数:

TSM\_SUCCESS  
TSM\_E\_INVALID\_HANDLE  
TSM\_E\_BAD\_PARAMETER  
TSM\_E\_INTERNAL\_ERROR

#### 5.4.17 可信密码模块完全自检

功能描述:

在初始化 TCM 时对每个内部功能都执行自检。

接口定义:

```
TSM_RESULT Tspi_TCM_SelfTestFull
(
    TSM_HTCM hTCM,      //in
);
```

输入参数描述:

——hTCM TCM 对象句柄。

输出参数描述:

——无。

返回参数:

TSM\_SUCCESS  
TSM\_E\_INVALID\_HANDLE  
TSM\_E\_INTERNAL\_ERROR

#### 5.4.18 获取可信密码模块自检结果

功能描述:

返回厂商关于该自检结果的说明信息。

接口定义:

```
TSM_RESULT Tspi_TCM_GetTestResult
(
    TSM_HTCM hTCM,          //in
```

```

    UINT32* pulTestResultLength,      // out
    BYTE** prgbTestResult          // out
);

```

输入参数描述：

——hTCM TCM 对象句柄。

输出参数描述：

——pulTestResultLength prgbTestResult 参数的字节长度。

——prgbTestResult TCM 制造商定义的信息。

返回参数：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR

```

#### 5.4.19 获取可信密码模块产生的随机数

功能描述：

获得由 TCM 产生的随机数。

接口定义：

```

TSM_RESULT Tspi_TCM_GetRandom
(
    TSM_HTCM hTCM,                // in
    UINT32 ulRandomDataLength,     // in
    BYTE** prgbRandomData        // out
);

```

输入参数描述：

——hTCM TCM 对象句柄。

——ulRandomDataLength 请求的随机数长度。

输出参数描述：

——prgbTestResult 返回随机数数据指针。

返回参数：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR

```

#### 5.4.20 获取可信密码模块单个事件

功能描述：

对一个给定的 PCR 索引和事件序号，返回一个 PCR 事件。

接口定义：

```

TSM_RESULT Tspi_TCM_GetEvent
(
    TSM_HTCM hTCM,                // in
    UINT32 ulPcrIndex,             // in

```

```

    UINT32 ulEventNumber,           // in
    TSM_PCR_EVENT* pPcrEvent     // out
);

```

输入参数描述：

——hTCM TCM 对象句柄。  
 ——ulPcrIndex 请求的 PCR 索引。  
 ——ulEventNumber 请求的事件序号。

输出参数描述：

——pPcrEvent 返回的 PCR 事件数据。

返回参数：

TSM\_SUCCESS  
 TSM\_E\_INVALID\_HANDLE  
 TSM\_E\_BAD\_PARAMETER  
 TSM\_E\_INTERNAL\_ERROR

#### 5.4.21 获取可信密码模块一组事件

功能描述：

对一个给定 PCR 索引，返回一组指定个数的 PCR 事件。

本函数为请求的事件数据分配内存。使用 Tspi\_Context\_FreeMemory 释放该内存。

接口定义：

```

TSM_RESULT Tspi_TCM_GetEvents
(
    TSM_HTCM hTCM,           // in
    UINT32 ulPcrIndex,       // in
    UINT32 ulStartNumber,    // in
    UINT32* pulEventNumber,  // in,out
    TSM_PCR_EVENT** prgPcrEvents // out
);

```

输入参数描述：

——hTCM TCM 对象句柄。  
 ——ulPcrIndex 请求的 PCR 索引。  
 ——ulStartNumber 请求的第一个事件的索引  
 ——pulEventNumber 请求的事件个数。

输出参数描述：

——pulEventNumber 获得的 prgPcrEvents 参数的事件数据结构个数。  
 ——prgPcrEvents 指向 PCR 事件数据数组。如果是 NULL，只有事件的个数在 pulEvent-  
 Number 参数中返回。

返回参数：

TSM\_SUCCESS  
 TSM\_E\_INVALID\_HANDLE  
 TSM\_E\_BAD\_PARAMETER  
 TSM\_E\_INTERNAL\_ERROR

#### 5.4.22 获取可信密码模块事件日志

功能描述：

返回全部的事件日志。

接口定义：

```
TSM_RESULT Tspi_TCM_GetEventLog
(
    TSM_HTCM hTCM,           //in
    UINT32* pulEventNumber,   //out
    TSM_PCR_EVENT** prgPcrEvents //out
);
```

输入参数描述：

——hTCM TCM 对象句柄。

输出参数描述：

——pulEventNumber 获得的 prgPcrEvents 参数的事件数据结构个数。

——prgPcrEvents 指向 PCR 事件数据数组。如果是 NULL, 只有事件的个数在 pulEventNumber 参数中返回。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

#### 5.4.23 可信密码模块 PCR 扩展

功能描述：

该函数扩展 PCR 寄存器并记录 PCR 事件日志。

接口定义：

```
TSM_RESULT Tspi_TCM_PcrExtend
(
    TSM_HTCM hTCM,           //in
    UINT32 ulPcrIndex,        //in
    UINT32 ulPcrDataLength,   //in
    BYTE* pbPcrData,          //in
    TSM_PCR_EVENT* pPcrEvent, //in
    UINT32* pulPcrValueLength, //out
    BYTE** prgbPcrValue       //out
);
```

输入参数描述：

——hTCM TCM 对象句柄。

——ulPcrIndex 需要扩展的 PCR 索引。

——ulPcrDataLength 需要扩展的数据(参数 pbPcrData)的长度。

——pbPcrData PCR 扩展操作数据。如果 pPcrEvent 不为 NULL, 则该数据根据 TSM\_PCR\_EVENT 结构的 rgbPcrValue 参数描述, 被用于杂凑计算创建。如果 pPcrEvent 为 NULL, 则该数据被直接扩展到 TCM 中而无需 TSP 处理。

——pPcrEvent 指向一个 TSM\_PCR\_EVENT 结构, 该结构包含事件入口信息。如果该指针是 NULL, 则没有事件入口被创建, 该操作仅执行一个扩展操作。如果该指针为非 NULL, 则结构成员被 TSM 设置。

输出参数描述:

——pulPcrValueLength 参数 prgbPcrValue 的字节长度。  
——prgbPcrValue 指向的内存块包含扩展操作后的 PCR 数据。

返回参数:

TSM\_SUCCESS  
TSM\_E\_INVALID\_HANDLE  
TSM\_E\_BAD\_PARAMETER  
TSM\_E\_INTERNAL\_ERROR

#### 5.4.24 读取可信密码模块 PCR 值

功能描述:

读一个 PCR 寄存器。

注: 本功能为 prgbPcrValue 数据分配内存, 该内存块必须调用 Tspi\_Context\_FreeMemory 功能释放。

接口定义:

```
TSM_RESULT Tspi_TCM_PcrRead
(
    TSM_HTCM hTCM,           //in
    UINT32 ulPcrIndex,       //in
    UINT32* pulPcrValueLength, //out
    BYTE** prgbPcrValue      //out
);
```

输入参数描述:

——hTCM TCM 对象句柄。。  
——ulPcrIndex 要读的 PCR 索引号。

输出参数描述:

——pulPcrValueLength 参数 prgbPcrValue 的字节长度。  
——prgbPcrValue 读取到的 PCR 数据。

返回参数:

TSM\_SUCCESS  
TSM\_E\_INVALID\_HANDLE  
TSM\_E\_BAD\_PARAMETER  
TSM\_E\_INTERNAL\_ERROR

#### 5.4.25 重置可信密码模块 PCR

功能描述:

重置 TCM 内的一个 PCR 寄存器的值。

接口定义：

```
TSM_RESULT Tspi_TCM_PcrReset
(
    TSM_HTCM hTCM,           //in
    TSM_HPCR hPcrComposite   //in
);
```

输入参数描述：

——hTCM TCM 对象句柄。  
——hPcrComposite PcrComposite 对象句柄

输出参数描述：

——无。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_TCM_NOT_RESETABLE
TSM_E_WRONG_LOCALITY
TSM_E_INTERNAL_ERROR
```

#### 5.4.26 引证 PCR

功能描述：

该函数调用提供当前平台的完整配置信息。

接口定义：

```
TSM_RESULT Tspi_TCM_Quote
(
    TSM_HTCM hTCM,           //in
    TSM_HKEY hIdentKey,       //in
    TSM_HPCR hPcrComposite,   //in
    TSM_VALIDATION* pValidationData, //in,out
);
```

输入参数描述：

——hTCM TCM 对象句柄。  
——hIdentKey 签名密钥对象句柄。  
——hPcrComposite PCRcomposite 对象句柄, 必须是 TSM\_PCRS\_STRUCT\_INFO\_LONG 结构类型。  
——pValidationData 验证数据结构。输入时提供外部数据, 计算签名时使用。

输出参数描述：

——pValidationData 验证数据结构。输出时若成功执行该命令, 则提供一个内存区, 容纳验证数据和签名数据。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

#### 5.4.27 读可信密码模块计数器

功能描述：

读取当前活动的单调计数器的值。

接口定义：

```
TSM_RESULT Tspi_TCM_ReadCounter
(
    TSM_HTCM hTCM,           //in
    UINT32* counterValue     //out
);
```

输入参数描述：

——hTCM TCM 对象句柄。

输出参数描述：

——counterValue 单调计数器的当前值。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

#### 5.4.28 读可信密码模块当前时钟

功能描述：

读取 TCM 内的当前时钟计数。

接口定义：

```
TSM_RESULT Tspi_TCM_ReadCurrentTicks
(
    TSM_HTCM hTCM,           //in
    TCM_CURRENT_TICKS* tickCount //out
);
```

输入参数描述：

——hTCM TCM 对象句柄。

输出参数描述：

——tickCount 时钟计数。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INTERNAL_ERROR
```

#### 5.4.29 获取可信密码模块审计摘要值

功能描述：

返回目前日志的摘要，一般而言，此值对每个 TCM 是唯一的。

若摘要被签名，则返回当前摘要、当前审计计数器值、被审计命令码的杂凑值、以及签名；若摘要未作签名，则返回当前摘要、当前审计计数器值、被审计命令码。

接口定义：

```
TSM_RESULT Tspi_TCM_GetAuditDigest
(
    TSM_HTCM hTCM,                      //in
    TSM_HKEY hKey,                       //in
    TSM_BOOL closeAudit,                 //in
    TCM_DIGEST* pAuditDigest,            //out
    TCM_COUNTER_VALUE* pCounterValue,   //out
    TSM_VALIDATION* pValidationData,    //out
    UINT32* ordSize,                    //out
    UINT32** ordList                   //out
);
```

输入参数描述：

- hTCM TCM 对象句柄。
- hKey 执行签名的密钥的句柄。
- closeAudit 此标志用于表示是否在签名之后结束当前审计会话。只用于 hKey 为非 NULL 的情况。

输出参数描述：

- pAuditDigest TCM 内的审计摘要值。
- pCounterValue 审计单调计数器的当前值。
- pValidationData 验证数据。当 hKey 为 NULL 时，此参数被忽略。当 hKey 为非 NULL 时，为验证签名所必要的信息。
- ordSize ordList 中被审计的命令个数。只在 hKey 为 NULL 时返回该值。
- ordList 审计命令链表。只在 hKey 为 NULL 时返回该值。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

#### 5.4.30 设置可信密码模块命令审计状态

功能描述：

设置一个 TCM 命令的审计状态。

接口定义：

```
TSM_RESULT Tspi_TCM_SetOrdinalAuditStatus
(
    TSM_HTCM hTCM,                      //in
    TCM_COMMAND_CODE ordinalToAudit,    //in
    TSM_BOOL auditState                 //in
);
```

输入参数描述：

- hTCM TCM 对象句柄。
- ordinalToAudit 需要审计的 TCM 命令码。
- auditState 命令的审计状态：TRUE，该命令需要审计；FALSE，命令不需要审计。

输出参数描述：

——无。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

## 5.5 密钥管理

### 5.5.1 概述

密钥管理类用于表示 TSM 密钥管理功能的入口。每个密钥对象的实例代表 TSM 密钥树中一个具体的密钥节点。每个需授权的密钥对象，需要分配一个策略对象，用于管理授权秘密信息。

### 5.5.2 改变实体授权数据

功能描述：

改变实体(对象)的授权数据并将该对象指派给策略对象。所有使用授权数据的类都提供本函数用以改变它们的授权数据。

接口定义：

```
TSM_RESULT Tspi_ChangeAuth
(
    TSM_HOBJECT hObjectToChange,           //in
    TSM_HOBJECT hParentObject,             //in
    TSM_HPOLICY hNewPolicy                //in
);
```

输入参数描述：

——hObjectToChange 需要修改授权数据的对象句柄。

——hParentObject 需要修改授权对象的父对象句柄。

——hNewPolicy 更新的授权信息策略对象句柄。

输出参数描述：

——无。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INTERNAL_ERROR
```

### 5.5.3 获取策略对象

功能描述：

返回当前工作对象的策略。如果应用程序没有创建策略对象，且在调用前没有为该工作对象指派策略，本函数将返回默认的上下文策略。为默认上下文策略设置新的授权数据信息，将影响与其相关的所有对象的后续操作。

接口定义：

```
TSM_RESULT Tspi_GetPolicyObject
(
    TSM_HOBJECT hObject,           //in
    TSM_FLAG policyType,          //in
    TSM_HPOLICY* phPolicy        //out
);
```

输入参数描述：

- hObject 对象句柄。
- policyType 定义的策略类型。
- 输出参数描述：

——phPolicy 返回指派的策略对象。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

#### 5.5.4 设置密钥属性(整型参数)

功能描述：

本函数设置密钥对象的 32 比特属性。如果请求的数据长度小于 UINT32, 必须将数据转换成正确的大小。

接口定义：

```
TSM_RESULT Tspi_SetAttribUint32
(
    TSM_HOBJECT hObject,           //in
    TSM_FLAG attribFlag,          //in
    TSM_FLAG subFlag,             //in
    UINT32 ulAttrib               //in
);
```

输入参数描述：

- hObject 需要设置属性的对象句柄。
- attribFlag 需要设置的属性。
- subFlag 需要设置的子属性。
- ulAttrib 属性设置的值。

属性定义：

属性标记	子属性标记	属性值	描述
TSM_TSPATTRIB_KEYREGISTER	0	TSM_TSPATTRIB_KEYREGISTER_USER	密钥注册到 TSP 中
	0	TSM_TSPATTRIB_KEYREGISTER_SYSTEM	密钥注册到 TCS 中

表 (续)

属性标记	子属性标记	属性值	描述
	0	TSM_TSPATTRIB_KEYREGISTER_NO	密钥没有在 TSM 中注册
TSM_TSPATTRIB_KEY_INFO	TSM_TSPATTRIB_KEYINFO_USAGE	TSM_KEYUSAGE_XX	TSM 密钥使用值, 表示密钥的使用类型。见密钥对象的属性子标记定义。
	TSM_TSPATTRIB_KEYINFO_MIGRATABLE	布尔值。	若为 TRUE, 则密钥是可迁移的。
	TSM_TSPATTRIB_KEYINFO_VOLATILE	布尔值。	若为 TRUE, 则密钥是易失性的。
	TSM_TSPATTRIB_KEYINFO_AUTHDATAUSAGE	布尔值。	若为 TRUE, 则密钥的使用需要授权。
	TSM_TSPATTRIB_KEYINFO_ALGORITHM	TSM_ALG_XX	TSM 算法 ID, 表示密钥算法。见算法 ID 定义。
	TSM_TSPATTRIB_KEYINFO_ENCSHEME	TSM_KEY_ENCSCH_EME_XX	TSM 加密方案, 见密钥加密方案定义。
	TSM_TSPATTRIB_KEYINFO_SIGSCHEME	TSM_KEY_SIGSCH_EME_XX	TSM 签名方案。见密钥签名方案定义。
	TSM_TSPATTRIB_KEYINFO_SIZE		密钥位长。
	TSM_TSPATTRIB_KEYINFO_KEYFLAGS		密钥标志信息。
	TSM_TSPATTRIB_KEYINFO_AUTHUSAGE		直接设置 KeyParams 中的 authDataUsage。

输出参数描述:

——无。

返回参数:

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
TSM_E_INVALID_ATTRIB_SUBFLAG
TSM_E_INVALID_ATTRIB_DATA
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR

```

### 5.5.5 获取密钥属性(整型参数)

功能描述：

本函数获取密钥对象的 32 比特属性。

接口定义：

```
TSM_RESULT Tspi_GetAttribUInt32
(
    TSM_HOBJECT hObject,           //in
    TSM_FLAG attribFlag,          //in
    TSM_FLAG subFlag,             //in
    UINT32* pulAttrib            //out
);
```

输入参数描述：

——hObject 需要查询属性的对象句柄。

——attribFlag 需要查询的属性。

——subFlag 需要查询的子属性。

属性定义：

属性标记	子属性标记	属性值	描述
TSM_TSPATTRIB_KEYREGISTER	0	TSM_TSPATTRIB_KEYREGISTER_USER	密钥注册到用户库中
	0	TSM_TSPATTRIB_KEYREGISTER_SYSTEM	密钥注册到系统库中
	0	TSM_TSPATTRIB_KEYREGISTER_NO	密钥没有在 TSM 中注册
TSM_TSPATTRIB_KEY_INFO	TSM_TSPATTRIB_KEYINFO_USAGE	TSM_KEYUSAGE_XX	TSM 密钥使用值, 表示密钥的使用类型。见密钥对象的属性子标记定义。
	TSM_TSPATTRIB_KEYINFO_MIGRATABLE	布尔值。	若为 TRUE, 则密钥是可迁移的。
	TSM_TSPATTRIB_KEYINFO_VOLATILE	布尔值。	若为 TRUE, 则密钥是易失性的。
	TSM_TSPATTRIB_KEYINFO_AUTHDATAUSAGE	布尔值。	若为 TRUE, 则密钥的使用需要授权。
	TSM_TSPATTRIB_KEYINFO_ALGORITHM	TSM_ALG_XX	TSM 算法 ID 值, 表示密钥算法。见算法 ID 定义。
	TSM_TSPATTRIB_KEYINFO_ENCSHEME	TSM_KEY_ENCSHEME_XX	TSM 加密方案, 见密钥加密方案定义。
	TSM_TSPATTRIB_KEYINFO_SIGSCHEME	TSM_KEY_SIGSCHEME_X_X	TSM 签名方案。见密钥签名方案定义。
	TSM_TSPATTRIB_KEYINFO_KEYFLAGS		密钥标志信息。

表 (续)

属性标记	子属性标记	属性值	描述
	TSM_TSPATTRIB_KEYINFO_AUTHUSAGE		返回 authDataUsage 的内容。
	TSM_TSPATTRIB_KEYINFO_KEYSTRUCT	TSM_KEY_STRUCT_XX	密钥结构类型。见密钥结构类型定义。
	TSM_TSPATTRIB_KEYINFO_SIZE		密钥位长。
TSM_TSPATTRIB_KEY_PCR	TSM_TSPATTRIB_KEYPCR_LOCALITY_ATCREATION	创建 blob 时的 locality modifier	
	TSM_TSPATTRIB_KEYPCR_LOCALITY_ATRELEASE	使用密钥所要求的 locality modifier	

