

可信密码模块符合性测试方法研究

李 昊^{1),2)} 胡 浩^{1),2),3)} 陈小峰^{1),2)}

¹⁾(中国科学院软件研究所信息安全国家重点实验室 北京 100190)

²⁾(信息安全共性技术国家工程研究中心 北京 100190)

³⁾(中国科学技术大学电子工程与信息科学系 合肥 230027)

摘 要 提出了一种可信密码模块(TCM)符合性测试的形式化方法,采用基于扩展有限状态机(EFSM)模型与测试向量相结合的方式对 TCM 的标准进行形式化建模.由于该建模方法结合 TCM 自身特点给出了命令依赖关系图的获取算法以及 EFSM 模型与测试向量获取算法,所以能够更好地对标准进行形式化建模,并用于测试用例的产生.通过测试结果分析以及与其他相关工作的对比,表明该方法能够有效地产生测试用例,并提高 TCM 符合性测试的错误检测率.

关键词 可信计算;TCM;形式化方法;符合性测试;EFSM

中图法分类号 TP309 **DOI号**: 10.3724/SP.J.1016.2009.00654

Research on Compliant Testing Method of Trusted Cryptography Module

LI Hao^{1),2)} HU Hao^{1),2),3)} CHEN Xiao-Feng^{1),2)}

¹⁾(State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences, Beijing 100190)

²⁾(National Engineering Research Center of Information Security, Beijing 100190)

³⁾(Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei 230027)

Abstract A formal method for compliant test of Trusted Cryptography Module (TCM) is proposed in this paper, which uses EFSM and testing vectors to build the formal model of TCM's standard. Since the modeling method presents the arithmetic for getting the dependency graph of the TCM's commands, the process of building EFSM and the way of getting testing vectors, the test case set can be built effectively. Through the experiments using the new method to do the compliant test of TCM and contrast to other related works, the results show that the proposed method in this paper is more effective in getting test cases and can bring better fault inspection rate.

Keywords trusted computing; TCM; formal method; compliant test; EFSM

1 引 言

可信计算最早是 TCG (Trusted Computing Group)组织提出的系统安全解决方案,它的核心思想是先建立信任根,然后通过信任链的构建将信任

延伸到操作系统层乃至应用层,从而构建起可信计算平台,保护其上的敏感信息. TPM (Trusted Platform Module,可信平台模块)或者 TCM (Trusted Cryptography Module,可信密码模块)等可信芯片是信任链的根,是构建可信平台时信任传递的基础.整个可信平台的安全性都依赖于可信芯片

TPM/TCM 的安全与可靠. 因此, 保证可信芯片的安全性是非常重要的.

为了规范可信计算平台的发展, 国内外都出台了可信芯片的相关标准, 包括国际上 TCG 组织发布的《TPM Main Specification Version 1.2》^[1] 以及国内发布的《可信计算密码支撑平台功能与接口规范》^[2]. 这些标准可以有效地指导并规范厂商对可信芯片的具体实现. 但是假设厂商的可信芯片产品都自觉地符合这些规范是不可靠的, 因此必须对可信芯片实施符合性测试来检验其对规范的符合程度.

可信芯片符合性测试是用于度量可信芯片产品与相关标准符合程度的. 该测试对检验可信芯片的功能性和互操作性等产品的情况至关重要, 是通用准则 (Common Criteria, CC) 评估的安全功能测试所不能取代的. CC 评估的安全功能测试是对评估对象的安全功能实现情况的检测, 而符合性测试则关注于产品对标准的符合程度的检测. 因为可信芯片标准中对产品的规范不仅仅限于安全功能, 所以可信芯片的符合性测试与 CC 评估两者是无法相互取代的, 它们共同为用户提供使用可信芯片的信心.

目前, CC 评估技术已经相对成熟, 并且对可信芯片的 CC 测评工作也已经相继展开^[3-4]. 然而, 可信芯片的符合性测试由于缺少有效的测试框架与测试方法, 所以进展缓慢. 本文将针对 TCM 符合性测试面临的问题以及 TCM 自身特点, 提出一种形式化测试方法来解决这些问题, 从而对 TCM 实施有效的符合性测试, 给出产品的标准符合程度量化结果. 由于国际上 TCG 组织提出的可信芯片 TPM 与国内的可信芯片 TCM 在结构上是相似的, 因此本文的 TCM 符合性测试方法同样适用于 TPM, 然而限于篇幅, 本文将主要以 TCM 的符合性测试为例进行论述.

本文第 2 节介绍国内外对可信芯片 (TPM/TCM) 符合性测试的研究现状; 第 3 节给出本文所采用的测试框架; 第 4 节给出具体的形式化建模方法; 第 5 节展示形式化模型的用例生成工作; 第 6 节介绍基于本文提出的符合性测试框架与形式化方法的测试实验, 并通过结果分析与方案对比表明所提出的框架与形式化方法的有效性; 最后总结全文并简要说明未来的工作.

2 相关工作

目前, 国内外可信芯片的符合性测试工作并不

是很多, 但是已经取得了一定成果^[5-7]. 这些研究都发现了可信芯片存在的一些问题, 并一致认为对可信芯片进行符合性测试是非常必要的.

Ahmad-Reza 等对 TPM 的核心功能部分进行了符合性测试, 他们根据 TPM 的特点对测试进行了分层, 然后根据 TPM 命令之间的依赖关系进行测试用例的编写^[5]. 他们指出, 在可信芯片的符合性测试中, 困难问题就是如何从非形式化的规范中获得形式化模型, 并利用形式化模型进行符合性测试^[5]. 文献[5]通过对 TPM 的核心功能进行符合性测试, 表明了符合性测试的重要性, 但是没有解决可信芯片标准的形式化建模问题.

在可信芯片的形式化建模方面, 文献[6]概括性地提出了基于有限状态机 (FSM) 的符合性测试模型, 定义了 TPM 符合性测试中的状态和转移, 提出了通过对比 TPM 产品的 FSM 模型以及 TPM 标准的 FSM 模型的输入和输出来检验 TPM 芯片对规范的符合情况^[6]. 但是该研究缺乏具体的符合性测试工作的支持, 同时也没有给出从非形式化的 TPM 规范获取 FSM 模型的方法.

我们曾提出过一种 TCM 芯片的符合性测试的框架和方法^[7], 在该框架下能够对 TCM 符合性测试进行分层, 不同层实施不同种类的测试, 并关注不同的测试指标, 最后将测试结果分散到 4 个符合度指标上, 该工作在一定程度上实现了 TCM 符合性测试结果的量化. 然而文献[7]虽然指出了在功能层可以采用扩展有限状态机 (EFSM) 模型, 在命令层可以采用测试向量等形式化建模方法, 但是如何根据 TCM 特点从非形式化的规范中抽象出形式化模型是我们未解决的问题. 本文期望在此框架和方法的基础上, 提出一种形式化建模方法来获取 TCM 的 EFSM 模型及测试向量集合, 从而提高符合性测试的错误检测率, 并保证符合性测试结果的可信度.

3 符合性测试框架

可信密码模块 TCM 是一种接收软件层命令, 完成指定功能, 然后输出结果的硬件模块. 从单条命令的执行来看, TCM 像一个黑盒, 根据输入计算输出. 我们可以认为它是一个有着清晰规范接口的软件模块, 对其每条命令都可以根据规范实施符合性测试. 然而还有一些 TCM 提供的功能需要多条命令的顺序执行才能完成, 我们对这种安全功能的完成情况也要进行测试. 因此, 我们将 TCM 的符合性

测试分为了两层:命令层与功能层^[7].

命令层进行的是单条命令的符合性测试,所关注的符合度指标为可靠性与鲁棒性.可靠性指标衡量的是 TCM 接收到合法输入时,其输出对标准的符合程度;鲁棒性指标衡量的是 TCM 接收到非法输入时,其输出对标准的符合程度.根据命令接口测试的特点,首先为每条命令构建测试向量,这些测试向量是由影响该命令执行的变量组成,接着本文将确定这些变量的取值范围,最后采用等价类划分法从测试向量生成测试用例.其中有效等价类获得的测试用例针对的是可靠性指标,而无效等价类获得的测试用例针对的是鲁棒性指标.

功能层测试检测的是多条命令组合执行,其执行结果对标准的符合程度,主要分为两种测试:功能性测试与安全性测试.功能性测试针对的是标准中提出的对 TCM 产品的功能要求,而安全性测试针对的是标准中提出的安全要求,例如防字典攻击等.本文将为 TCM 相关规范建立 EFSM 模型,然后按照 Bourhfir 提出的算法^[8]进行测试用例的提取,最后实施测试.同时,我们认为本文的安全性测试与 CC 评估中的安全功能测试有部分重合,安全性测试可以作为 CC 评估的安全功能测试的补充,为其单独编写测试用例.而且安全性测试在文献^[5]中也有详细论述,因此不在本文的讨论范围内.具体的 TCM 符合性测试框架如图 1 所示.