输出参数描述：

——pulAttrib 指向查询到的属性值。

返回参数：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
TSM_E_INVALID_ATTRIB_SUBFLAG
TSM_E_INVALID_ATTRIB_DATA
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR

```

### 5.5.6 设置密钥属性(变长参数)

功能描述：

设置密钥对象的非 32 比特属性。属性数据的结构和大小依赖于属性。

接口定义：

```

TSM_RESULT Tspi_SetAttribData
(
    TSM_HOBJECT hObject,           //in
    TSM_FLAG attribFlag,          //in
    TSM_FLAG subFlag,             //in
    UINT32 ulAttribDataSize,       //in
    BYTE* rgbAttribData          //in
);

```

输入参数描述：

——hObject 需要设置属性的对象句柄。

——attribFlag 需要设置的属性的标志。(见属性说明表)

——subFlags 需要设置的属性的子标志。(见属性说明表)

属性说明表：

属性标记	属性子标记	数据描述
TSM_TSPATTRIB_KEY_BLOB	TSM_TSPATTRIB_SM2KEYBLOB_BLOB	SM2 密钥数据块
	TSM_TSPATTRIB_SM2KEYBLOB_PUBLIC_KEY	SM2 密钥数据块的公钥部分
	TSM_TSPATTRIB_SM2KEYBLOB_PRIVATE_KEY	已加密的 SM2 密钥数据块的公钥部分
	TSM_TSPATTRIB_SM4KEYBLOB_BLOB	SM4 密钥数据块

——ulAttribDataSize prgbAttribData 参数的大小(以字节为单位)。

——rgbAttribData 此参数指向一个存放设置属性值的缓冲区。

返回值：

——无。

### 5.5.7 获取设置密钥属性(变长参数)

功能描述：

获取密钥对象的非 32 比特属性。属性数据的结构和大小依赖于属性。

接口定义：

```
TSM_RESULT Tspi_GetAttribData
(
    TSM_HOBJECT hObject,           // in
    TSM_FLAG attribFlag,          // in
    TSM_FLAG subFlag,             // in
    UINT32* pulAttribDataSize,     // out
    BYTE** prgbAttribData        // out
);
```

输入参数描述：

——hObject 需要获取属性的对象句柄。

——attribFlag 需要获取的属性的标记。(见属性说明表)

——subFlags 需要获取的子属性的标记。(见属性说明表)

属性说明表：

属性标记	子属性标记	数据描述
TSM_TSPATTRIB_KEY_BLOB	TSM_TSPATTRIB_KEYBLOB_BLOB	以密钥数据块形式返回密钥信息。
	TSM_TSPATTRIB_KEYBLOB_PUBLIC_KEY	SM2 密钥数据块的公钥信息。
	TSM_TSPATTRIB_KEYBLOB_PRIVATE_KEY	SM2 密钥数据块的已被加密的私钥信息。
	TSM_TSPATTRIB_SM4KEYBLOB_BLOB	SM4 密钥数据块的密钥信息。
TSM_TSPATTRIB_KEY_INFO	TSM_TSPATTRIB_KEYINFO_VERSION	返回数据 prgbAttribData 以 TSM_VERSION 结构返回的版本信息。

表 (续)

属性标记	子属性标记	数据描述
TSM_TSPATTRIB_KEY_UUID	0	包含 UUID 由密钥来赋值的 TSM_UUID 结构
TSM_TSPATTRIB_KEY_PCR_CREATION_PCR_SELECTION	TSM_TSPATTRIB_KEYPCR_CREATION_PCR_SELECTION	创建数据块时所选择的 PCR。
	TSM_TSPATTRIB_KEYPCR_RELEASE_PCR_SELECTION	在用到密钥时所选择的 PCR。
	SS_TSPATTRIB_KEYPCR_DIGEST_ATCREATION	创建 PCR selection 结构时对应的 PCR 摘要。
	TSM_TSPATTRIB_KEYPCR_DIGEST_ATRELEASE	释放 PCR selection 结构时对应的 PCR 摘要。

输出参数描述：

—— pulAttribDataSize 返回的 prgbAttribData 参数的大小(以字节为单位)。

—— prgbAttribData 命令成功返回,此参数指向一个存放指定属性值的缓冲区。

返回值：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
TSM_E_INVALID_ATTRIB_SUBFLAG
TSM_E_INVALID_ATTRIB_DATA
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR

```

### 5.5.8 加载密钥

功能描述：

将密钥加载到 TCM 中,TCM 负责将密钥解密,并缓存加载在 TCM 中;只有执行了密钥加载后,才能使用密钥进行加、解密和签名等密钥服务功能。

接口定义：

```

TSM_RESULT Tspi_Key_LoadKey
(
    TSM_HKEY hKey,           // in
    TSM_HKEY hUnwrappingKey // in
);

```

输入参数描述：

—— hKey 被载入的密钥对象的句柄。

—— hUnwrappingKey 用于解开 hKey 的密钥句柄。

输出参数描述：

—— 无

返回值：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

### 5.5.9 卸载密钥

功能描述：

卸载 TCM 内的密钥。即对于指定的密钥对象中的在 TCM 加载的句柄，通知 TCM 不再使用，可以释放资源。

接口定义：

```
TSM_RESULT Tspi_Key_UnloadKey
(
    TSM_HKEY hKey // in
);
```

输入参数描述：

——hKey 要卸载的密钥对象句柄。

输出参数描述：

——无

返回值：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
TCM_KEY_OWNER_CONTROL
```

### 5.5.10 获取密钥公钥

功能描述：

这个接口只对于 SM2 非对称密钥有效，对于已经加载的密钥通过密钥句柄获取密钥对象的公钥。

接口定义：

```
TSM_RESULT Tspi_Key_GetPubKey
(
    TSM_HKEY hKey,           // in
    UINT32* pulPubKeyLength, // out
    BYTE** prgbPubKey        // out
);
```

输入参数描述：

——hKey 密钥对象句柄。

输出参数描述：

——pulPubKeyLength 接收 prgbPubKey 参数的数据长度。(单位：字节)

——prgbPubKey 指向内存中 hKey 密钥对象的公钥。

返回值：

```
TSM_SUCCESS  
TSM_E_INVALID_HANDLE  
TSM_E_BAD_PARAMETER  
TSM_E_INTERNAL_ERROR
```

### 5.5.11 签署密钥

功能描述：

使用一个 SM2 非对称密钥对另一个 SM2 非对称密钥的公钥采用带 SM3 的签名算法来签名。

接口定义：

```
TSM_RESULT Tspi_Key_CertifyKey  
(  
    TSM_HKEY hKey,           //in  
    TSM_HKEY hCertifyingKey, //in  
    TSM_VALIDATION* pValidationData //in,out  
) ;
```

输入参数描述：

——hKey 密钥对象句柄,其公钥部分已被签名。  
——hCertifyingKey 用来对 hKey 进行签名的密钥的句柄。  
——pValidationData 是一个指向 TSM\_VALIDATION 结构的指针,其成员 rgbValidationData 包含着这个命令的签名数据。

输出参数描述：

——pValidationData 是一个指向 TSM\_VALIDATION 结构的指针。

返回值：

```
TSM_SUCCESS  
TSM_E_INVALID_HANDLE  
TSM_E_BAD_PARAMETER  
TSM_E_INTERNAL_ERROR
```

### 5.5.12 创建密钥

功能描述：

在 TCM 内部创建密钥并用 hWrappingKey 指向的密钥加密。

接口定义：

```
TSM_RESULT Tspi_Key_CreateKey  
(  
    TSM_HKEY hKey,           //in  
    TSM_HKEY hWrappingKey, //in  
    TSM_HPCR hPcrComposite //in,  
) ;
```

输入参数描述：

——hKey 要创建的密钥对象的句柄。  
——hWrappingKey 用来加密新创建的密钥的密钥句柄。  
——hPcrComposite 为 PCR 对象的句柄,如果此句柄的值不为 NULL,新创建的密钥将被绑定到由这个对象所描述的 PCR。

输出参数描述：

——无

返回值：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_KEY_NO_MIGRATION_POLICY
TSM_E_INTERNAL_ERROR
```

### 5.5.13 封装密钥

功能描述：

用 hWrappingKey 指向的密钥加密 hKey 指向的密钥。

如果 hWrappingKey 指向的密钥是对称密钥则需要发送到 TCM 内进行加密,如果为非对称则在 TSM 内进行加密。

接口定义：

```
TSM_RESULT Tspi_Key_WrapKey
(
    TSM_HKEY hKey,           //in
    TSM_HKEY hWrappingKey,   //in
    TSM_HPCR hPcrComposite  //in,
);
```

输入参数描述：

——hKey 要加密的密钥对象的句柄。

——hWrappingKey 用来加密 hKey 的密钥句柄。

——hPcrComposite 为 Tspi\_PcrComposite 的对象的句柄,如果此句柄的值非 NULL,新创建的密钥将会被绑定到由这个对象所描述的 PCR。

输出参数描述：

——无

返回值：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

### 5.5.14 创建迁移授权

功能描述：

此命令生成一个用于迁移密钥的授权数据,允许 TCM 所有者指定迁移策略,之后的密钥迁移无需 TCM 所有者参与。

接口定义：

```
TSM_RESULT Tspi_Key_AuthorizeMigrationKey
(
    TSM_HTCM hTCM,           //in
    TSM_HKEY hMigrationKey,   //in
```

```

    TSM_MIGRATE_SCHEME migrationScheme, //in
    UINT32* pulMigrationKeyAuthSize, //out
    BYTE** ppulMigrationKeyAuth, //out
);

```

输入参数描述：

——hTCM	对象句柄。
——hMigrationKey	密钥对象句柄，该密钥属于迁移的目标平台，其公钥用于保护被迁移密钥。
——migrationScheme	迁移方案标识，为 TSM_MS_MIGRATE 或者是 TSM_MS_REWRAP。

输出参数描述：

——pulMigrationKeyAuthSize	ppulMigrationKeyAuth 的数据长度。
——ppulMigrationKeyAuth	指向 MigrationKey 的授权数据。

返回值：

TSM_SUCCESS	
TSM_E_INVALID_HANDLE	
TSM_E_BAD_PARAMETER	
TSM_E_INTERNAL_ERROR	

### 5.5.15 创建迁移密钥数据块

功能描述：

生成待迁移的数据块。

接口定义：

```

TSM_RESULT Tspi_Key_CreateMigrationBlob
(
    TSM_HKEY hKeyToMigrate, //in
    TSM_HKEY hParentKey, //in
    UINT32 ulmigrationKeyAuthSize, //in
    BYTE* rgbmigrationKeyAuth, //in
    UINT32* pulMigratedDataSize, //out
    BYTE** prgbMigratedData, //out
    UINT32* pulEncSymKeySize, //out
    BYTE** prgbEncSymKey, //out
);

```

输入参数描述：

——hKeyToMigrate	指向被迁移的密钥句柄。
——hParentKey	hKeyToMigrate 指向的密钥的保护操作密钥句柄。
——ulmigrationKeyAuthSize	MigrationKeyAuth 的数据长度。
——rgbmigrationKeyAuth	MigrationKey 的授权数据。

输出参数描述：

——pulMigratedDataSize	prgbMigratedData 的数据长度。
——prgbMigratedData	在成功执行这个命令的情况下这个参数返回迁移数据。
——pulEncSymKeySize	指向被加密的对称密钥的数据长度的指针。
——prgbEncSymKey	指向对称密钥数据的指针，该对称密钥已被 Tspi_Key_AuthorizeMigrationKey() 所认证的公钥加密保护。

返回值：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_KEY_NO_MIGRATION_POLICY
TSM_E_INTERNAL_ERROR
```

### 5.5.16 导入迁移密钥数据块

功能描述：

将 Tspi\_Key\_CreateMigrationBlob 产生的迁移数据块转换成普通的用户密钥。

接口定义：

```
TSM_RESULT Tspi_Key_ConvertMigrationBlob
(
    TSM_HKEY hMEK,           //in
    TSM_HKEY hParentKey,     //in
    TSM_HKEY hKeyToMigrate,   //in
    UINT32 ulMigratedDataSize, //in
    BYTE* rgbMigratedData,   //in
    UINT32 ulEncSymKeySize,   //in
    BYTE* rgbEncSymKey,       //in
);
```

输入参数描述：

——hMEK	指向迁移时的保护密钥句柄,此为非对称密钥。
——hParentKey	给由 hKeyToMigrate 指向的密钥加密的保护操作密钥句柄。
——hKeyToMigrate	指向被迁移的密钥对象句柄。
——ulMigratedDataSize	由 rgbMigrationBlob 提供的迁移数据块数据的长度。
——rgbMigratedData	由上面的函数 Tspi_Key_CreateMigrationBlob() 返回的迁移数据块的数据。
——ulEncSymKeySize	已加密的对称密钥的长度。
——rgbEncSymKey	指向已加密的对称密钥的指针,IV 使用系统默认数值。

输出参数描述：

——无

返回值：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

## 5.6 数据加密与解密

该类用于将外部(例如用户,应用程序)产生的数据与系统关联起来(与平台或 PCR 绑定),或者用于为外部提供数据加密/解密服务。

进行授权处理时,可以给该类分配一个策略对象。

### 5.6.1 改变实体授权

功能描述：

改变数据对象的授权数据,为数据对象指派策略对象。

接口定义：

```
TSM_RESULT Tspi_ChangeAuth
(
    TSM_HOBJECT hObjectToChange,          //in
    TSM_HOBJECT hParentObject,           //in
    TSM_HPOLICY hNewPolicy             //in
);
```

输入参数描述：

——hObjectToChange 需要修改授权数据的密钥对象句柄。

——hParentObject 由 hObjectToChange 所指向的对象的保护操作密钥对象的句柄。

——hNewPolicy 更新的授权信息策略对象句柄。

输出参数描述：

——无。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INTERNAL_ERROR
```

### 5.6.2 获取策略对象

功能描述：

返回数据对象被指派的策略对象。

接口定义：

```
TSM_RESULT Tspi_GetPolicyObject
(
    TSM_HOBJECT hObject,                //in
    TSM_FLAG policyType,              //in
    TSM_HPOLICY* phPolicy            //out
);
```

输入参数描述：

——hObject 对象句柄。

——policyType 定义的策略类型。

输出参数描述：

——phPolicy 返回指派的策略对象。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

### 5.6.3 获取数据属性(整型参数)

功能描述：

获取数据对象的 32bit 属性。

接口定义：

```
TSM_RESULT Tspi_GetAttribUInt32
(
    TSM_HOBJECT hObject,           // in
    TSM_FLAG attribFlag,          // in
    TSM_FLAG subFlag,             // in
    UINT32* pulAttrib            // out
);
```

输入数据描述：

——hObject 需要获取属性的对象句柄。

——attribFlag 需要获取的属性标记(见属性说明表)。

——subFlag 需要获取的子属性标记(见属性说明表)。

属性说明表：

属性标记	属性子标识	数据描述
TSM_TSPATTRIB_ENCDATA_SY-MENC_MODE	TSM_TSPATTRIB_ENCDATA_SYMENC_MODE_ECB	加密采用 ECB 模式
	TSM_TSPATTRIB_ENCDATA_SYMENC_MODE_CBC	加密采用 CBC 模式

输出数据描述：

——pulAttrib 接收属性设置的值

返回值：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

### 5.6.4 设置数据属性(变长参数)

功能描述：

设置数据对象的非 32 比特属性。属性数据的结构和大小依赖于属性。

接口定义：

```
TSM_RESULT Tspi_SetAttribData
(
    TSM_HOBJECT hObject,           // in
    TSM_FLAG attribFlag,          // in
    TSM_FLAG subFlag,             // in
    UINT32 ulAttribDataSize,       // in
    BYTE* rgbAttribData          // in
);
```

输入参数描述：

- hObject 需要设置属性的对象句柄。
- attribFlag 指示要设置的属性的标记。(见属性说明表)
- subFlag 指示要设置的子属性的标记。(见属性说明表)

属性说明表：

属性标记	子属性标记	数据描述
TSM_TSPATTRIB_ENCDATA_BLOB	TSM_TSPATTRIB_ENCDATABLOB_BLOB	表示已加密的数据块。

- ulAttribDataSize prgbAttribData 参数的大小(以字节为单位)。
- rgbAttribData 此参数指向一个存放设置属性值的缓冲区。

返回值：

TSM\_SUCCESS  
TSM\_E\_INVALID\_HANDLE  
TSM\_E\_BAD\_PARAMETER  
TSM\_E\_INTERNAL\_ERROR

### 5.6.5 获取数据属性

功能描述：

本函数获取对象的非 32 比特属性。属性数据的结构和大小依属性而定。

接口定义：

```
TSM_RESULT Tspi_GetAttribData
(
    TSM_HOBJECT hObject,           //in
    TSM_FLAG attribFlag,          //in
    TSM_FLAG subFlag,             //in
    UINT32 * pulAttribDataSize,   //out
    BYTE ** prgbAttribData       //out
);
```

输入参数描述：

- hObject 需要获取属性的对象句柄。
- attribFlag 指示要获取的属性的标记。(见属性说明表)
- subFlag 指示要获取的子属性的标记。(见属性说明表)

属性说明表：

属性标记	子属性标记	数据描述
TSM_TSPATTRIB_ENCDATA_BLOB	TSM_TSPATTRIB_ENCDATABLOB_BLOB	已加密数据的数据块
TSM_TSPATTRIB_ENCDATA_PCR	TSM_TSPATTRIB_ENCDATAPCRONG_CREATION_SELECTION	封装(Sealing)时, 获取表征起作用的 PCR 位图
	TSM_TSPATTRIB_ENCDATAPCRONG_RELEASE_SELECTION	解封装时, 获取表征起作用的 PCR 位图

表 (续)

属性标记	子属性标记	数据描述
	TSM_TSPATTRIB_ENCDATAPCRONG_DIGEST_ATCREATION	封装时, 获取所选择的 PCR Composite 摘要
	TSM_TSPATTRIB_ENCDATAPCRONG_DIGEST_ATRELEASE	解封装时, 获取所选择的 PCR Composite 摘要
TSM_TSPATTRIB_ENCDATA_IV	NULL	采用 CBC 加密模式时的初始化向量

输出参数描述:

——pulAttribDataSize 返回的 prgbAttribData 参数的大小(以字节为单位)。

——prgbAttribData 命令成功返回,此参数指向一个存放指定属性值的缓冲区。

返回值:

TSM\_SUCCESS  
TSM\_E\_INVALID\_HANDLE  
TSM\_E\_INVALID\_ATTRIB\_FLAG  
TSM\_E\_INVALID\_ATTRIB\_SUBFLAG  
TSM\_E\_INVALID\_ATTRIB\_DATA

#### 5.6.6 数据加密

功能描述:

本函数对明文数据进行加密。

采用何种密码算法取决于密钥属性。

应用程序调用本命令时,如果采用 SM4 对称加密,则加密模式为 CBC 模式(参见密码算法要求的 SM4 使用模式);如果采用 SM2 非对称加密,传入的明文不得超过 512 字节。

接口定义:

```
TSM_RESULT Tspi_Data_Encrypt
(
    TSM_HENCDATA hEncData,          //in
    TSM_HKEY hEncKey,              //in
    TSM_BOOL bFinal,               //in
    TCM_SM4_IV* rgbDataIV,        //in
    BYTE* rgbDataToEncrypt,        //in
    UINT32 ulDataLength           //in
);
```

输入参数描述:

——hEncData	含有被加密数据的对象句柄。
——hEncKey	用于加密数据的密钥对象句柄。
——bFinal	用于标识是否是应用加密的明文最后一个分组。
——rgbDataIV	用于对称加密使用的 IV,固定 16 字节,可以为空。如果为空,则由 TSM 管理 IV,默认初始 IV 为 16 字节 0x00。

——rgbDataToEncrypt 明文数据。  
 ——ulDataLength 明文数据字节长度。

输出参数描述：

——无

返回参数：

TSM\_SUCCESS  
 TSM\_E\_INVALID\_HANDLE  
 TSM\_E\_BAD\_PARAMETER  
 TSM\_E\_ENC\_INVALID\_LENGTH  
 TSM\_E\_ENC\_NO\_DATA  
 TSM\_E\_INTERNAL\_ERROR

### 5.6.7 数据解密

功能描述：

本函数对密文数据进行解密。  
 采用何种算法取决于密钥属性。

注：本函数为明文数据分配了内存空间，调用者需显式释放该空间。

应用程序调用本命令时，如果采用 SM4 对称解密，则解密模式为 CBC 模式（参见密码算法要求的 SM4 使用模式）；如果采用 SM2 非对称加密解密，不超过最大密文数据。

接口定义：

```
TSM_RESULT Tspi_Data_Decrypt
(
    TSM_HENCDATA hEncData, //in
    TSM_HKEY hEncKey, //in
    TSM_BOOL bFinal, //in
    TCM_SM4_IV* rgbDataIV, //in
    UINT32* ulDataLength, //out
    BYTE** rgbDataDecrypted //out
);
```

输入参数描述：

——hEncData 含有被加密数据的对象句柄。  
 ——hEncKey 用于解密数据的密钥对象句柄。  
 ——bFinal 用于标识是否是应用解密的密文最后一个分组。  
 ——rgbDataIV 用于解密使用的 IV，固定 16 字节。

输出参数描述：

——ulDataLength 明文数据字节长度。  
 ——rgbDataEncrypted 一旦命令成功执行，该参数指向包含解密数据的存储区。

返回参数：

TSM\_SUCCESS  
 TSM\_E\_INVALID\_HANDLE  
 TSM\_E\_BAD\_PARAMETER  
 TSM\_E\_ENC\_INVALID\_LENGTH  
 TSM\_E\_ENC\_NO\_DATA  
 TSM\_E\_INTERNAL\_ERROR

### 5.6.8 数据封装

功能描述：

用来对一个数据块进行加密。如果要对该加密的数据进行解密，则必须使用同一平台的 Tspi\_Data\_Unseal 函数。

注：这里使用的加密密钥应是不可迁移密钥。

接口定义：

```
TSM_RESULT Tspi_Data_Seal
(
    TSM_HENCDATA hEncData,           //in
    TSM_HKEY hEncKey,               //in
    UINT32 ulDataLength,            //in
    BYTE* rgbDataToSeal,            //in
    TSM_HPCRS hPcrComposite       //in
);
```

输入参数描述：

——hEncData	数据对象句柄,当命令操作成功后,则该数据对象包含被加密数据。
——hEncKey	对数据进行加密的密钥对象句柄,该密钥是不可迁移的。
——ulDataLength	为参数 rgbDataToSeal 的数据长度(以字节为单位)。
——rgbDataToSeal	指向含有将要被加密的数据的内存块。
——hPcrComposite	PCR Composite 对象的句柄。

输出参数描述：

——无

返回值：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_ENC_INVALID_LENGTH
TSM_E_ENC_NO_DATA
TSM_E_ENC_INVALID_TYPE
TSM_E_INTERNAL_ERROR
```

### 5.6.9 数据解封

功能描述：

对指定的数据块进行解封。

接口定义：

```
TSM_RESULT Tspi_Data_Unseal
(
    TSM_HENCDATA hEncData,           //in
    TSM_HKEY hKey,                  //in
    UINT32* pulUnsealedDataLength,   //out
    BYTE** prgbUnsealedData        //out
);
```

输入参数描述:

- hEncData 包含有加密数据的数据对象的句柄。
- hKey 用来给数据进行解封且地址不可迁移的密钥对象句柄。

输出参数描述:

- pulUnsealedDataLength 指向参数 prgbUnsealedData 的数据长度的指针(以字节为单位)。
- prgbUnsealedData 如果操作成功,为指向包含明文信息缓冲区的指针。

返回值:

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_ENC_INVALID_LENGTH
TSM_E_ENC_NO_DATA
TSM_E_ENC_INVALID_TYPE
TSM_E_INTERNAL_ERROR
```

### 5.6.10 数字信封封装

功能描述:

本函数对明文数据进行数字信封加密封装。该数字信封操作仅用于密码支撑平台内部数据封装处理,其封装格式见附录 A。

加密后的数据通过 Tspi\_GetAttribData 来获取数字信封数据。

接口定义:

```
TSM_RESULT Tspi_Data_Envelop
(
    TSM_HENCDATA hEncData,           //in
    TSM_HKEY hEncKey,               //in
    BYTE* rgbDataToEncrypt,         //in
    UINT32 ulDataLength,            //in
);
```

输入参数描述:

- hEncData 含有被数字信封封装数据的对象句柄。
- hEncKey 用于加密数据的密钥对象句柄。
- rgbDataToEncrypt 明文数据。
- ulDataLength 明文数据字节长度。

输出参数描述:

- 无

返回参数:

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_ENC_INVALID_LENGTH
TSM_E_ENC_NO_DATA
TSM_E_INTERNAL_ERROR
```

### 5.6.11 数字信封解密

功能描述：

本函数对数字信封封装数据进行解密。

注：本函数为明文数据分配了内存空间，调用者需显式释放该空间。

接口定义：

```
TSM_RESULT Tspi_Data_Unenvelop
(
    TSM_HENCDATA hEncData,           // in
    TSM_HKEY hEncKey,              // in
    UINT32 * ulDataLength,         // out
    BYTE ** rgbDataDecrypted      // out
);
```

输入参数描述：

——hEncData 含有被数字信封封装数据的对象句柄。

——hEncKey 用于解密数据的密钥对象句柄。

输出参数描述：

——ulDataLength 明文数据字节长度。

——rgbDataEncrypted 一旦命令成功执行，该参数指向包含解密数据的存储区。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_ENC_INVALID_LENGTH
TSM_E_ENC_NO_DATA
TSM_E_INTERNAL_ERROR
```

## 5.7 PCR 管理

### 5.7.1 概述

PCR 操作类可用于建立系统平台的信任级别。该类提供了对 PCR 进行选择、读、写等操作的简便方法。

所有需要 PCR 信息的函数，在其参数表中都使用了这个类的对象句柄。

### 5.7.2 设置 PCR Locality 属性

功能描述：

本函数设置 PCRCComposite 对象中的 LocalityAtRelease 值。

接口定义：

```
TSM_RESULT Tspi_PcrComposite_SetPcrLocality
(
    TSM_HPCRS hPcrComposite,        // in
    UINT32 LocalityValue          // in
);
```

输入参数描述：

- hPcrComposite PCR composite 对象句柄。
- LocalityValue 要设置的 LocalityAtRelease 值。

输出参数描述：

- 无。

返回参数：

```
TSM_SUCCESS  
TSM_E_INVALID_HANDLE  
TSM_E_INVALID_OBJ_ACCESS  
TSM_E_BAD_PARAMETER  
TSM_E_INTERNAL_ERROR
```

### 5.7.3 获取 PCR Locality 属性

功能描述：

本函数从 PCRCComposite 对象中获取 LocalityAtRelease 值。

接口定义：

```
TSM_RESULT Tspi_PcrComposite_GetPcrLocality  
(  
    TSM_HPCRS hPcrComposite,           //in  
    UINT32* pLocalityValue           //out  
)
```

输入参数描述：

- hPcrComposite PCR composite 对象句柄。

输出参数描述：

- pLocalityValue 返回的 Locality 值。

返回参数：

```
TSM_SUCCESS  
TSM_E_INVALID_HANDLE  
TSM_E_INVALID_OBJ_ACCESS  
TSM_E_BAD_PARAMETER  
TSM_E_INTERNAL_ERROR
```

### 5.7.4 获取 PCR 摘要

功能描述：

本函数获取 PCRCComposite 对象中的 digestAtRelease 值。

接口定义：

```
TSM_RESULT Tspi_PcrComposite_GetCompositeHash  
(  
    TSM_HPCRS hPcrComposite,           //in  
    UINT32* pLen,                   //out  
    BYTE** ppbHashData             //out  
)
```

输入参数描述：

——hPcrComposite PCR composite 对象句柄，从中可返回杂凑摘要值。

输出参数描述：

——pLen 参数 ppbHashData 的字节长度。

——ppbHashData 创建或释放时的摘要数据。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_OBJ_ACCESS
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

### 5.7.5 设置 PCR 值

功能描述：

本函数对一个 PCRcomposite 对象中给定的 PCR 索引设置摘要值。

接口定义：

```
TSM_RESULT Tspi_PcrComposite_SetPcrValue
(
    TSM_HPCR hPcrComposite,           //in
    UINT32 ulPcrIndex,                //in
    UINT32 ulPcrValueLength,          //in
    BYTE* rgbPcrValue               //in
);
```

输入参数描述：

——hPcrComposite 要设置 PCR 值的 PCR composite 对象句柄。

——ulPcrIndex 要设置数据的 PCR 索引号。

——ulPcrValueLength 参数 rgbPcrValue 的字节长度。

——rgbPcrValue 要设置的数据。

输出参数描述：

——无。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

### 5.7.6 获取 PCR 值

功能描述：

返回在 PCR composite 对象中指定 PCR 索引的摘要值。

接口定义：

```
TSM_RESULT Tspi_PcrComposite_GetPcrValue
(
    TSM_HPCR hPcrComposite,           //in
    UINT32 ulPcrIndex,                //in
```

```

    UINT32* pulPcrValueLength,      //out
    BYTE** prgbPcrValue          //out
);

```

输入参数描述：

——hPcrComposite 要从中返回 PCR 值的 PCR composite 对象句柄。  
 ——ulPcrIndex 要读取数据的 PCR 索引。

输出参数描述：

——pulPcrValueLength 参数 prgbPcrValue 的字节长度。  
 ——prgbPcrValue 指向返回的由 ulPcrIndex 参数指示的 PCR 值。

返回参数：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR

```

### 5.7.7 选择 PCR 索引

功能描述：

本函数在 PCR composite 对象中选择一个 PCR 索引。该函数在创建和释放 PCR composite 对象时使用。PCR composite 对象必须由 Tspi\_Context\_CreateObject 创建。例如：在调用 Tspi\_TCM\_Quote 之前，需用本函数来选择 PCR 寄存器。

接口定义：

```

TSM_RESULT Tspi_PcrComposite_SelectPcrIndex
(
    TSM_HPCRS hPcrComposite,      //in
    UINT32 ulPcrIndex,           //in
    UINT32 Direction            //in
);

```

输入参数描述：

——hPcrComposite 要从中返回 PCR 索引的 PCR composite 对象句柄。  
 ——ulPcrIndex 要选择的 PCR 索引号。  
 ——Direction 指明选择的 PCR 是 Creation 还是 Release。

输出参数描述：

——无。

返回参数：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR

```

### 5.7.8 非易失性存储管理

NV 存储管理类用于在 TCM 的非易失性存储区域中存储属性信息，用于 NV 区域的定义、读、写和释放。

该类建立 NV 存储区的大小、索引、各种读写授权，其中授权可以基于 PCR 值或者授权数据。

### 5.7.9 设置非易失性存储区属性(整型参数)

功能描述：

本函数设置 NV 存储对象的 32bit 属性。

接口定义：

```
TSM_RESULT Tspi_SetAttribUint32
(
    TSM_HOBJECT hObject,           // in
    TSM_FLAG attribFlag,          // in
    TSM_FLAG subFlag,             // in
    UINT32 ulAttrib              // in
);
```

输入参数描述：

- hObject 需要设置属性的对象句柄。
- attribFlag 需要设置的属性标记。
- subFlag 需要设置的子属性标记。
- ulAttrib 属性设置的值。

属性定义：

属性	子属性	属性值	描述
TSM_TSPATTRIB_NV_INDEX	0	UINT32	与此对象关联的 NV 存储区的索引
TSM_TSPATTRIB_NV_PERMISSIONS	0	UINT32	权限值
TSM_TSPATTRIB_NV_DATASIZE	0	UINT32	定义的 NV 存储区的大小

输出参数描述：

- 无。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
TSM_E_INVALID_ATTRIB_SUBFLAG
TSM_E_INVALID_ATTRIB_DATA
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

### 5.7.10 获取非易失性存储区属性(整型参数)

功能描述：

本函数获取 NV 存储对象的 32 比特属性。

接口定义：

```
TSM_RESULT Tspi_GetAttribUint32
```

```

(
    TSM_HOBJECT hObject,           // in
    TSM_FLAG attribFlag,          // in
    TSM_FLAG subFlag,             // in
    UINT32* pulAttrib           // out
);

```

输入参数描述：

——hObject 需要查询属性的对象句柄。

——attribFlag 需要查询的属性。

——subFlag 需要查询的子属性。

属性定义：

属性	子属性	属性值	描述
TSM_TSPATTRIB_NV_IN_DEX	0	UINT32	与此对象相关联的 NV 存储区域的索引号
TSM_TSPATTRIB_NV_PERMISSIONS	0	UINT32	权限许可的值
TSM_TSPATTRIB_NV_DATASIZE	0	UINT32	定义的 NV 存储区域大小
TSM_TSPATTRIB_NV_STATE	TSM_TSPATTRIB_NVSTATE_READSTCLEAR	Boolean	每次启动时设置为 FALSE，在 datasize 为 0 的 ReadValueXXX 之后，设置为 TRUE。
	TSM_TSPATTRIB_NVSTATE_WRITESTCLEAR	Boolean	每次启动时设置为 FALSE，在 datasize 为 0 的 WriteValueXXX 之后，设置为 TRUE。
	TSM_TSPATTRIB_NVSTATE_WRIEDEFINE	Boolean	在定义 NV 空间之后设置为 FALSE，在 datasize 为 0 的 WriteValue 的成功调用之后，设置为 TRUE。
TSM_TSPATTRIB_NV_PCR	TSM_TSPATTRIB_NVPCR_READLOCALITYATRELEASE	BYTE	Locality 掩码，用于 NV 区域的 PCR 读取限制。
	TSM_TSPATTRIB_NVPCR_WRITELOCALITYATRELEASE	BYTE	Locality 掩码，用于 NV 区域的 PCR 写限制。

输出参数描述：

——pulAttrib 指向查询到的属性值。

返回参数：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
TSM_E_INVALID_ATTRIB_SUBFLAG
TSM_E_INVALID_ATTRIB_DATA

```

TSM\_E\_BAD\_PARAMETER  
TSM\_E\_INTERNAL\_ERROR

### 5.7.11 获取非易失性存储区属性(变长参数)

功能描述：

获取 NV 存储对象的非 32 比特属性。属性数据的结构和大小依赖于属性。

接口定义：

```
TSM_RESULT Tspi_GetAttribData
(
    TSM_HOBJECT hObject,           //in
    TSM_FLAG attribFlag,          //in
    TSM_FLAG subFlag,             //in
    UINT32* pulAttribDataSize,    //out
    BYTE** prgbAttribData        //out
);
```

输入参数描述：

——hObject 需要获取属性的对象句柄。

——attribFlag 需要获取的属性标识。

——subFlag 需要获取的子属性标识。

输出参数描述：

——pulAttribDataSize 取得的 rgbAttribData 参数的大小(以字节为单位)。若参数 rgbAttribData 为一个 TSM\_UNICODE 字符串，则该大小还包括结束符 NULL。

——prgbAttribData 此参数指向一个存放获取的属性值的缓冲区。

属性定义：

属性	子属性	属性值
TSM_TSPATTRIB_NV_PCR	TSM_TSPATTRIB_NVPCR_RE ADPCRSELECTION	PCR 选择掩码, 用于 NV 区域的 PCR 读取限制。
	TSM_TSPATTRIB_NVPCR_RE ADDIGESTATRELEASE	DigestAtRelease 子属性, 用于 NV 区域 PCRread 限制。
	TSM_TSPATTRIB_NVPCR_W RITEPCRSELECTION	PCR 选择掩码, 用于 NV 区域的 PCR 读取限制。
	TSM_TSPATTRIB_NVPCR_W RITEDIGESTATRELEASE	DigestAtRelease 子属性, 用于 NV 区域 PCRwrite 限制。

返回参数：

TSM\_SUCCESS  
TSM\_E\_INVALID\_HANDLE  
TSM\_E\_INVALID\_ATTRIB\_FLAG  
TSM\_E\_INVALID\_ATTRIB\_SUBFLAG  
TSM\_E\_INVALID\_ATTRIB\_DATA  
TSM\_E\_BAD\_PARAMETER  
TSM\_E\_INTERNAL\_ERROR

### 5.7.12 创建非易失性存储区空间

功能描述：

创建指定的非易失性存储空间。

接口定义：

```
TSM_RESULT Tspi_NV_DefineSpace
(
    TSM_HNVSTORE      hNVStore,           //in
    TSM_HPCRS         hReadPcrComposite, //in,可以为 NULL
    TSM_HPCRS         hWritePcrComposite //in,可以为 NULL
);
```

输入参数描述：

——hNVStore	非易失性存储对象句柄
——hReadPcrComposite	PCR 对象句柄。当这个参数为 NULL 时,没有 PCR 数据与指定的非易失性存储空间绑定;当不为 NULL 时,新创建的非易失性存储区需要这个句柄对象描述的 PCR 内容来成功的执行读操作。
——hWritePcrComposite	PCR 对象句柄。当这个参数为 NULL 时,没有 PCR 数据与指定的非易失性存储空间绑定;当不为 NULL 时,新创建的非易失性存储区需要这个句柄对象描述的 PCR 内容来成功的执行写操作。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
TSM_E_NV_AREA_NOT_EXIST
TCM_BAD_INDEX
TCM_AUTH_CONFLICT
TCM_AUTHFAIL
TCM_OWNERSET
TCM_BAD_DATASIZE
TCM_MAXNVWRITE
TCM_INVALID_STRUCTURE
TCM_NOWRITE
```

注：本函数用于定义非易失性存储空间。注意,本函数与 Tspi\_NV\_ReleaseSpace 调用同一个 TCM 功能命令。当两次调用这个 TCM 功能命令时,会删除定义的非易失性存储空间。所以必须检查这个定义的非易失性存储空间是否已经定义,如果已经定义,必须返回 TSM\_E\_NV\_AREA\_EXIST 错误码。