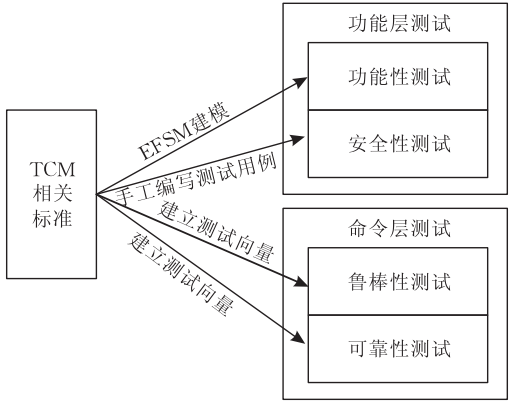


图 1 TCM 符合性测试框架

4 形式化建模

4.1 命令依赖关系图

TCM 的命令之间存在着依赖关系,使得这些命令的调用需要遵循一定的顺序.功能性、可靠性与鲁棒性的测试中,形式化建模都需要考虑这种依赖

关系.

虽然 Ahmad-Reza 等人的测试工作中也运用了这种依赖关系^[5],但是其依赖关系图的建立完全依赖于符合性测试人员对可信芯片 TPM 的了解与熟悉程度.然而由于 TPM 的复杂性,其命令之间的依赖关系也是错综复杂的,所以完全凭借测试人员的经验去获取这种依赖关系,必然导致获取到的依赖关系是不正确或不完整的.同样,对于可信密码模块 TCM 来说,其命令之间的依赖关系也是错综复杂的,不能完全凭借测试人员的经验进行获取,因此本文根据 TCM 具有的功能模块化,并且模块之间相互依赖,命令之间相互依赖的特点,提出一种按照命令之间控制流与数据流上的关系来获得依赖关系的方法.

我们认为,命令之间的依赖关系主要包括:输入参数中数据的依赖以及命令执行顺序上的依赖.因此本文将命令间的依赖关系分为了控制流依赖关系与数据流依赖关系,具体定义如下.

定义 1. 如果命令 A 的执行需要命令 B 先执行,并且命令 A 的输入参数不包含命令 B 的输出,那么称命令 A 在控制流上依赖于命令 B,这种依赖关系称为控制流依赖关系.

定义 2. 如果命令 A 的执行需要命令 B 先执行,并且命令 A 的输入参数包含命令 B 的输出,那么称命令 A 在数据流上依赖于命令 B,这种依赖关系称为数据流依赖关系.

在定义 1 与定义 2 的基础上,我们给出 TCM 命令间依赖关系的获取过程如下:

1. 因为 TCM 功能的模块化特点,所以可按照规范将命令划分为多个子模块.每个子模块由若干条 TCM 命令组成,完成 TCM 的特定功能.在本文中 TCM 被分为 23 个子模块:启动子模块、状态处理子模块、自检子模块、工作模式管理子模块、Owner 管理子模块等(与《可信计算密码支撑平台功能与接口规范》^[2]中的划分一致),如图 2 所示.

2. 依照规范,对于子模块执行顺序进行排序.这些子模块的执行是有依赖关系的,一些模块的运行必须要求某些其他模块先运行.子模块 A 对其他子模块的依赖关系可以根据 A 中的命令对其他模块中命令的依赖很容易地获得.具体的获取方法为:若任意的命令 $x \in A, y \in B$, 且 x 依赖于 y , 则模块 A 依赖于模块 B.譬如,密钥管理子模块中 TCM_CreateWrapKey 命令的执行依赖于 Owner 管理子模块中 TCM_TakeOwnership 命令的成功执行,因此密钥管理子模块就依赖于 Owner 管理子模块.图 2 为按照此方法获得的 TCM 功能子模块之间的依赖关系.

3. 在每个子模块内部,获取子模块命令依赖关系图.本

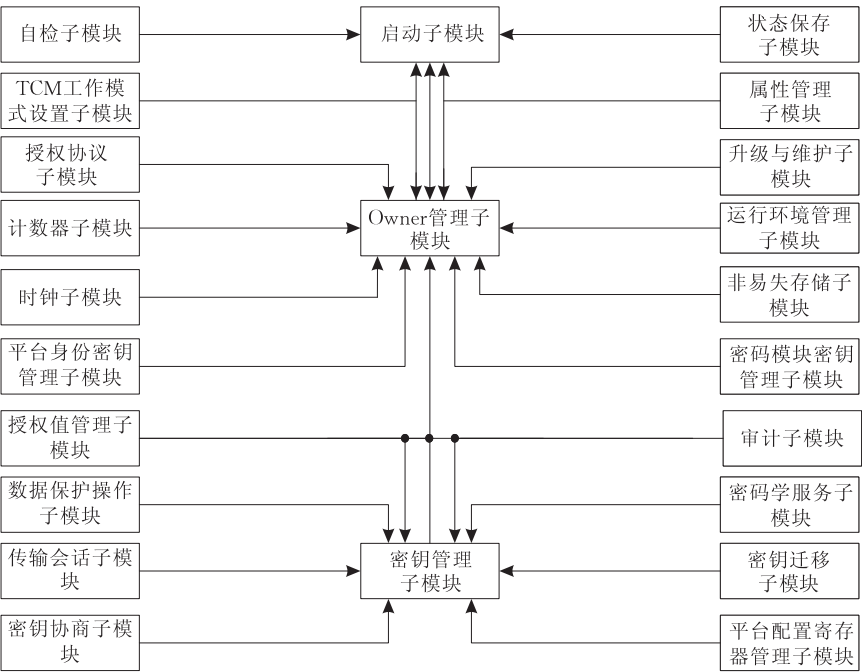


图 2 TCM 功能子模块之间依赖关系

文将以 Owner 管理子模块命令依赖关系以及密钥管理子模块命令依赖关系为例论述这一过程。

Owner 管理子模块共有 5 条 TCM 命令，它们的依赖关系如图 3 所示。

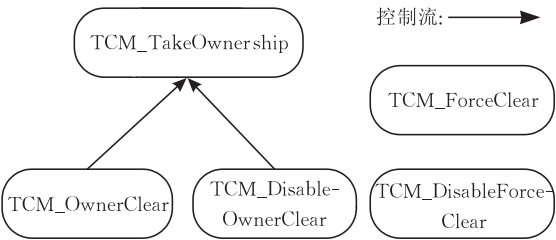


图 3 Owner 管理子模块命令依赖关系图

其中，TCM_OwnerClear 以及 TCM_DisableOwnerClear 依赖于 TCM_TakeOwnership 命令的成功执行，所以画出它们之间的控制流依赖关系，而 TCM_DisableOwnerClear 命令的执行，只是影响到 TCM_OwnerClear 的执行结果，它们都受到 Owner 是否存在这一环境的影响，但它们之间并不存在依赖关系。

密钥管理子模块也有 5 条 TCM 命令，它们之间的依赖关系如图 4 所示。

因为 TCM_LoadKey 命令的执行需要 TCM_CreateWrapKey 命令的输出作为参数，所以它们之间存在数据流依赖关系，但是却不存在控制流的依赖关系，这是由于 TCM_LoadKey 的执行不一定需要 TCM_CreateWrapKey 命令作为前驱，它们之间不存在这种执行顺序的依赖。同理，TCM_WrapKey 与 TCM_LoadKey 之间存在数据流依赖关系。而 TCM_GetPubKey 的执行不但需要 TCM_LoadKey 所载入的密钥句柄作为输入参数，并且该命令的执行必须要求密钥对象首先被载入 TCM，因此 TCM_GetPubKey 在数据流与