这个命令执行需要所有者授权。

### 5.7.13 释放非易失性存储区空间

功能描述：

释放指定的非易失性存储空间。

接口定义：

```
TSM_RESULT Tspi_NV_ReleaseSpace
```

```

(
    TSM_HNVSTORE      hNVStore      //in
);

```

输入参数描述：

——hNVStore 非易失性存储对象句柄。

返回参数：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
TSM_E_NV_AREA_NOT_EXIST
TCM_AREA_LOCKED

```

注：本函数用于释放非易失性存储空间。注意，本函数与 Tspi\_NV\_DefineSpace 调用同一个 TCM 功能命令。所以必须检查这个定义的非易失性存储空间是否已经定义，如果没有定义，必须返回 TSM\_E\_NV\_AREA\_NOT\_EXIST 错误码。

这个命令执行需要所有者授权。

#### 5.7.14 数据写入非易失性存储区

功能描述：

将数据写入指定的非易失性存储区域。

接口定义：

```

TSM_RESULT Tspi_NV_WriteValue
(
    TSM_HNVSTORE      hNVStore,      //in
    UINT32            offset,        //in
    UINT32            ulDataLength,   //in
    BYTE*             rgbDataToWrite //in
);

```

输入参数描述：

——hNVStore 非易失性存储对象句柄。

——offset 偏移量，即在 NV 区域中的写入起始地址。

——ulDataLength rgbDataToWrite 的长度。即需要写到非易失性存储区域中的长度。

——rgbDataToWrite 指向需要写入数据的缓冲区。

返回参数：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
TCM_BAD_INDEX
TCM_MAXNVWRITE
TCM_AUTH_CONFLICT
TCM_AUTHFAIL
TCM_AREA_LOCKED

```

```

TCM_BAD_LOCALITY
TCM_BAD_PRESENCE
TCM_DISABLED_CMD
TCM_NOSPACE
TCM_NOT_FULLWRITE
TCM_WRONGPCRVALUE

```

**注：**如果有一个策略对象分配给这个对象，策略对象中的授权数据被用于这个操作的授权。如果没有策略对象分配给这个对象，执行一个非授权写操作。

### 5.7.15 从非易失性存储区读取数据

**功能描述：**

从指定的非易失性存储区域中读取数据。

**接口定义：**

```

TSM_RESULT Tspi_NV_ReadValue
(
    TSM_HNVSTORE      hNVStore,           // in
    UINT32             offset,            // in
    UINT32*            pulDataLength,     // in,out
    BYTE**             prgbDataRead,      // out
);

```

**输入参数描述：**

——hNVStore	非易失性存储对象句柄。
——offset	需要读取数据在非易失性存储区域中的偏移值。
——pulDataLength	输入的 prgbDataRead 的缓冲区长度。

**输出参数描述：**

——pulDataLength	输出的 prgbDataRead 的长度。
——prgbDataRead	指向返回读取数据的缓冲区。

**返回参数：**

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
TCM_BAD_INDEX
TCM_AUTH_CONFLICT
TCM_AUTHFAIL
TCM_BAD_LOCALITY
TCM_BAD_PRESENCE
TCM_DISABLED_CMD
TCM_NOSPACE
TCM_WRONGPCRVALUE

```

**注：**如果有一个策略对象分配给这个对象，策略对象中的授权数据被用于这个操作的授权。如果没有策略对象分配给这个对象，执行一个非授权读操作。

## 5.8 杂凑操作

### 5.8.1 设置杂凑对象属性(整型参数)

功能描述：

设置杂凑对象的 32 比特属性。

接口定义：

```
TSM_RESULT Tspi_SetAttribUInt32
(
    TSM_HOBJECT    hHash,           // in
    TSM_FLAG       attribFlag,     // in
    TSM_FLAG       subFlag,        // in
    UINT32         ulAttrib       // in
);
```

输入参数描述：

——hHash 杂凑对象句柄。

——attribFlag 声明需要设置的属性标记。

——subFlag 声明需要设置的子属性标记。

——ulAttrib 设置的属性值。

属性说明表：

属性标记	子属性标记	值	描述
TSM_TSPATTRIB_HASH_SCHEME	0	TSM_TSPATTRIB_HASHSCHEME_NORMAL(默认)	使用普通的杂凑方法进行杂凑操作。设置这个属性后,仅能执行 Tspi_Hash_UpdateHashValue,可以执行多次。
	0	TSM_TSPATTRIB_HASHSCHEME_GB	使用普通的杂凑方法进行杂凑操作。设置这个属性后,仅能执行 Tspi_Hash_SetUserMessageData,可以执行多次,但会破坏以前执行的结果。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
TSM_E_INVALID_ATTRIB_SUBFLAG
TSM_E_INVALID_ATTRIB_DATA
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

注：当 Tspi\_Hash\_SetUserMessageData、Tspi\_Hash\_SetUserMessageData、Tspi\_Hash\_SetHashValue、Tspi\_Hash\_GetHashValue、Tspi\_Hash\_Sign、Tspi\_Hash\_VerifySignature 和 Tspi\_Hash\_TickStampBlob 调用后,不能使用 Tspi\_SetAttribUInt32(TSM\_TSPATTRIB\_HASH\_SCHEME)再次更改属性。

### 5.8.2 获取杂凑对象属性(整型参数)

功能描述：

获取杂凑对象的 32 比特属性。

接口定义：

```
TSM_RESULT Tspi_GetAttribUInt32
(
    TSM_HOBJECT      hHash,          // in
    TSM_FLAG         attribFlag,     // in
    TSM_FLAG         subFlag,        // in
    UINT32*          pulAttrib      // out
);
```

输入参数描述：

——hHash 杂凑对象句柄。

——attribFlag 声明需要设置的属性标记。

——subFlag 声明需要设置的子属性标记。

属性说明表：

属性标记	子属性标记	值	描述
TSM_TSPATTRIB_HASH_SCHEME	0	TSM_TSPATTRIB_HASH_SCHEME_NORM_AL(默认)	使用普通的杂凑方法进行杂凑操作。仅能执行 Tspi_Hash_UpdateHashValue 函数,可以执行多次。
	0	TSM_TSPATTRIB_HASH_SCHEME_GB	使用普通的杂凑方法进行杂凑操作。仅能执行 Tspi_Hash_SetUserMessageData 函数,可以执行多次,但会破坏以前执行的结果。

输出参数描述：

——pulAttrib 获取的属性值。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
TSM_E_INVALID_ATTRIB_SUBFLAG
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
```

### 5.8.3 设置杂凑对象属性(变长参数)

功能描述：

设置数据对象的非 32 比特属性。属性数据的结构和大小依赖于属性。

接口定义：

```
TSM_RESULT Tspi_SetAttribData
(

```

```

TSM_HOBJECT    hHash,           //in
TSM_FLAG       attribFlag,     //in
TSM_FLAG       subFlag,        //in
UINT32         ulAttribDataSize, //in
BYTE*          rgbAttribData   //in
);

```

输入参数描述：

- hHash 需要设置属性的 Hash 对象句柄。
- attribFlag 声明需要设置的属性标记。
- subFlag 声明需要设置的子属性标记。

属性说明表：

属性标记	子属性标记	数据描述
TSM_TSPATTRIB_ALG_IDENTIFIER	0	设置密码杂凑算法标识

- ulAttribDataSize rgbAttribData 的长度
- rgbAttribData 需设置的属性值(SM3 的算法标识符)。

返回参数：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INVALID_ATTRIB_FLAG
TSM_E_INVALID_ATTRIB_SUBFLAG
TSM_E_INVALID_ATTRIB_DATA
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR

```

#### 5.8.4 对用户数据进行杂凑操作

功能描述：

对输入的用户信息，消息以及相关的公钥信息按照签名过程中定义的密码杂凑算法进行计算。

接口定义：

```

TSM_RESULT Tspi_Hash_SetUserMessageData
(
    TSM_HHASH    hHash,           //in
    TSM_HKEY     hKey,            //in
    UINT32       ulUserIDSize,   //in
    BYTE*        rgbUserID,      //in
    UINT32       ulMessageSize,  //in
    BYTE*        rgbMessage,     //in
);

```

输入参数描述：

- hHash 杂凑对象句柄。
- hKey 公钥的密钥句柄。
- ulUserIDSize 输入的用户信息的长度。
- rgbUserID 输入的用户信息。

——ulMessageSize 输入的消息长度。

——rgbMessage 输入的消息。

返回参数：

TSM\_SUCCESS

TSM\_E\_INVALID\_HANDLE

TSM\_E\_BAD\_PARAMETER

TSM\_E\_HASH\_INVALID\_LENGTH

TSM\_E\_HASH\_NO\_DATA

TSM\_E\_INTERNAL\_ERROR

**注：**仅当使用了 Tspi\_SetAttribUInt32 设置 TSM\_TSPATTRIB\_HASHSCHEME\_GB 属性后，才能使用 Tspi\_Hash\_SetUserMessageData 方法对用户信息与消息进行杂凑运算。当使用了 Tspi\_Hash\_SetUserMessageData 方法后，不能再使用 Tspi\_SetAttribUInt32 设置 TSM\_TSPATTRIB\_HASHSCHEME\_NORMAL 属性，也不能使用 Tspi\_Hash\_UpdateHashValue 方法。可以使用 Tspi\_Hash\_GetHashValue、Tspi\_Hash\_Sign、Tspi\_Hash\_verifySignature 和 Tspi\_Hash\_TickStampBlob 等函数。

### 5.8.5 设置杂凑值

功能描述：

本函数设置杂凑类的 HASH 值。如果杂凑对象的标志位为 TSM\_HASH\_OTHER，那么就要调用 Tspi\_SetAttribData 设置密码杂凑算法。

接口定义：

```
TSM_RESULT Tspi_Hash_SetHashValue
(
    TSM_HHASH hHash,           // in
    UINT32 ulHashValueLength,  // in
    BYTE* rgbHashValue         // in
);
```

输入参数描述：

——hHash 杂凑对象句柄。

——ulHashValueLength 参数 rgbHashValue 的长度(以字节为单位)。

——rgbHashValue 指向 Hash 值存储空间。

输出参数描述：

——无

返回参数

TSM\_SUCCESS

TSM\_E\_INVALID\_HANDLE

TSM\_E\_BAD\_PARAMETER

TSM\_E\_HASH\_INVALID\_LENGTH

TSM\_E\_HASH\_NO\_DATA

TSM\_E\_INTERNAL\_ERROR

### 5.8.6 获取杂凑值

功能描述：

本函数返回杂凑类的杂凑值。

接口定义：

```
TSM_RESULT Tspi_Hash_GetHashValue
(
    TSM_HHASH hHash,           //in
    UINT32* pulHashValueLength, //out
    BYTE** prgbHashValue      //out
);
```

输入参数描述：

——hHash 杂凑对象句柄

输出参数描述：

——pulHashValueLength 参数 prgbSignature. 的长度(以字节为单位)。

——prgbSignature 指向杂凑值存储空间。

返回参数

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_HASH_INVALID_LENGTH
TSM_E_HASH_NO_DATA
TSM_E_INTERNAL_ERROR
```

### 5.8.7 更新杂凑值

功能描述：

本函数更新杂凑对象的杂凑值。

接口定义：

```
TSM_RESULT Tspi_Hash_UpdateHashValue
(
    TSM_HHASH hHash,           //in
    UINT32 ulDataLength,       //in
    BYTE* rgbdData            //in
);
```

输入参数描述：

——hHash 待更新的杂凑对象句柄。

——ulDataLength 参数 rgbdData 的数据长度(以字节为单位)。

——rgbdData. 指向要被更新的数据。

输出参数描述：

——无

返回参数

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_HASH_INVALID_LENGTH
TSM_E_HASH_NO_DATA
TSM_E_INTERNAL_ERROR
```

### 5.8.8 对杂凑值签名

功能描述：

对杂凑后的结果使用签名密钥进行签名。

接口定义：

```
TSM_RESULT Tspi_Hash_Sign
(
    TSM_HHASH     hHash,           //in
    TSM_HKEY      hKey,           //in
    UINT32*       pulSignatureLength, //out
    BYTE**        prgbSignature   //out
);
```

输入参数描述：

——hHash 需要签名的杂凑对象句柄。

——hKey 用于签名的密钥句柄。

输出参数描述：

——pulSignatureLength 如果操作成功，为返回的签名数据的长度。

——prgbSignature 如果操作成功，为返回的签名数据。

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_HASH_INVALID_LENGTH
TSM_E_HASH_NO_DATA
TSM_E_HASH_NO_IDENTIFIER
TSM_E_INTERNAL_ERROR
```

注：签名的数据必须被设置在一个杂凑对象的实例中。使用 Tspi\_Hash\_SetHashValue()、Tspi\_Hash\_UpdateHash()或 Tspi\_Hash\_SetUserMessageData()可以设置或计算杂凑数据。Tspi\_Hash\_Sign 方法为 prgbSignature 分配内存。释放这个内存必须使用 Tspi\_Context\_FreeMemory 方法。

### 5.8.9 验证杂凑值签名

功能描述：

本函数用于验证一个杂凑类的签名。

接口定义：

```
TSM_RESULT Tspi_Hash_VerifySignature
(
    TSM_HHASH hHash,           //in
    TSM_HKEY hKey,            //in
    UINT32 ulSignatureLength, //in
    BYTE* rgbSignature        //in
);
```

输入参数描述：

——hHash	待验证杂凑对象句柄。
——hKey	签名密钥句柄。
——ulSignatureLength	签名数据 rgbSignature 长度。
——rgbSignature	指向签名数据。

输出参数描述：

——无

返回参数

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_HASH_INVALID_LENGTH
TSM_E_HASH_NO_DATA
TSM_E_INVALID_SIGSCHEME
TSM_E_INTERNAL_ERROR

### 5.8.10 给杂凑类加时间戳

功能描述：

该方法用于给一个杂凑类加一个时间戳。它将一个时钟计数(tickvalue)与一个 blob 相关联，用以标识该 blob 在 tickvalue 对应的时间之前就已存在。

接口定义：

```
TSM_RESULT Tspi_Hash_TickStampBlob
(
    TSM_HHASH hHash,                      //in
    TSM_HKEY hIdentKey,                   //in
    TSM_VALIDATION* pValidationData     //in
);
```

输入参数描述：

——hHash	指向 20 字节的杂凑对象句柄。
——hIdentKey	用于执行时间戳的身份密钥。
——pValidationData	指向用来验证签名的数据。

输出参数描述：

——无

返回参数

TSM_SUCCESS
TSM_E_INVALID_TCM_HANDLE
TSM_E_INVALID_KEY_HANDLE
TSM_E_INTERNAL_ERROR

## 5.9 密钥协商

### 5.9.1 创建会话

功能描述：

与 TCM 创建一个密钥协商会话句柄，并返回一个临时的曲线上的点。

密钥协商双方 A 与 B 使用这个函数生成临时点分别为 Ra,Rb。用户 A 将 Ra 传送给用户 B, 用户 B 将 Rb 传送给用户 A, 再使用 Tspi\_Exchange\_GetKeyExchange 进行密钥协商计算。

当已经使用 Tspi\_Exchange\_CreateKeyExchange 创建了密钥协商句柄, 再使用 Tspi\_Exchange\_CreateKeyExchange 函数时, 返回 TSM\_E\_EXCHANGE\_HANDLE\_EXIST 错误码, 必须使用 Tspi\_Exchange\_ReleaseExchangeSession 释放这个密钥协商句柄后, 可以再次使用 Tspi\_Exchange\_CreateKeyExchange 与 TCM 建立密钥协商句柄。

接口定义:

```
TSM_RESULT Tspi_Exchange_CreateKeyExchange
(
    TSM_HEXCHANGE    hKeyExchange,           // in
    UINT32*          pcRxSize,              // out
    BYTE**           prgbRxPoint          // out
);
```

输入参数描述:

——hKeyExchange 密钥协商句柄。

输出参数描述:

——pcRxSize 输出 prgbRxPoint 的长度。

——prgbRxPoint 返回一个临时的 SM2 曲线上的点, 按照国标密码算法规定的未压缩编码形式的字符串。

返回参数:

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
TSM_E_EXCHANGE_HANDLE_EXIST
```

## 5.9.2 获取会话密钥

功能描述:

与 TCM 创建一个密钥协商会话句柄, 并返回一个临时的曲线上的点。一次协商会话只能做一次密钥交换。

如果并没有使用 Tspi\_Exchange\_CreateKeyExchange 创建密钥协商句柄, 本函数必须返回返回 TSM\_E\_EXCHANGE\_HANDLE\_NOT\_EXIST 错误码。

在操作成功后, hSessionKey 句柄表示成功协商之后的对称密钥句柄。

假定用户 A 与用户 B 进行密钥协商, 当双方使用 Tspi\_Exchange\_GetKeyExchange 函数操作成功后, 用户 A 的本地校验数据为 Sa, 发给对方的校验数据为 S1; 用户 B 的本地校验数据为 Sb, 发给对方的校验数据为 S2。用户 A 需要比较 Sa 与 S2 是否相等; 用户 B 需要比较 Sba 与 S1 是否相等。如果验证失败, 双方使用生成的对称密钥进行加解密会失败。

接口定义:

```
TSM_RESULT Tspi_Exchange_GetKeyExchange
(
    TSM_HEXCHANGE    hKeyExchange,           // in
    ...
```

```

    TSM_HKEY           hPermanentKey,      // in
    TSM_EXCHANGE_TAG   cExchangeTag       // in
    UINT32              cPointSize,        // in
    BYTE*               rgbPoint,         // in
    UINT32              cRaSize,          // in
    BYTE*               rgbRa,            // in
    UINT32              cRbSize,          // in
    BYTE*               rgbRb,            // in
    UINT32              cRxSize,          // in
    BYTE*               rgbRx,            // in
    TSM_HKEY*           phKey,            // in,out
    UINT32*             pcSxSize,         // out
    BYTE**              prgbSxData,       // out
    UINT32*             pcSySize,         // out
    BYTE**              prgbSyData,       // out
);

```

输入参数描述：

——hKeyExchange	密钥协商句柄。
——hPermanentKey	已经加载到 TCM 的本地静态密钥句柄。
——cExchangeTag	协商标识为密钥协商的身份标识,1 代表发起方,2 代表响应方。
——hKey	产生的密钥结构属性,用来作为生成密钥的的存储结构,不与 PCR 相绑定。
——cPointSize	对方静态密钥公钥信息长度。
——rgbPoint	对方静态密钥公钥信息。
——cRaSize	本地个人信息长度。
——rgbRa	本地个人信息。
——cRbSize	对方个人信息长度。
——rgbRb	对方个人信息。
——cRxSize	对方临时密钥公钥信息长度。
——rgbRx	对方临时密钥公钥信息。

输出参数描述：

——hKey	协商的共享密钥。
——pcSxSize	用于本地验证协商过程的数据长度。
——prgbSxData	用于本地验证协商过程的数据。
——pcSySize	提供给对方进行验证过程的数据长度。
——prgbSyData	提供给对方进行验证过程的数据。

返回参数：

```

TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_BAD_PARAMETER
TSM_E_INTERNAL_ERROR
TSM_E_EXCHANGE_HANDLE_NOT_EXIST

```

### 5.9.3 释放会话

功能描述：

释放与 TCM 建立的密钥协商会话句柄。

如果并没有使用 Tspi\_Exchange\_CreateKeyExchange 创建密钥协商句柄，本函数必须返回 TSM\_E\_EXCHANGE\_HANDLE\_NOT\_EXIST 错误码。

接口定义：

```
TSM_RESULT Tspi_Exchange_ReleaseExchangeSession
(
    TSM_HEXCHANGE hKeyExchange,           //in
);
```

输入参数描述：

——hKeyExchange 密钥协商句柄

返回参数：

```
TSM_SUCCESS
TSM_E_INVALID_HANDLE
TSM_E_INTERNAL_ERROR
TSM_E_EXCHANGE_HANDLE_NOT_EXIST
```

附录 A  
(规范性附录)  
接口规范数据结构

## A. 1 基础定义

### A. 1. 1 数据类型

#### A. 1. 1. 1 指针和句柄长度

指针和句柄长度均为 32 位。

#### A. 1. 1. 2 基本类型

类型	定义
UINT32	无符号 32 位整型
BYTE	无符号字符型
TSM_BOOL	带符号字符型
TSM_UNICODE	看作 16 比特的序列(比特串)
PVOID	32 位指针

#### A. 1. 1. 3 布尔类型

名称	值	描述
TRUE	0x01	真
FALSE	0x00	假

#### A. 1. 1. 4 导出类型

类型	定义	描述
TSM_FLAG	UINT32	对象属性
TSM_HOBJECT	UINT32	基本对象句柄
TSM_ALGORITHM_ID	UINT32	TSM 算法标
TSM_MIGRATE_SCHEME	UINT32	TSM 迁移方案标识
TSM_KEY_USAGE_ID	UINT32	TSM 密钥使用类型标识
TSM_KEY_ENC_SCHEME	UINT32	TSM 加密方案标识
TSM_KEY_SIG_SCHEME	UINT32	TSM 签名方案标识
TSM_EVENTTYPE	UINT32	TSM 事件类型
TSM_COUNTER_ID	UINT32	计数器标识
TSM_RESULT	UINT32	TSM 接口命令结果

### A.1.1.5 对象类型

类型	定义	描述
TSM_HCONTEXT	TSM_HOBJECT	上下文对象句柄
TSM_HPOLICY	TSM_HOBJECT	安全策略对象句柄
TSM_HTCM	TSM_HOBJECT	TCM 对象句柄
TSM_HKEY	TSM_HOBJECT	密钥对象句柄
TSM_HENCDATA	TSM_HOBJECT	加密数据对象句柄
TSM_HPCRS	TSM_HOBJECT	PCR 组合对象句柄
TSM_HHASH	TSM_HOBJECT	杂凑对象句柄
TSM_HNVSTORE	TSM_HOBJECT	NV 数据对象句柄
TSM_HMIGDATA	TSM_HOBJECT	迁移数据处理对象句柄
TSM_HEXCHANGE	TSM_HOBJECT	密钥协商对象句柄

### A.1.2 定义的常量

#### A.1.2.1 对象类型定义

对象类型的定义与 Tspi\_Context\_CreateObject 方法一起使用。定义的对象类型基于数据类型 TSM\_FLAG。

对象类型	描述
TSM_OBJECT_TYPE_POLICY	策略对象
TSM_OBJECT_TYPE_KEY	密钥对象(包括对称与非对称)
TSM_OBJECT_TYPE_ENCDATA	加密数据对象;限定使用范围的数据、密封数据或信封封装数据
TSM_OBJECT_TYPE_PCRS	PCR 对象
TSM_OBJECT_TYPE_HASH	杂凑对象
TSM_OBJECT_TYPE_NV	非易失性存储对象
TSM_OBJECT_TYPE_MIGDATA	迁移数据处理对象
TSM_OBJECT_TYPE_EXCHANGE	密钥协商对象

#### A.1.2.2 对象初始化定义

对象初始化标记的定义与 Tspi\_Context\_CreateObject 方法一起使用。定义初始化标记基于数据类型 TSM\_FLAG。

初始化标记	描述
TSM_KEY_SIZE_DEFAULT_SYM	缺省的对称密钥长度
TSM_KEY_SIZE_DEFAULT_ASY	缺省的非对称密钥长度

表 (续)

初始化标记	描述
TSM_KEY_SIZE_128	SM4 的密钥长度为 128-bit
TSM_KEY_SIZE_256	SM2 的私钥长度为 256-bit
TSM_KEY_SIZE_512	SM2 的公钥长度为 512-bit
TSM_SM2KEY_TYPE_STORAGE	SM2 存储加密密钥
TSM_SM2KEY_TYPE_SIGNING	SM2 签名密钥
TSM_SM2KEY_TYPE_BIND	SM2 加密密钥
TSM_SM2KEY_TYPE_IDENTITY	SM2 身份标识密钥
TSM_SM2KEY_TYPE_AUTHCHANGE	临时性 SM2 密钥, 用于改变授权数据值
TSM_SM2KEY_TYPE_MIGRATE	迁移保护密钥
TSM_SM4KEY_TYPE_STORAGE	SM4 存储加密密钥
TSM_SM4KEY_TYPE_BIND	SM4 加密密钥
TSM_KEY_NON_VOLATILE	非易失性密钥, 启动时可以不加载
TSM_KEY_VOLATILE	易失性密钥, 启动时必须加载
TSM_KEY_NOT_MIGRATABLE	不可迁移密钥(缺省属性)
TSM_KEY_MIGRATABLE	可迁移的密钥
TSM_KEY_NO_AUTHORIZATION	无需授权的密钥(缺省属性)
TSM_KEY_AUTHORIZATION	使用需授权的密钥
TSM_KEY_AUTHORIZATION_PRIV_USE_ONLY	密钥的私钥部分使用时需授权的密钥
TSM_KEY_STRUCT_KEY	使用 TCM 密钥对象
TSM_KEY_EMPTY_KEY	非 TCM 密钥模板(空 TSM 密钥对象)
TSM_KEY_TSP_SMK	使用 TCM SMK 模板(用于 SMK 的 TSM 密钥对象)
TSM_ENCDATA_SEAL	用于数据封装操作的数据对象
TSM_ENCDATA_BIND	用于加密操作的数据对象
TSM_ENCDATA_ENVELOP	用于数字信封操作的数据对象
TSM_HASH_DEFAULT	缺省密码杂凑算法
TSM_HASH_SCH	SM3 算法的杂凑对象
TSM_HASH_OTHER	其它算法的杂凑对象
TSM_POLICY_USAGE	用于授权的策略对象
TSM_POLICY_MIGRATION	用于密钥迁移的策略对象
TSM_POLICY_OPERATOR	用于操作者授权的策略对象
TSM_PCRS_STRUCT_INFO	使用 TCM 的 PCR 对象
TSM_EXCHANGE_DEFAULT	密钥协商对象

### A.1.2.3 上下文对象的属性定义

属性标记：

TSM_TSPATTRIB_CONTEXT_SILENT_MODE	获取或设置一个上下文对象的休眠模式
TSM_TSPATTRIB_CONTEXT_MACHINE_NAME	获得 TSM 的机器名
TSM_TSPATTRIB_CONTEXT_VERSION_MODE	获取或设置版本,该信息可用于处理上下文对象的模式
TSM_TSPATTRIB_CONTEXT_CONNECTION_VERSION	获得连接的最高支持版本(TSM 和 TCM 的最高通用版本)
TSM_TSPATTRIB_CONTEXT_TRANSPORT	获取或设置与该上下文对象相关联的传输会话的相关属性
TSM_TSPATTRIB_SECRET_HASH_MODE	获得/设置串的杂凑操作模式
TSM_TSPATTRIB_CONTEXTTRANS_CONTROL	打开与关闭传输会话
TSM_TSPATTRIB_CONTEXTTRANS_MODE	控制传输会话的特性
TSM_TSPATTRIB_SECRET_HASH_MODE_POPUP	获取或设置在弹出模式下杂凑操作

属性值：

TSM_TSPATTRIB_CONTEXT_NOT_SILENT	请求用户提供密码时,显示 TSM 对话框
TSM_TSPATTRIB_CONTEXT_SILENT	不显示 TSM 对话框(默认)
TSM_TSPATTRIB_TRANSPORT_NO_DEFAULT_ENCRYPTION	使传输会话中数据加密功能关闭
TSM_TSPATTRIB_TRANSPORT_DEFAULT_ENCRYPTION	使传输会话中数据加密功能打开
TSM_TSPATTRIB_TRANSPORT_EXCLUSIVE	排它传输模式

### A.1.2.4 TCM 对象属性定义

属性标记：

TSM_TSPATTRIB_TCMCAP_SET_VENDOR	允许厂商在 TCM 中按照常规受保护区域位置的需求设置特定区域
TSM_TSPATTRIB_TCM_ORDINAL_AUDIT_STA_TUS	向审计列表添加或清除一个命令码
TSM_TSPATTRIB_TCMCAP_MIN_COUNTER	表示单调计数器递增的最长时间间隔,该间隔以 1/10 秒为度量单位。
TSM_TSPATTRIB_TCMCAP_FLAG_VOLATILE	返回 TCM 启动标志。

证书部分的属性标记：

标记	子标记	属性描述
TSM_TSPATTRIB_TCM_CREDENTIAL	TSM_TCMATTRIB_EKCERT	密码模块证书 blob
	TSM_TCMATTRIB_PLATFORMCERT	平台身份数字证书 blob

表(续)

标记	子标记	属性描述
TSM_TSPATTRIB_TCM_ORDINAL_AUDIT_STATUS	TCM_CAP_PROP_TCM_SET_ORDINAL_AUDIT	向审计列表中加入一个命令码
	TCM_CAP_PROP_TCM_CLEAR_ORDINAL_AUDIT	要添加到审计列表中或者要从审计列表中删除的命令码

### A.1.2.5 策略对象属性定义

属性标记：

TSM_TSPATTRIB_POLICY_SECRET_LIFETIME	设置/获得授权数据的生命周期
TSM_TSPATTRIB_POLICY_POPUPSTRING	设置一个以 NULL 结尾的 TSM_UNICODE 字符串,该字符串在 TSM 策略弹出对话框中显示。
TSM_TSPATTRIB_SECRET_HASH_MODE	获得/设置上下文或策略对象的杂凑操作模式

子属性标记：

TSM_TSPATTRIB_POLSECRET_LIFETIME_ALWAYS	授权数据总是有效
TSM_TSPATTRIB_POLSECRET_LIFETIME_COUNTER	授权数据可多次使用
TSM_TSPATTRIB_POLSECRET_LIFETIME_TIMER	授权数据有效期为 n 秒
TSM_TSPATTRIB_SECRET_HASH_MODE_POPUP	获得/设置弹出模式的杂凑行为

### A.1.2.6 密钥对象属性定义

属性标记：

TSM_TSPATTRIB_KEY_REGISTER	获得/设置密钥所注册的永久存储区
TSM_TSPATTRIB_KEY_BLOB	获得/设置密钥 blob
TSM_TSPATTRIB_KEY_INFO	获得密钥信息
TSM_TSPATTRIB_KEY_UUID	获得 TSM_UUID 结构,该结构包含为密钥所分配的 UUID
TSM_TSPATTRIB_KEY_PCR	获得密钥所封装到的 PCR 信息(用于采用 TSM_KEY_STRUCT_KEY 结构的密钥)
TSM_TSPATTRIB_KEY_CONTROLBIT	获得加载的密钥属性

子属性标记：

TSM_TSPATTRIB_KEYREGISTER_USER	密钥注册到用户永久存储区
TSM_TSPATTRIB_KEYREGISTER_SYSTEM	密钥注册到系统永久存储区
TSM_TSPATTRIB_KEYREGISTER_NO	密钥未注册到永久存储区
TSM_TSPATTRIB_KEYBLOB_BLOB	密钥 blob 形式的密钥信息
TSM_TSPATTRIB_KEYBLOB_PUBLIC_KEY	公钥 blob 形式的公钥信息

表 (续)

TSM_TSPATTRIB_KEYBLOB_PRIVATE_KEY	私钥 blob, 是加密的私钥信息
TSM_TSPATTRIB_KEYINFO_SIZE	密钥的比特长度
TSM_TSPATTRIB_KEYINFO_USAGE	密钥使用信息
TSM_TSPATTRIB_KEYINFO_KEYFLAGS	密钥标志
TSM_TSPATTRIB_KEYINFO_AUTHUSAGE	密钥授权使用信息
TSM_TSPATTRIB_KEYINFO_ALGORITHM	密钥算法标识
TSM_TSPATTRIB_KEYINFO_SIGSCHEME	密钥签名方案
TSM_TSPATTRIB_KEYINFO_ENCSCHEME	密钥加密方案
TSM_TSPATTRIB_KEYINFO_MIGRATABLE	若为真则密钥是可迁移的
TSM_TSPATTRIB_KEYINFO_VOLATILE	若为真则密钥是易失性的
TSM_TSPATTRIB_KEYINFO_AUTHDATAUSAGE	若为真则需要授权
TSM_TSPATTRIB_KEYINFO_VERSION	TSM 版本结构信息
TSM_TSPATTRIB_KEYINFO_KEYSTRUCT	密钥结构类型
TSM_TSPATTRIB_KEYPCR_LOCALITY_A TCREATION	创建 blob 时的 Locality
TSM_TSPATTRIB_KEYPCR_LOCALITY_A TRELEASE	使用密钥所需要的 locality
TSM_TSPATTRIB_KEYPCR_CREATION_SELECTION	选定创建 blob 时活动的 PCR
TSM_TSPATTRIB_KEYPCR_RELEASE_SELECTION	选定使用密钥所需要的 PCR
TSM_TSPATTRIB_KEYPCR_DIGEST_ATC REATION	digestAtCreation 值
TSM_TSPATTRIB_KEYPCR_DIGEST_ATR ELEASE	digestAtRelease 值

#### A. 1.2.7 数据对象属性定义

属性标记：

TSM_TSPATTRIB_ENCDATA_BLOB	获得/设置数据 blob
TSM_TSPATTRIB_ENCDATA_PCR	获得/设置用于封装数据的 PCR 信息(对于使用 TSM_PCRS_STRUCT_INFO 结构的加密数据对象)
TSM_TSPATTRIB_ENCDATA_SEAL	获得/设置封装操作的参数

子属性标记：

TSM_TSPATTRIB_ENCDATABLOB_BLOB	数据 blob, 表示加密的数据, 取决于其类型(封装、加密或数字信封)
TSM_TSPATTRIB_ENCDATAPCR_DIGEST_ATCREATION	获得封装时的 PCR 的组合摘要值
TSM_TSPATTRIB_ENCDATAPCR_DIGEST_ATRELEASE	获得为解封装选择的 PCR 的组合摘要值

表 (续)

TSM_TSPATTRIB_ENCDATAPCR_SELECTION	获得表示活动 PCR 的位图
TSM_TSPATTRIB_ENCDATAPCRLONG_LOCALITY_ATCREATION	获得封装时的 Locality 值
TSM_TSPATTRIB_ENCDATAPCRLONG_LOCALITY_ATRELEASE	获得解封时的 Locality 值
TSM_TSPATTRIB_ENCDATAPCRLONG_CREATION_SELECTION	获得表示封装时活动的 PCR 的位图
TSM_TSPATTRIB_ENCDATAPCRLONG_RELEASE_SELECTION	获得表示解封时活动的 PCR 的位图
TSM_TSPATTRIB_ENCDATAPCRLONG_DIGEST_ATCREATION	获得封装时选择的 PCR 的组合摘要
TSM_TSPATTRIB_ENCDATAPCRLONG_DIGEST_ATRELEASE	获得为解封选择的 PCR 的组合摘要值

#### A. 1.2.8 NV 对象属性定义

属性标记：

TSM_TSPATTRIB_NV_INDEX	NV 存储区的索引
TSM_TSPATTRIB_NV_PERMISSIONS	NV 权限
TSM_TSPATTRIB_NV_STATE	获得 NV 存储区的各种状态, 只有在 NV 空间已经被定义的情况下才可用
TSM_TSPATTRIB_NV_DATASIZE	定义的 NV 存储区的大小
TSM_TSPATTRIB_NV_PCR	NV 存储区的 PCR 限制

子标记：

TSM_TSPATTRIB_NVSTATE_READSTCLEAR	每次启动 TCM 时设置为 FALSE, 在 datasize 为 0 的 ReadValuexxx 之后, 设置为 TRUE。
TSM_TSPATTRIB_NVSTATE_WREESTCLEAR	每次启动 TCM 时设置为 FALSE, 在 datasize 为 0 的 WriteValuexxx 之后, 设置为 TRUE。
TSM_TSPATTRIB_NVSTATE_WREDDEFINE	在 TCM 定义 NV 空间之后设置为 FALSE, 在 datasize 为 0 的 WriteValue 的成功调用之后, 设置为 TRUE。
TSM_TSPATTRIB_NVPCR_READPCRSELECTION	Locality 选择掩码, 用于 NV 区域的 PCR 读取限制。
TSM_TSPATTRIB_NVPCR_READDIGESTATRELEASE	digestAtRelease, 用于 NV 区域的 PCR 读取限制。
TSM_TSPATTRIB_NVPCR_READLOCALITYATRELEASE	Locality 掩码, 用于 NV 区域的 PCR 读取限制。
TSM_TSPATTRIB_NVPCR_WRITEPCRSELECTION	Locality 选择掩码, 用于 NV 区域的 PCR 写限制
TSM_TSPATTRIB_NVPCR_WREDIGESTATRELEASE	DigestAtRelease, 用于 NV 区域的 PCR 写限制
TSM_TSPATTRIB_NVPCR_WRITELOCALITYATRELEASE	Locality 掩码, 用于 NV 区域的 PCR 写限制。

NV 常量：

NV Index Domain Bits	描述
TSM_NV_TCM	值“T”，该索引为 TCM 厂商保留位。0 表示本标准定义，1 表示厂商特定值。
TSM_NV_PLATFORM	值“P”，该索引为平台厂商保留位。1 表示厂商特定值。
TSM_NV_USER	值“U”，该索引为平台用户保留。1 表示平台用户特定值。
TSM_NV_DEFINED	值“D”，该索引已被定义。1 表示该索引被永久定义，任何定义该索引空间的操作都将失败。

NV Index Masks	描述
TSM_NV_MASK_DOMAIN_BITS	位，值“1”用于索引域
TSM_NV_MASK_RESERVED	保留位
TSM_NV_MASK_PURVIEW	掩码范围
TSM_NV_MASK_INDEX	这些位用于索引

NV Required Indexes	描述
TSM_NV_INDEX_LOCK	长度必须为 0。该值打开 NV 授权保护。一旦执行，所有 NV 存储区所定义的保护都被打开，该值从不重置。
TSM_NV_INDEX0	长度必须为 0。该值允许设置永久锁定位，该位只能在 TCM 启动时重置。

NV Permissions	描述
TCM_NV_PER_READ_STCLEAR	TCM 启动后该值只能被读一次。Datasize 为 0 的读操作将设置为锁定。
TCM_NV_PER_AUTHREAD	读该值需要授权
TCM_NV_PER_OWNERREAD	读该值需要 TCM 所有者授权
TCM_NV_PER_PPREAD	读该值需要物理现场
TCM_NV_PER_GLOBALLOCK	在成功地向索引 0 写之前该值都是可写的。对此属性的锁定可由 TCM 启动命令复位。锁定由 SV->bGlobalLock 保持。
TCM_NV_PER_WRITE_STCLEAR	Datasize 为 0 的指定索引是成功的。此属性的锁定由 TCM 启动命令复位。bWriteSTClear 中每个区域都保持锁定。
TCM_NV_PER_WRITEDEFINE	TCM 定义 NV 空间后，该值只能被写一次。bWriteDefine 中每个区域都保持锁定，datasize 为 0 的索引写操作将设置锁定。
TCM_NV_PER_WRITEALL	该值必须在一个单一的操作中被写入。
TCM_NV_PER_AUTHWRITE	写该值需要授权
TCM_NV_PER_OWNERWRITE	写该值需要 TCM 所有者授权
TCM_NV_PER_PPWRITE	写该值需要物理现场

### A. 1. 2. 9 迁移数据对象属性定义

属性	子属性	描述
TSM_MIGATTRIB_MIGRATION_BLOB	TSM_MIGATTRIB_MIGRATION_SM4_BLOB	CreateBlob 操作输出的数据包(采用 SM4 算法)
	TSM_MIGATTRIB_MIGRATION_XOR_BLOB	CreateBlob 操作输出的数据包
	TSM_MIGATTRIB_MIGRATION_REWRAPPED_BLOB	密钥迁移操作后的数据格式
	TSM_MIGATTRIB_MIG_DESTINATION_PUBKEY_BLOB	经过认证的目标平台公钥
TSM_MIGATTRIB_PAYLOAD_TYPE	TSM_MIGATTRIB_PT_MIGRATE_RESTRICTED	本地创建的密钥
	TSM_MIGATTRIB_PT_MIGRATE_EXTERNAL	迁移到此的密钥

### A. 1. 2. 10 杂凑对象属性定义

属性标记：

TSM_TSPATTRIB_ALG_IDENTIFIER	获得/设置密码杂凑算法标识
------------------------------	---------------

### A. 1. 2. 11 Secret Mode 策略定义

策略模式标记定义,可用于 Tspi\_Policy\_SetSecret()方法. 定义的授权数据模式基于 TSM\_FLAG 数据类型

Secret Mode	描述
TSM_SECRET_MODE_NONE	不设置策略授权信息,这与 0x00 的授权不一样。
TSM_SECRET_MODE_SM3	输入经外部杂凑计算得到的 32 个字节数组作为授权数据。TSM 不会修改这个输入数据。
TSM_SECRET_MODE_PLAIN	输入一个任意字节的数组,然后将其进行杂凑运算,作为授权数据。
TSM_SECRET_MODE_POPUP	TSM 向用户询问一个 TSM_UNICODE 类型的授权口令串,该串需要进行杂凑运算作为授权数据。

### A. 1. 2. 12 Secret 生命周期策略定义

通过 Tspi\_SetAttribUint32()和 Tspi\_GetAttribUint32()函数能够设置/获取授权数据的生命周期。生命周期模式的定义是基于 TSM\_FLAG 类型的。

TSM_SECRET_LIFETIME_ALWAYS	授权数据将一直有效
----------------------------	-----------

## TCM 状态标记定义

Flag	描述	使用
TSM_TCMSTATUS_DISABLEOWNERCLEAR	永久禁止 TCM 所有者进行 ClearOwner 操作。此时,方法 ClearOwner() 中的 fForcedClear 参数将不再允许取 FALSE 值。这个设置需要所有者授权。	SetStatus GetStatus
TSM_TCMSTATUS_DISABLEFORCECLEAR	临时禁止 TCM 所有者的强制清除操作(这种禁止只在本次系统运行时有效,在下一次系统重新启动时将被取消)。此时,方法 ClearOwner() 中的 fForcedClear 参数将暂时不允许取 TRUE 值(直到下次系统重新启动为止)	SetStatus GetStatus
TSM_TCMSTATUS_OWNERSETDISABLE	fTCMState=TRUE: 表示设置 TCM 的状态为 Disabled 时,不需要 TCM 所有者的授权。	SetStatus GetStatus
TSM_TCMSTATUS_PHYSICALDISABLE	fTCMState=TRUE: 表示设置 TCM 的状态为 Disabled 时,必须是物理现场的。	SetStatus GetStatus
TSM_TCMSTATUS_PHYSICALSETDEACTIVATED	fTCMState=TRUE: 表示设置 TCM 的状态为 Deactivated 时,必须是物理现场的。	SetStatus GetStatus
TSM_TCMSTATUS_SETTEMPORARYDEACTIVATED	暂时将 TCM 的状态设置为 Deactivated(直到下次系统重新启动为止)。	SetStatus GetStatus
TSM_TCMSTATUS_SETOWNERINSTALL	fTCMState=TRUE: 表示允许使用 TakeOwnership() 方法来取得 TCM 的所有者关系。这个操作需要物理现场。	SetStatus GetStatus
TSM_TCMSTATUS_DISABLEPUBEKREAD	永久禁止在没有 TCM 所有者授权的情况下读取 EK 公钥信息的操作,即设置该属性后,必须有 TCM 所有者授权才能进行读取 EK 公钥信息的操作。设置了这个属性后,GetPubEndorsementKey() 方法中的 fOwnerAuthorized 参数取 FALSE 值不可能再有效了。设置这个属性值需要所有者授权。	SetStatus GetStatus
TSM_TCMSTATUS_DISABLELED	将 TCM 设置为可用或不可用状态。	SetStatus GetStatus
TSM_TCMSTATUS_DEACTIVATED	将 TCM 设置为激活或非激活状态。	SetStatus GetStatus
TSM_TCMSTATUS_PHYSICALPRESENT_LIFETIMELOCK	* pfTCMState=TRUE: 表示在 TCM 生存期内, TCM 的 physicalPresenceHWEable 和 physicalPresenceCMDEnable 标志都不允许修改。	GetStatus
TSM_TCMSTATUS_PHYSICALPRESENT_HWENABLE	* pfTCMState=TRUE: 表示 TCM 物理在线的硬件信号被允许可以用做物理在线的标志。	GetStatus
TSM_TCMSTATUS_PHYSICALPRESENT_CMDENABLE	* pfTCMState=TRUE: 表示允许使用 TCM 命令 TSC_PhysicalPresence 来表明物理在线。	GetStatus
TSM_TCMSTATUS_CEKP_USED	* pfTCMState=TRUE: 表明 EK 密钥对是使用 CreateEndorsementKey() 方法来生成的。 * pfTCMState=FALSE: 表明 EK 密钥对是由厂家创建的。	GetStatus
TSM_TCMSTATUS_PHYSICALPRESENCE	* pfTCMState=TRUE: 表示物理在线的软件标志。	GetStatus

表 (续)

Flag	描述	使用
TSM_TCMSTATUS_PHY_SPRES_LOCK	* pfTCMState=TRUE: 表示改变物理在线的标志的操作是不允许的。	GetStatus
TSM_TCMSTATUS_ENABLE_REVKEEK	表明是否允许 EK 被重新设置	GetStats
TSM_TCMSTATUS_NV_LOCK	* pfTCMState=TRUE: 表示 NV 授权访问是必需的。	GetStats

### A. 1.2.13 算法标识符定义

算法标识符的定义基于 TSM\_ALGORITHM\_ID 数据类型

算法标识符	描述
TCM_ALG_SM2	SM2 算法
TCM_ALG_SM4	SM4 密码算法
TCM_ALG_SCH	SM3 密码算法
TCM_ALG_HMAC	HMAC 算法
TCM_ALG_XOR	XOR 算法
TCM_ALG_KDF	KDF 算法

### A. 1.2.14 功能特性标记定义

以下标记定义了需要查询的功能特性,其定义是基于 TSM\_FLAG 的。

TCM 功能特性标记:

功能特性	描述
TSM_TCMCAP_ORD	查询 TCM 是否支持该命令。
TSM_TCMCAP_ALG	查询是否支持该算法。
TSM_TCMCAP_FLAG	返回所有永久性和易失性比特标志位的序列。
TSM_TCMCAP_PROPERTY	判断 TCM 的物理特性。
TSM_TCMCAP_VERSION	查询当前 TCM 版本。
TSM_TCMCAP_NV_LIST	获取用来定义 NV 存储区的索引列表。
TSM_TCMCAP_NV_INDEX	获取指定的 NV 存储区的值
TSM_TCMCAP_MFR	获取厂商特定的 TCM 和 TCM 状态信息。
TSM_TCMCAP_SYM_MODE	布尔值。查询 TCM 是否支持特定类型的对称加密。
TSM_TCMCAP_HANDLE	返回 TCM 中当前已加载对象的句柄列表
TSM_TCMCAP_TRANS_ES	查询 TCM 是否在传输会话中支持特定的加密方案。
TSM_TCMCAP_AUTH_ENCRYPT	查询 TCM 是否在授权会话的 AuthData 加密中支持特定的加密方案
TSM_TCSCAP_TRANSPORT	查询是否支持传输功能

Tspi\_TCM\_GetCapability() 函数通过将 Capability 数据字段设置为 TSM\_TCMLCAP\_FLAG，得到代表 TCM 功能特性的一串字节串，这些串是主机字节序(big-endian)的。

返回的前四字节表示 UINT32 形式的永久性标记，接下来的四个字节表示 UINT32 形式的易失性标记。永久性标记的 31-28 位在第一个字节返回，易失性标记的 3-0 比特在最后一个字节返回。位 0 定义为 UINT32 的第 1 位。

永久性标记：

TCM_PF_DISABLE	0x00000001
TCM_PF_OWNERSHIP	0x00000002
TCM_PF_DEACTIVATED	0x00000004
TCM_PF_READPUBEK	0x00000008
TCM_PF_DISABLEOWNERCLEAR	0x00000010
TCM_PF_PHYSICALPRESENCELIFETIMELOCK	0x00000040
TCM_PF_PHYSICALPRESENCEHWENABLE	0x00000080
TCM_PF_PHYSICALPRESENCECMENABLE	0x00000100
TCM_PF_CEKPUSED	0x00000200
TCM_PF_TCMPOST	0x00000400
TCM_PF_TCMPOSTLOCK	0x00000800

易失性标记：

TCM_SF_DEACTIVATED	0x00000001
TCM_SF_DISABLEFORCECLEAR	0x00000002
TCM_SF_PHYSICALPRESENCE	0x00000004
TCM_SF_PHYSICALPRESENCELOCK	0x00000008

没有在以上图表中定义的比特位未被使用。

功能特性	描述
TSM_TCSCAP_ALG	查询一个算法是否被支持。
TSM_TCSCAP_VERSION	查询当前的 TSM 版本
TSM_TCSCAP_MANUFACTURER	查询当前的 TSM 厂商信息
TSM_TCSCAP_CACHING	查询是否支持密钥和授权的缓存
TSM_TCSCAP_PERSSTORAGE	查询是否支持永久性存储

TSM 功能特性：

Capability Area	描述
TSM_TSPCAP_ALG	查询是否支持某个算法。
TSM_TSPCAP_VERSION	查询当前的 TSM 版本
TSM_TSPCAP_PERSSTORAGE	查询是否支持永久性存储
TSM_TSPCAP_MANUFACTURER	查询当前的 TSM 厂商信息

### A. 1. 2. 15 子属性标记定义

子属性标记从属于属性标记,其定义基于 TSM\_TCMCAP\_PROPERTY。

TCM 子属性标记:

子属性	返回值
TSM_TCMCAP_PROP_PCR	UINT32 值。返回 TCM 支持的 PCR 寄存器个数
TSM_TCMCAP_PROP_PCRMAPP	返回 TCM_PCR_ATTRIBUTES 的比特标志位。TCM_PCR_ATTRIBUTES 是与 TSM_TCMCAPPROP_PCR 相关的。
TSM_TCMCAP_PROP_MANUFACTURER	UINT32 值。返回 TCM 厂商的标识。
TSM_TCMCAP_PROP_SLOTS 或 TSM_TCMCAP_PROP_KEYS	UINT32 值。返回 TCM 可以加载的 256 位 SM2 密钥的最大个数。可随时间和情况而改变。
TSM_TCMCAP_PROP_OWNER	布尔值。返回 TRUE 表示 TCM 成功地创建了一个所有者。
TSM_TCMCAP_PROP_MAXKEYS	UINT32 值。返回 TCM 所支持的 256 位 SM2 密钥的最大个数,不含 Ek 和 SMK。
TSM_TCMCAP_PROP_AUTHSESSIONS	UINT32 值。可用的授权会话的个数,可随时间和情况而改变。
TSM_TCMCAP_PROP_MAXAUTHSESSIONS	UINT32 值。返回 TCM 支持的可加载授权会话的最大个数,可随时间和情况而改变。
TSM_TCMCAP_PROP_TRANSESSIONS	UINT32 值。返回可用传输会话的个数,可随时间和情况而改变。
TSM_TCMCAP_PROP_MAXTRANSESSIONS	UINT32 值。返回 TCM 支持的可加载传输会话的最大个数。
TSM_TCMCAP_PROP_SESSIONS	UINT32 值。返回会话池中可用会话的个数。会话池中的会话包括授权会话和传输会话,可随时间和情况而改变。
TSM_TCMCAP_PROP_MAXSESSIONS	UINT32 值。返回 TCM 支持的最大会话个数,包括授权会话和传输会话。
TSM_TCMCAP_PROP_CONTEXTS	UINT32 值。返回可保存的会话个数,可随时间和情况而改变。
TSM_TCMCAP_PROP_MAXCONTEXTS	UINT32 值。返回保存的会话的最大个数。
TSM_TCMCAP_PROP_COUNTERS	UINT32 值。返回可用单调计数器的个数,可随时间和情况而改变。
TSM_TCMCAP_PROP_MAXCOUNTERS	UINT32 值。返回 TCM_CreateCounter 控制的单调计数器的最大个数。
TSM_TCMCAP_PROP_MINCOUNTERINCTIME	UINT32 值。表示单调计数器递增的最短时间间隔,该间隔以 1/10 秒为度量单位。
TSM_TCMCAP_PROP_ACTIVECOUNTER	返回当前计数器的 ID。若没有活动的计数器,则返回 0xff..ff。

表 (续)

子属性	返回值
TSM_TCMCAP_PROP_TISTIMEOUTS	UINT32 值四元数组, 每个元素表示以毫秒计的超时值, 顺序如下: TIMEOUT_A, TIMEOUT_B, TIMEOUT_C, TIMEOUT_D。由平台特定的接口规范决定在何处使用这些超时值。
TSM_TCMCAP_PROP_STARTUPEFFECTS	返回 TCM_STARTUP_EFFECTS 结构
TSM_TCMCAP_PROP_MAXCONTEXTCOUNTDIST	UINT32 值。返回上下文计数值的最大间距, 至少必须为 2^16-1。
TSM_TCMCAP_PROP_DURATION	返回 UINT32 值三元数组, 每个元素依次表示如下三类命令以毫秒计的周期值: SMALL_DURATION, MEDIUM_DURATION, LONG_DURATION
TSM_TCMCAP_PROP_MAXNVAVAILABLE	UINT32 值。返回可分配的 NV 区域的最大个数, 可随时间和情况而改变。
TSM_TCMCAP_PROP_MAXNVWRITE	在 TCM 还没有所有者时, 此值表示对 NV 进行写操作的次数。
TSM_TCMCAP_PROP_REVISION	BYTE, 用标准结构标识 TCM 的主版本号和次版本号, 主版本号在前, 次版本号在后
TSM_TCMCAP_PROP_LOCALITIES_AVAIL	TCM 中可用的 localities 的数目
TSM_TCMCAP_PROP_INPUTBUFFERSIZE	UINT32. 返回 TCM 中用作输入的缓冲区的大小, 以字节为单位

TSM 子属性:

子属性	描述
TSM_TCSCAP_PROP_MANUFACTURER_STR	返回由 TSM 生产商指定内容的字符串, 它的数据类型为 TSM_UNICODE。
TSM_TCSCAP_PROP_MANUFACTURER_ID	返回一个 UINT32 类型的数值, 标识特定的 TSM 生产商。

TSM\_TCSCAP\_CACHING 子属性定义:

子属性	描述
TSM_TCSCAP_PROP_KEYCACHE	类型为 TSM_BOOL 的数值, 标识是否支持密钥缓存。
TSM_TCSCAP_PROP_AUTHCACHE	类型为 TSM_BOOL 的数值, 标识是否支持授权会话缓存

TSM 子属性:

子属性	描述
TSM_TSPCAP_PROP_MANUFACTURER_STR	返回由生产商定制内容的字符串, 它的数据类型为 TSM_UNICODE
TSM_TSPCAP_PROP_MANUFACTURER_ID	返回一个 UINT32 类型的数值, 标识特定的 TSM 生产商

### A. 1. 2. 16 永久存储区标记定义

永久性存储类型	描述
TSM_PS_TYPE_USER	密钥被注册到用户永久存储空间
TSM_PS_TYPE_SYSTEM	密钥被注册到系统永久存储空间

### A. 1. 2. 17 迁移方案定义

迁移方案	描述
TSM_MS_MIGRATE	迁移时用来保护另一个密钥的密钥,用法是先调用函数 Tspi_Key_CreateMigrationBlob,再调用函数 Tspi_Key_ConvertMigrationBlob
TSM_MS_REWRAP	迁移保护密钥的公钥部分,通过调用函数 Tspi_Key_CreateMigrationBlob 对被迁移密钥进行重新加密。

### A. 1. 2. 18 密钥用途定义

密钥用途	描述
TCM_SM2KEY_STORAGE	用于存储保护密钥树中其它密钥的 SM2 密钥
TCM_SM2KEY_SIGNING	用于签名的 SM2 密钥
TCM_SM2KEY_BIND	用于数据加解密的 SM2 密钥
TCM_SM2KEY_IDENTITY	SM2 密钥,该密钥只用于那些需要 TCM 身份的操作
TCM_SM2KEY_MIGRATE	可用于密钥迁移的 SM2 密钥
TCM_SM2KEY_PEK	SM2 平台加密密钥
TCM_SM4KEY_STORAGE	用于存储保护密钥树中其它密钥的 SM4 密钥
TCM_SM4KEY_BIND	用于数据加解密的 SM4 密钥

### A. 1. 2. 19 密钥长度定义

密钥长度	描述
TSM_KEY_SIZEVAL_128BIT	密钥长度为 128 位
TSM_KEY_SIZEVAL_256BIT	密钥长度为 256 位(SM2 私钥)
TSM_KEY_SIZEVAL_512BIT	密钥长度为 512 位(SM2 公钥)

### A. 1. 2. 20 密钥类型标记

密钥类型	描述
TSM_KEYFLAG_MIGRATABLE	若密钥是可迁移的,则此值需设置为 1
TSM_KEYFLAG_VOLATILEKEY	若密钥是易失性的,则此值需设置为 1

### A. 1. 2. 21 密钥结构类型

下表定义的密钥数据结构由 Tspi\_GetAttribUInt32(TSM\_TSPATTRIB\_KEY\_INFO, TSM\_TSPATTRIB\_KEYINFO\_KEYSTRUCT) 返回, 或者由 Tspi\_SetAttribUInt32(TSM\_TSPATTRIB\_KEY\_INFO, TSM\_TSPATTRIB\_KEYINFO\_KEYSTRUCT) 设置。

密钥结构类型	描述
TSM_KEY_STRUCT_KEY	密钥对象采用 TCM 密钥结构

### A. 1. 2. 22 密钥授权

密钥授权	描述
TSM_KEYAUTH_AUTH_NEVER	密钥不需要授权
TSM_KEYAUTH_AUTH_ALWAYS	密钥需要授权
TSM_KEYAUTH_AUTH_PRIV_USE_ONLY	此值表示如果有命令需要用到密钥的私钥部分, 则一定需要授权, 如果有命令只需使用密钥的公钥部分而不需要密钥的私钥部分, 则不需要授权。

### A. 1. 2. 23 密钥使用方案定义

算法标识符	描述
TSM_ES_NONE	没有设置加密方案
TSM_ES_SM2	采用 SM2 算法进行加密
TSM_ES_SM4_CBC	采用 SM4_CBC 方案进行加密
TSM_ES_SM4_ECB	采用 SM4_ECB 方案进行加密

### A. 1. 2. 24 签名方案定义

算法标识	描述
TSM_SS_NONE	没有设置签名方案
TSM_SS_SM2	采用国标签名方案

### A. 1. 2. 25 PCR 结构类型

PCR 结构类型	描述
TSM_PCRS_STRUCT_INFO	Pcr 对象采用 TCM 的 PCR 数据结构

### A. 1. 2. 26 共享秘密

公开的共享秘密	描述
TSM_WELL_KNOWN_SECRET	32 位全零

## A.2 数据结构

### A.2.1 TSM\_VERSION

定义：

```
typedef struct tdTSM_VERSION
{
    BYTE bMajor;
    BYTE bMinor;
    BYTE bRevMajor;
    BYTE bRevMinor;
} TSM_VERSION;
```

成员变量：

——bMajor	本 TSM 规范实现的主版本号标识码；
——bMinor	本 TSM 规范实现的次版本号标识码；
——bRevMajor	TSM 厂商实现的主版本号取值，由 TSM 厂商确定；
——bRevMinor	TSM 厂商实现的次版本号取值，由 TSM 厂商确定。

### A.2.2 TSM\_PCR\_EVENT

该结构提供有关单个 PCR 扩展事件的信息。

定义：

```
typedef struct tdTSM_PCR_EVENT
{
    TSM_VERSION versionInfo;
    UINT32 ulPcrIndex;
    TSM_EVENTTYPE eventType;
    UINT32 ulPcrValueLength;
    BYTE* rgbPcrValue;
    UINT32 ulEventLength;
    BYTE* rgbEvent;
} TSM_PCR_EVENT;
```

成员变量：

——versionInfo	由 TSM 设置的版本数据；
——ulPcrIndex	由 TSM 设置的该事件所属的 PCR 索引
——eventType	事件类型标记
——ulPcrValueLength	参数 rgbPcrValue 的数据长度(以字节为单位)，由 TSM 设置
——rgbPcrValue	指向通过调用函数 Tspi_TCM_PcrExtend 扩展到 TCM 的值
——ulEventLength	参数 rgbEvent 的数据长度(以字节为单位)
——rgbEvent	PCR 事件信息

### A.2.3 TSM\_EVENT\_CERT

证书结构，用于 TSM\_EV\_CODE\_CERT 类型的事件

定义：

```

typedef struct tdTSM_EVENT_CERT
{
    TSM_VERSION versionInfo;
    UINT32 ulCertificateHashLength
    BYTE* rgbCertificateHash;
    UINT32 ulEntityDigestLength
    BYTE* rgbEntityDigest;
    TSM_BOOL fDigestChecked;
    TSM_BOOL fDigestVerified;
    UINT32 ulIssuerLength;
    BYTE* rgbIssuer;
} TSM_EVENT_CERT;

```

成员变量：

——versionInfo	版本信息
——ulCertificateHashLength	参数 rgbCertificateHash 的数据长度(以字节为单位)
——rgbCertificateHash	指向整个 VE(Validation Entity)证书的杂凑值
——ulEntityDigestLength	参数 rgbEntityDigest 的数据长度(以字节为单位)
——rgbEntityDigest	指向整个证书的实际摘要值
——fDigestChecked	如果此值为 TRUE,则需要将整个事件日志与证书中的摘要值进行比较;如果此值为 FALSE,则不需要检查
——fDigestVerified	只有在 fDigestChecked 为 TRUE 的情况下,此值才起作用。如果度量值与证书中的摘要值匹配,则此值为 TRUE,否则为 FALSE
——ulIssuerLength	参数 rgbIssuer 的数据长度(以字节为单位)
——rgbIssuer	指向颁发者证书

#### A. 2. 4 TSM\_UUID

本数据结构提供关于一个 UUID 标识符的信息,该标识符在一个给定平台密钥树范围内是唯一的。有些 UUID 专门为一些特定密钥保留,比如 SMK。

这些 UUID 用于在 TSM 密钥管理器的永久性存储区中注册密钥。

UUID 定义遵从 IEEE 802。

定义：

```

typedef struct tdTSM_UUID
{
    UINT32 ulTimeLow;
    UINT16 usTimeMid;
    UINT16 usTimeHigh;
    BYTE bClockSeqHigh;
    BYTE bClockSeqLow;
    BYTE rgbNode [6];
} TSM_UUID;

```

成员变量：

——ulTimeLow:	时间戳的低字段
——usTimeMid:	时间戳的中间字段

——usTimeHigh: 时间戳的高字段乘以版本号  
 ——bClockSeqHigh: 时钟序列高字段乘以变量  
 ——bClockSeqLow: 时钟序列低字段  
 ——rgbNode: 唯一性节点标识符

### A.2.5 TSM\_KM\_KEYINFO

TSM\_KM\_KEYINFO 数据结构提供关于注册在 TSM 永久性存储区的密钥信息。

定义:

```
typedef struct tdTSM_KM_KEYINFO
{
    TSM_VERSION versionInfo;
    TSM_UUID keyUUID;
    TSM_UUID parentKeyUUID;
    BYTE bAuthDataUsage;
    TSM_FLAG persistentStorageType;
    TSM_FLAG persistentStorageTypeParent;
    TSM_BOOL fIsLoaded; // TRUE:actually loaded in TCM
    UINT32 ulVendorDataLength; // may be 0
    BYTE* rgbVendorData; // may be NULL
} TSM_KM_KEYINFO;
```

成员变量:

——versionInfo	版本数据
——keyUUID	注册在 TSM 密钥管理器永久性存储区中的密钥 UUID
——parentKeyUUID	注册在 TSM 密钥管理器永久性存储区中的父密钥 UUID, 用来加密由 keyUUID 寻址的密钥
——bAuthDataUsage	表示使用密钥是否需要授权的标记, 目前定义的值为 0x00 与 0x01, 值 0x00 表示密钥使用不需要授权, 值 0x01 表示每次密钥使用都需要进行授权, 其它值将保留给未来版本使用。
——persistentStorageType	标记, 表示密钥注册的永久性存储区
——persistentStorageTypeParent	标记, 表示父密钥注册的永久性存储区
——fIsLoaded	标记, 表示该密钥是否加载到 TCM 中; TRUE 表示密钥已加载到 TCM 中, FALSE 表示密钥未加载到 TCM 中
——ulVendorDataLength	参数 rgbVendorData 的长度(字节)
——rgbVendorData:	指向厂商特定数据的指针

### A.2.6 TSM\_VALIDATION

TSM\_VALIDATION 数据结构用于验证数字签名。

下列函数使用该数据结构:

```
Tspi_TCM_CertifySelfTest,
Tspi_TCM_GetCapabilitySigned,
Tspi_Key_CertifyKey,
Tspi_TCM_CreateEndorsementKey,
Tspi_TCM_GetPubEndorsementKey
```

Tspi\_TCM\_CreateRevocableEndorsementKey

Tspi\_TCM\_Quote

Tspi\_Context\_CloseSignTransport

定义：

```
typedef struct tdTSM_VALIDATION
{
    TSM_VERSION versionInfo;
    UINT32 ulExternalDataLength;
    BYTE* rgbExternalData;
    UINT32 ulDataLength;
    BYTE* rgbData;
    UINT32 ulValidationLength;
    BYTE* rgbValidationData;
} TSM_VALIDATION;
```

成员变量：

——versionInfo	版本数据
——ulExternalData	rgbExternalData 的长度(字节)
——rgbExternalData	内存指针,该内存包含用于抗重放攻击的随机数
——ulDataLength	rgbData 的长度(字节)
——rgbData	用于计算验证有效性的数据
——ulValidationLength	rgbValidationData 的长度(字节)
——rgbValidationData	指向验证数据的指针

#### A. 2. 6. 1 TCM\_COUNTER\_VALUE

本数据结构返回计数器值。计数器长度为 4 个字节。

定义：

```
typedef struct tdTCM_COUNTER_VALUE
{
    TCM_STRUCTURE_TAG tag;
    BYTE label[4];
    TCM_ACTUAL_COUNT counter;
} TCM_COUNTER_VALUE;
```

成员变量：

——tag	标记,对计数器,该值为 0x000E
——label	计数器标签(4 字节)
——counter	32 位的计数器值

#### A. 2. 6. 2 TSM\_ENVELOP

本数据结构用于数字信封函数

定义：

```
typedef struct tdTSM_ENVELOP
```

```

{
    UINT32 encSymSize;
    BYTE* encSym; // 对 TCM_STORE_SYMKEY 结构加密后的数据缓冲区
    UINT32 dataSize;
    BYTE* data;
} TSM_ENVELOP;

成员变量：
——encSymSize encSym 的数据长度(字节)
——encSym 被加密的对称密钥
——dataSize 参数 data 的长度
——data 用对称密钥加密的数据

```

### A.3 授权数据处理

TSM 提供策略对象帮助调用程序处理和缓存授权对象的秘密。

下列对象是授权对象：

- a) TCM；
- b) 密钥；
- c) 加密数据。

TSM 知道何时使用对象的授权信息或者从对象授权信息衍生出来的会话授权信息。一个授权对象只指派一个策略对象，但一个策略对象可以指派给 0 到 n 个授权对象，以方便多个授权对象拥有相同的授权信息。如果一个授权对象没有明确地分配一个策略对象，TSM 自动给它分配默认的策略对象。

当调用需要授权的函数时，应用程序不需要提供授权数据。这个服务不是限制性的，可通过应用程序根据策略对象调整。服务提供者应该使用非分页内存存放提供给它的秘密，这是依赖平台的。释放此内存前，首先将内存区清零。

策略对象的默认模式是 TSM\_SECRET\_MODE\_PLAIN，也可设置为 TSM\_SECRET\_MODE\_POPUP，TSM\_SECRET\_MODE\_SCH

模式：

TSM\_SECRET\_MODE\_POPUP

显示一个输入密码的对话框。用户提供的密码被当作一个以 null 结尾的字符串来处理，并使用 SM3 作杂凑处理得到授权信息。一旦提供了此信息，可能在适当的策略对象中缓存授权信息（依赖策略对象的设置），这样对话框不会再次弹出。

TSM\_SECRET\_MODE\_PLAIN 或 TSM\_SECRET\_MODE\_SM3

应用程序可以基于每个策略设置授权信息。此授权信息可为需要授权的命令提供授权。

类型	定义
TSM_SECRET_MODE_SCH	输入经外部杂凑计算得到的 32 个字节数组作为授权数据
TSM_SECRET_MODE_PLAIN	输入一个任意字节的数组，然后将其进行杂凑运算，作为授权数据。

TSM\_SECRET\_MODE\_NONE：

对于工作对象没有授权数据。

#### A.4 返回码定义

以下定义了 TSM 的返回码定义：

类型	说明
TSM_E_INVALID_OBJECT_TYPE	对象类型对于本操作不符合
TSM_E_INVALID_OBJECT_INIT_FLAG	无效的对象初始化标识
TSM_E_INVALID_HANDLE	无效的对象句柄
TSM_E_NO_CONNECTION	没有建立系统服务连接
TSM_E_CONNECTION_FAILED	与系统服务建立连接失败
TSM_E_CONNECTION_BROKEN	与系统服务已建立连接,但已失败
TSM_E_HASH_INVALID_ALG	无效的密码杂凑算法
TSM_E_HASH_INVALID_LENGTH	杂凑长度与算法不符
TSM_E_HASH_NO_DATA	杂凑对象没有数据
TSM_E_SILENT_CONTEXT	上下文等待用户输入
TSM_E_INVALID_ATTRIB_FLAG	属性标记设置错误
TSM_E_INVALID_ATTRIB_SUBFLAG	子属性标记设置错误
TSM_E_INVALID_ATTRIB_DATA	属性数据错误
TSM_E_NO_PCRS_SET	没有选择或设置 PCR 寄存器
TSM_E_KEY_NOT_LOADED	指定密钥没有被加载
TSM_E_KEY_NOT_SET	密钥信息不可用
TSM_E_VALIDATION_FAILED	数据有效性验证失败
TSM_E_TSP_AUTHREQUIRED	需要授权
TSM_E_TSP_AUTH2REQUIRED	需要多授权
TSM_E_TSP_AUTHFAIL	授权错误
TSM_E_TSP_AUTH2FAIL	多授权错误
TSM_E_KEY_NO_MIGRATION_POLICY	没有为密钥设置迁移策略
TSM_E_POLICY_NO_SECRET	指定的策略对象没有设置授权信息
TSM_E_INVALID_OBJ_ACCESS	对象状态错误导致操作失败
TSM_E_INVALID_ENCSHEME	无效的加密方案标识
TSM_E_INVALID_SIGSCHEME	无效的签名方案标识
TSM_E_INVALID_NEGSCHEME	无效的密钥协商方案标识
TSM_E_ENC_INVALID_LENGTH	无效的加密数据长度
TSM_E_ENC_NO_DATA	无数据供加密
TSM_E_ENC_INVALID_TYPE	无效的加密类型
TSM_E_INVALID_KEYUSAGE	无效的密钥类型
TSM_E_VERIFICATION_FAILED	签名验证错误

表 (续)

类型	说明
TSM_E_HASH_NO_IDENTIFIER	未定义的密码杂凑算法标识
TSM_E_BAD_PARAMETER	参数错误
TSM_E_INTERNAL_ERROR	TCM 内部错误
TSM_E_INVALID_RESOURCE	内存指针错误
TSM_E_PS_KEY_NOTFOUND	密钥不在永久存储区域
TSM_E_NOTIMPL	功能未支持
TSM_TCM_NOT_RESETABLE	PCR 不可重置
TSM_E_WRONG_LOCALITY	命令无法在该 locality 下执行
TSM_E_KEY_NO_MIGRATION_POLICY	需要迁移授权
TSM_E_NV_AREA_NOT_EXIST	定义的 NV 区域不存在
TSM_E_NV_AREA_EXIST	对已经使用的 NV 区重新定义
TSM_WRONGPCRVALUE	PCR 与当前定义的 PCR 不匹配
TSM_NOSPACE	NV 区域已经满
TSM_BAD_PRESENCE	物理在线标识错误
TSM_BAD_LOCALITY	对 Locality 操作不正确
TSM_AUTH_CONFLICT	授权数据冲突
TSM_AUTHFAIL	授权失败
TSM_OWNERSET	所有者已经存在
TSM_BAD_DATASIZE	与指定密钥的数据长度不符合
TSM_MAXNVWRITE	超出 NV 区域可写操作次数
TSM_INVALID_STRUCTURE	结构错误
TSM_AREA_LOCKED	NV 区域已被锁定
TSM_KEY_OWNER_CONTROL	需要所有者来管理密钥
TSM_E_NO_ACTIVE_COUNTER	芯片没有激活计数器
TSM_SUCCESS	命令处理成功

**附录 B**  
**(规范性附录)**  
**数字证书格式**

**B. 1 概述**

本附录采用抽象语法表示法(ASN.1)的特定编码规则(DER)对下列证书项中的各项信息进行编码,组成特定的证书数据结构。ASN.1 DER 编码是关于每个元素的标记、长度和值的编码系统。

本证书格式见 GM/T BBBB 基于 SM2 算法的数字证书格式规范。

**B. 2 证书可信计算扩展域 TCMExtension**

本节为可信计算密码支撑平台定义的证书扩展项,每一扩展和一 OID 联系起来。OID 是 id-tcm 的成员,定义如下:

id-tcm OBJECT IDENTIFIER ::= {1.2.156.10179.4.1.1.1}

可信计算 TCM 信息 tcmInfomation,用于指明 TCM 规范及生产厂商有关信息,定义如下:

id-tcm-tcmInfomation OBJECT IDENTIFIER ::= {id-tcm 1}

tcmInfomation ::= SEQUENCE {

certTag CertTagFlags,

tcmModuleInfo TCMMModuleInfo,

specVersion DisplayText,

tcmClaim DisplayText

}

CertTagFlags ::= BITSTRING{

EK(0),

PIK (1),

PEK (2)

}

TCMMModuleInfo ::= SEQUENCE {

TCMVender [0]EXPLICIT DisplayText,

TCMModel [1]EXPLICIT DisplayText OPTIONAL,

TCMVersion [2]EXPLICIT DisplayText OPTIONAL

}

DisplayText ::= CHOICE {

visibleString VisibleString (SIZE (1..200)),

bmpString BMPString (SIZE (1..200)),

utf8String UTF8String (SIZE (1..200))

}

DisplayText 类型被定义为 VisibleString、BMPString 和 UTF8String 类型之一。在本标准中必须使用 UTF8String 进行编码。

这个扩展必须被标记为关键。

证书类型标签(CertTag)：指明该证书的类型，该标签使得可信计算密码支撑平台中不同类型证书(EK、PIK、PEK证书)得以有效区分。

可信密码模块信息(TCMModuleInfo)：可信密码模块型号用于标识可信密码模块的类型。包括三个子项分别是：可信密码模块生产商(TCMVender)、可信密码模块型号(TCMMModel)、可信密码模块版本(TCMVersion)。其中后两项TCMMModel、TCMVersion为生产商自定义信息。

可信密码模块生产商(TCMVender)：用于标识可信密码模块的生产商的名称。

可信密码模块型号(TCMMModel)：生产商用以标识所生产TCM的型号，国家密码管理局授予或自己定义。

可信密码模块版本(TCMVersion)：生产商用以标识所生产TCM的固件(Firmware)的版本号。

规范版本号(SpecVersion)：标识实现该可信密码模块时所遵循的“可信计算密码支撑平台技术规范”版本号，形式为“可信计算密码支撑平台技术规范 v1.0.0.0”。

可信密码模块声明(TCMClaim)：此项可以包括有关发布过程安全属性的声明。例如，密码模块密钥生成过程的描述，密码模块密钥是在平台的生产阶段生成的还是在部署阶段生成的。但并不仅限于此。它也可能包括设计和实现该可信密码模块的符合性评估过程。表述内容：“生产阶段生成”或“部署阶段生成”。

附录 C  
(规范性附录)  
主板应用接口

### C. 1 概述

本部分定义了可信密码模块为计算机基本输入输出系统固件层和操作系统层提供服务的接口规范。

### C. 2 UEFI 驱动接口

本部分所定义内容为 TCM 在计算机基本输入输出系统 UEFI 固件对外提供的标准接口,该接口分为两部分:UEFI\_TCM\_PROTOCOL 和 UEFI\_TCMSDRV\_PROTOCOL。UEFI\_TCM\_PROTOCOL 用于提供建立信任链所需接口;UEFI\_TCMSDRV\_PROTOCOL 提供 BIOS 界面或 Shell 环境下对 TCM 操作接口。

#### C. 2. 1 UEFI\_TCM\_PROTOCOL 接口

结合 UEFI 规范,UEFI\_TCM\_PROTOCOL 定义如下,该 PROTOCOL 符合 UEFI 相关规范。

```
GUID #define UEFI_TCM_PROTOCOL_GUID\
{\
0xf541796d,0xa62e,0x4954,0xa7,0x75,0x95,0x84,0xf6,0x1b,0x9c,0xdd\
}

typedef struct
{
    UEFI_TCM_STATUS_CHECK           StatusCheck;
    UEFI_TCM_HASH_ALL               HashAll;
    UEFI_TCM_LOG_EVENT              LogEvent;
    UEFI_TCM_PASS_THROUGH_TO_TCM   PassThroughToTCM;
    UEFI_TCM_HASH_LOG_EXTEND_EVENT HashLogExtendEvent;
} UEFI_TCM_PROTOCOL;
```

**表 UEFI\_TCM\_PROTOCOL 成员描述**

数据	描述
StatusCheck	提供 TCM 信息的服务,反映 TCM 的当前状态
HashAll	提供 hash 操作的服务,生成程序或数据指纹
LogEvent	增加一个 Event Log 的服务
PassThroughToTCM	提供 TCM 的服务
HashLogExtendEvent	提供在数据缓冲器中进行 hash 操作的服务、将 hash 结果放入 TCM PCR 中和增加一个 Event Log 的服务

当调用 ExitBootService(详细规则见 Unified EFI Version 2.3 规范)函数后,启动服务结束。所有在启动服务期间占用的内存都需要释放。当调用 ExitBootService() 函数时,OS 将必须装载自己的 TCM 驱动。

### C.2.1.1 UEFI\_TCM\_STATUS\_CHECK

**功能描述:**

该服务用于获取 TCM 当前状况的信息以及 Event Log 相关内容。

**接口定义:**

```
typedef EFI_STATUS(EFIAPI* UEFI_TCM_STATUS_CHECK)
(
    IN struct_UEFI_TCM_PROTOCOL*           This,
    OUT TCM_BOOT_SERVICE_CAPABILITY*      ProtocolCapability,
    OUT UINT32*                           TCMFeatureFlags,
    OUT UEFI_PHYSICAL_ADDRESS*            EventLogLocation
    OUT UEFI_PHYSICAL_ADDRESS*            EventLogLastEntry
);
```

**参数描述:**

数据	描述
This	指向 UEFI_TCM_PROTOCOL 的指针
ProtocolCapability	改参数用于返回当前 TCM 的状态信息,TCM_UEFI_BOOT_SERVICE_CAPABILITY 结构参见相关定义
TCMFeatureFlags	该指针用于指示 feature
EventLogLocation	指示 Event Log 位置的指针
EventLogLastEntry	如果 Event Log 不止一个 Entry,这个指针将指示内存中最后一个 Entry 的起始地址

### C.2.1.2 UEFI\_TCM\_HASH\_ALL

**功能描述:**

该服务用于提供 HASH 操作。

**接口定义:**

```
typedef EFI_STATUS(EFIAPI* UEFI_TCM_HASH_ALL)
(
    IN struct_UEFI_TCM_PROTOCOL*           This,
    IN UINT8*                            HashData,
    IN UINT64                            HashDataLen,
    IN TCM_ALGORITHM_ID                  AlgorithmId,
    IN OUT UINT64*                       HashedDataLen,
    IN OUT UINT8**                      HashedDataResult
);
```

参数描述：

数据	描述
This	指向 UEFI_TCM_PROTOCOL 的指针
HashData	指向将要被 Hash 的数据缓冲区的指针
HashDataLen	用于描述将要被 Hash 的数据缓冲区的长度
AlgorithmId	用于标示 Hash 算法
HashedDataLen	Hash 结果的长度
HashedDataResult	指向 Hash 结果的指针

### C. 2. 1. 3 UEFI\_TCM\_LOG\_EVENT

功能描述：

该服务用于提供 Log Event 性能的操作，并将结果添加到 Event Log 的入口。

接口定义：

```
typedef EFI_STATUS(EFIAPI* UEFI_TCM_LOG_EVENT)
(
    IN struct_UEFI_TCM_PROTOCOL*           This,
    IN TCM_PCR_EVENT*                     TCMLogData,
    IN OUT UINT32*                        EventNumber,
    IN UINT32                             Flags
);
```

参数描述：

数据	描述
This	指向 UEFI_TCM_PROTOCOL 的指针
TCMLogData	指向内存中 TCM_PCR_EVENT 数据结构的首地址的指针
EventNumber	用于描述刚刚被 Log 的 Event 的数目
Flags	标志位, 目前只有 0x01 Flag 已经使用, 表示其它操作不被执行, 其它的位暂时保留

### C. 2. 1. 4 UEFI\_TCM\_PASS\_THROUGH\_TO\_TCM

功能描述：

该调用用于以字节流形式向 TCM 发送命令, 同时负责返回 TCM 以字节形式的结果。

接口定义：

```
typedef EFI_STATUS(EFIAPI* UEFI_TCM_PASS_THROUGH_TO_TCM)
(
    IN struct_UEFI_TCM_PROTOCOL*           This,
    IN UINT32                            TCMInputParameterBlockSize,
    IN UINT8*                            TCMInputParameterBlock,
    IN UINT32                            TCMOutputParameterBlockSize,
    IN UINT8*                            TCMOutputParameterBlock
);
```

参数说明：

数据	描述
This	指向 UEFI_TCM_PROTOCOL 的指针
TCMInputParameterBlockSize	TCM 输入参数块的大小
TCMInputParameterBlock	指向 TCM 输入参数块的指针
TCMOutputParameterBlockSize	TCM 输出参数块的大小
TCMOutputParameterBlock	指向 TCM 输出参数块的指针

### C. 2. 1. 5 UEFI\_TCM\_HASH\_LOG\_EXTEND\_EVENT

功能描述：

该函数用于提供 Hash 以及 Extends Event 的功能。

接口定义：

```
typedef EFI_STATUS(EFIAPI* UEFI_TCM_HASH_LOG_EXTEND_EVENT)
(
    IN struct_UEFI_TCM_PROTOCOL*           This,
    IN UEFI_PHYSICAL_ADDRESS               HashData,
    IN UINT64                             HashDataLen,
    IN TCM_ALGORITHM_ID                   AlgorithmId,
    IN OUT TCM_PCR_EVENT*                TCMLogData,
    IN OUT UINT32*                       EventNumber,
    OUT UEFI_PHYSICAL_ADDRESS*            EventLogLastEntry
);
```

参数描述：

数据	描述
This	指向 UEFI_TCM_PROTOCOL 的指针
HashData	指向将要被 Hash 的数据缓冲区的指针
HashDataLen	用于描述将要被 Hash 的数据缓冲区的长度
AlgorithmId	用于标示 Hash 算法
TCMLogData	指向物理地址中包含 TCM_PCR_EVENT 结构起始位置的指针
EventNumber	用于描述刚刚被 Log 的 Event 的数目
EventLogLastEntry	指向物理地址中刚刚放置的 Event Log 首字节的指针

### C. 2. 2 UEFI\_TCMSDRV\_PROTOCOL 接口

本部分是为了扩展应用开发而专门编写的, UEFI\_TCMSDRV\_PROTOCOL 中详细定义了与 TCM 操作相关的诸多函数指针。

使用该 Protocol 可以实现 TCM 驱动与应用分离, 如可以使用该 Protocol 编写 TCM Setup 相关响应, 也可以编写 TCM 在 shell 下的使用工具。UEFI\_TCMSDRV\_PROTOCOL 将常用的 TCM 操作进行了封装, 便于驱动和应用程序的调用, 从而使驱动部分和应用完全的脱离。以便于驱动也应用(如 setup 界面响应)的单独开发, 减少代码耦合。

```

GUID
#define UEFI_TCMSDRV_PROTOCOL_GUID\
{\
    0x483e0283,0x9f20,0x4205,0xae,0x1,0x8,0x1,0x8b,0x0,0xa8,0xb4\
}

```

驱动文件中以 Protocol 的形式提供下列接口给用户调用。

```

typedef struct_UEFI_TCMSDRV_PROTOCOL {
    MP_TCMCOMCALL          Tcm_StartUp_Clear;      // mpTCMSTClr;
    MP_TCMCOMCALL          Tcm_StartUp_State;     // mpTCMSTSs;
    MP_TCMCOMCALL          Tcm_Enable;           // mpTCMEnable;
    MP_TCMCOMCALL          Tcm_Disable;          // mpTCMDisable;
    MP_TCMCOMCALL          Tcm_ClearOwner;       // mpTCMClrOwmer;
    MP_GETSTATUS           Tcm_GetStatus;        // mpTCMGetStatus;
    MP_PASSTHROUGHTOTCM   TcmLibPassThrough;    // Pass through to TCM;
} UEFI_TCMSDRV_PROTOCOL;

```

### C. 2. 2. 1 Tcm\_StartUp\_Clear

功能描述：

正常 POST 状态下,通过执行 Tcm\_StartUp\_Clear 接口进行 StartUp\_Clear 型自检  
接口定义：

```

EFI_STATUS TCMINTERNALAPI
Tcm_StartUp_Clear(
    IN VOID
)

```

参数描述：

无

### C. 2. 2. 2 Tcm\_StartUp\_State

功能描述：

正常 POST 状态下,通过执行 Tcm\_StartUp\_State 接口进行 StartUp\_State 型自检  
接口定义：

```

EFI_STATUS TCMINTERNALAPI
Tcm_StartUp_State(
    IN VOID
)

```

参数描述：

无

### C. 2. 2. 3 Tcm\_Enable

功能描述：

启动 TCM,通过执行 Tcm\_Enable 接口进行 Enable 和 Active 操作  
接口定义：

```

EFI_STATUS TCMINTERNALAPI

```

```
Tcm_Enable(
    IN VOID
)
```

参数描述：

无

#### C. 2. 2. 4 Tcm\_Disable

功能描述：

启动 TCM,通过执行 Tcm\_Disable 接口进行 Deactivate 和 Disable 操作  
接口定义：

```
EFI_STATUS TCMINTERNALAPI
Tcm_Disable(
    IN VOID
)
```

参数描述：

无

#### C. 2. 2. 5 Tcm\_ClearOwner

功能描述：

启动 TCM,通过执行 Tcm\_ClearOwner 接口进行强制清除操作  
接口定义：

```
EFI_STATUS TCMINTERNALAPI
Tcm_ClearOwner(
    IN VOID
)
```

参数描述：

无

#### C. 2. 2. 6 Tcm\_GetStatus

功能描述：

启动 TCM,通过执行 Tcm\_ClearOwner 接口进行强制清除操作  
接口定义：

```
EFI_STATUS TCMINTERNALAPI
Tcm_GetStatus (
    IN OUT UINT8*      EnabledSts,
    IN OUT UINT8*      ActivedSts,
    IN OUT UINT8*      OwnerSts
)
```

参数描述：

数据	描述
EnabledSts	是否处于 Enable 状态,等于 1 表示 TCM 处于 enable 状态
ActivedSts	是否处于 Active 状态,等于 1 表示 TCM 已激活
OwnerSts	Owner 是否安装,等于 1 表示 Owner 已安装

### C. 2. 2. 7 TcmLibPassThrough

功能描述：

以二进制流形式向 TCM 发送命令字，并获取 TCM 的响应。

接口定义：

```
EFI_STATUS TCMINTERNALAPI
TcmLibPassThrough (
    IN     TIS_PC_REGISTERS_PTR     TisReg,
    IN     UINT8*                  TcmInputParameterBlock,
    IN     UINT32                  TcmInputParameterBlockSize,
    IN     UINT8*                  TcmOutputParameterBlock,
    IN     UINT32                  TcmOutputParameterBlockSize
)
```

参数描述：

数据	描述
TisReg	TCM 内存映射地址
TcmInputParameterBlock	输入二进制流首地址
TcmInputParameterBlockSize	输入二进制流长度
TcmOutputParameterBlock	输出二进制流首地址
TcmOutputParameterBlockSize	输出二进制流长度

函数返回值列表

返回码	描述
EFI_SUCCESS	函数成功返回
EFI_TIMEOUT	TCM t 传输超时
EFI_WARN_BUFFER_TOO_SMALL	TCM 受到的数据比 buffer 大
EFI_DEVICE_ERROR	设备错误
EFI_INVALID_PARAMETER	TCM 命令流错误
EFI_NO_RESPONSE	TCM 无响应
EFI_PROTOCOL_ERROR	Protocol 安装失败

### C. 3 UEFI 事件日志

Event Log 是对平台程序或者数据记录作指纹的 PCR 记录时，描述详细的用于记录相关信息的数据结构。在授权方的可信度验证过程中需要使用 Event Log 数据结构(TCM\_PCR\_EVENT)。

```
typedef struct
{
    TCM_PCRINDEX      PCRIndex;
    TCM_EVENTTYPE     EventType;
    TCM_DIGEST        Digest;
```

```

    UINT32          EventSize;
    UINT8           Event [1];
} TCM_PCR_EVENT;

```

其中 TCM\_PCR\_EVENT. EventType 所需的事件类型如下表所示：

UEFI 事件类型列表

事件类型	值	描述
EV_POST_CODE	0x01	TCM_PCR_EVENT. PCRIndex=0 TCM_PCR_EVENT. digest=TBB 的杂凑结果 TCM_PCR_EVENT. Event[1]=EFI_PLATFORM_FIRMWARE_BLOB 结构
EV_SEPARATOR	0x04	TCM_PCR_EVENT. PCRIndex=任意 PCR[0-7] TCM_PCR_EVENT. digest=SM3 杂凑 0x00000000 TCM_PCR_EVENT. Event[1]=0x00000000
EV_S_TBB_VERSION	0x08	TCM_PCR_EVENT. PCRIndex=0 TCM_PCR_EVENT. digest=TBB 的版本字符串的杂凑结果 TCM_PCR_EVENT. Event[1]=TBB 的版本字符串
EV_S_TBB_CONTENTS	0x07	TCM_PCR_EVENT. PCRIndex=0 TCM_PCR_EVENT. digest=TBB 的版本相关内容的杂凑结果 TCM_PCR_EVENT. Event[1]=EFI_PLATFORM_FIRMWARE_BLOB 结构
EV_EFI_EVENT_BASE	0x80000000	后续 EFI 平台相关的事件的基准值
EV_EFI_VARIABLE_DRIVER_CONFIG	EV_EFI_EVENT_BASE + 0x1	TCM_PCR_EVENT. PCRIndex=1,3 或者 5 TCM_PCR_EVENT. digest=变量, GUID, Unicode 字符串的杂凑结果 TCM_PCR_EVENT. Event[1]=EFI_VARIABLE_DATA
EV_EFI_VARIABLE_BOOT	EV_EFI_EVENT_BASE + 0x2	TCM_PCR_EVENT. PCRIndex=5 TCM_PCR_EVENT. digest=EFI 引导服务相关的变量 TCM_PCR_EVENT. Event[1]=EFI_VARIABLE_DATA
EV_EFI_BOOT_SERVICES_APPLICATION	EV_EFI_EVENT_BASE + 0x3	TCM_PCR_EVENT. PCRIndex=2,4 TCM_PCR_EVENT. digest=EFI 引导时服务的应用程序的杂凑结果 TCM_PCR_EVENT. Event[1]=EFI_IMAGE_LOAD_EVENT
EV_EFI_BOOT_SERVICES_DRIVER	EV_EFI_EVENT_BASE + 0x4	TCM_PCR_EVENT. PCRIndex=0,2 TCM_PCR_EVENT. digest=EFI 引导时服务的驱动程序的杂凑结果 TCM_PCR_EVENT. Event[1]=EFI_IMAGE_LOAD_EVENT

表 (续)

事件类型	值	描述
EV_EFI_RUNTIME_SERVICES_DRIVER	EV_EFI_EVENT_BASE +0x5	TCM_PCR_EVENT. PCRIndex=0,2 TCM_PCR_EVENT. digest=EFI运行时服务的驱动程序的杂凑结果 TCM_PCR_EVENT. Event[1]=EFI_IMAGE_LOAD_EVENT
EV_EFI_GPT_EVENT	EV_EFI_EVENT_BASE +0x6	TCM_PCR_EVENT. PCRIndex=5 TCM_PCR_EVENT. digest=GPT 表的杂凑结果 TCM_PCR_EVENT. Event[1]=EFI_GPT_DATA
EV_EFI_ACTION	EV_EFI_EVENT_BASE +0x7	TCM_PCR_EVENT. PCRIndex=4,5 TCM_PCR_EVENT. digest=Event[1]项中字符串的杂凑结果 TCM_PCR_EVENT. Event[1]=EV_EFI_ACTION 字符串
EV_EFI_PLATFORM_FIRMWARE_BLOB	EV_EFI_EVENT_BASE +0x8	TCM_PCR_EVENT. PCRIndex=>1 TCM_PCR_EVENT. digest=EFI_PLATFORM_FIRMWARE_BLOB 的杂凑结果 TCM_PCR_EVENT. Event[1]=EFI_PLATFORM_FIRMWARE_BLOB

## 参 考 文 献

<http://www.trustedcomputinggroup.org/home>:

- [1] TCG TPM Specification Version 1.2 Revision 94
  - [2] TCG Software Stack (TSS) Specifications version 1.2
  - [3] TCG PC Specific Implementation Specification Version 1.1
  - [4] TCG PC Client Specific TPM Interface Specification (TIS) Version 1.2 FINAL Revision 1.00
  - [5] TCG Specification Architecture Overview Specification Revision 1.4
  - [6] FIPS PUB 198, The Keyed-Hash Message Authentication Code (HMAC)
-