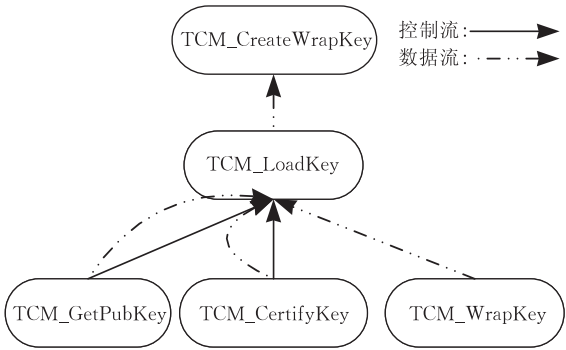


图 4 密钥管理子模块命令依赖关系图

控制流上都依赖于 TCM_LoadKey 命令，同理，TCM_CertifyKey 也同时在数据流与控制流上依赖于 TCM_LoadKey。

4. 将获取到的各子模块依赖关系复合以得到完整的 TCM 命令间依赖关系图。复合的过程如算法 1 所示。

算法 1. 依赖关系复合算法。

```
CompDeps() {
    输入：子模块集合  $G = \{M_1, M_2, \dots\}$ ，子模块间的依赖关系集合  $D = \{MD_1, MD_2, \dots\}$ 
    输出：复合后所有命令间的依赖关系集合  $\Omega$ 
    for each  $MD_i \in D$  do {
        // 获取依赖关系  $MD_i$  的被依赖模块  $M_s$ 
         $M_s = \text{GetStartModule}(MD_i)$ ;
        // 获取依赖关系  $MD_i$  的依赖模块  $M_e$ 
         $M_e = \text{GetEndModule}(MD_i)$ ;
        // 获取模块  $M_s$  的内部命令集合  $\omega_s$ 
         $\omega_s = \text{GetComs}(M_s)$ ;
        // 获取模块  $M_e$  的内部命令集合  $\omega_e$ 
         $\omega_e = \text{GetComs}(M_e)$ ;
    }
```

```
for each  $C_j \in \omega_c$  do{
    for each  $C_k \in \omega_s$  do{
        //若  $C_j$  与  $C_k$  存在依赖关系  $D_{i,k}$ ,
        //则将其加入  $\Omega$ 
        if (CheckDep ( $C_j, C_k$ ))
            AddDep ( $\Omega, D_{i,k}$ );
    }//end for
} //end for
//将模块  $M_i$  的内部命令依赖关系集合  $\omega_i$  添加到  $\Omega$  中
 $\Omega += \omega_i$ 
```

```
} //end for
return  $\Omega$ ;
```

此时获取的依赖关系集合 Ω 存在冗余,这是由于控制流依赖关系的传递性质造成的.也就是在控制流上,命令 A 依赖于命令 B ,而命令 B 依赖于命令 C ,那么命令 A 对命令 C 的依赖关系就是冗余的,可以从 Ω 中删除.

由于篇幅所限,本文仅以 Owner 管理子模块与密钥管理子模块为例展示依赖关系的获取结果.按照算法 1,并去除冗余的控制流依赖关系后,Owner 管理子模块与密钥管理子模块的复合结果如图 5 所示.

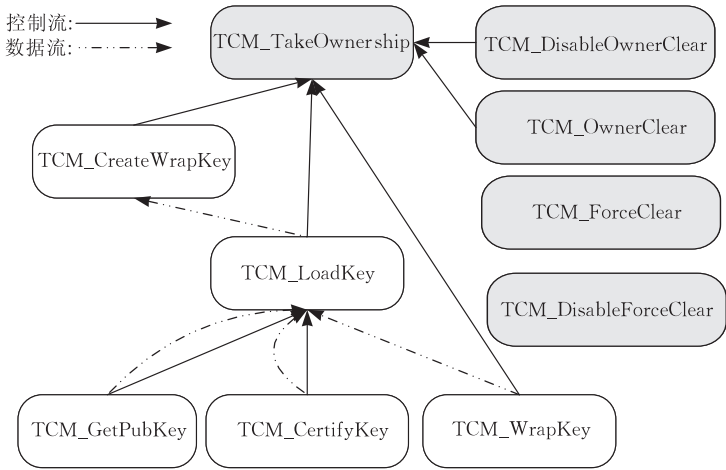


图 5 复合后命令依赖关系图

4.2 EFSM 建模

在本文的测试框架中,功能性测试用例是从 TCM 相关标准的 EFSM 模型中获取的.下面将首先给出 EFSM 模型的定义,然后给出建立 TCM 相关标准的 EFSM 模型的算法.

定义 3. 用六元组来表示一个扩展的有限状态机 EFSM.记为 $M=\langle S,s_0,I,O,T,V\rangle$.其中 M 是 EFSM 的名称. S 是一个非空的状态集合, s_0 是初始状态, I 是一个非空的输入消息集合, O 是一个非空的输出消息集合, V 是变量集合, T 是状态转移集合.对于任意的 $t \in T$, t 是一个六元组 (s,x,P,op,y,s') ,其中 $s,s' \in S$,分别是初始状态和终止状态; $x \in I$,是状态迁移 t 的输入; $y \in O$ 是状态迁移 t 的输出; P 是状态迁移 t 的前置条件,可能为空; op 是状态迁移中的操作,由一系列的输出语句和变量赋值语句组成.

因为复合后的依赖关系图中变量的数目非常多,造成从复合后的依赖关系图直接获取 EFSM 模型图时,状态空间异常庞大,所以状态的提取是困难的.因此本文基于 TCM 模块化的特点首先对每个

功能子模块进行 EFSM 建模,最后进行复合形成整体的 EFSM 模型.其中,每个功能子模块的 EFSM 模型建立过程如下.

1. 确定该模块所涉及变量的集合 V .
2. 根据该模块内部的命令依赖关系,获得命令的执行序列集合.
3. 为命令执行序列集合中的每条序列建立 EFSM 模型.观察每条命令执行前后的 V 与 V' ,用来确定状态转移 t .如果命令的执行产生了新的 V ,则增加新的状态到 EFSM 中.
4. 观察每条命令序列对应的 EFSM 模型,将相同的状态与状态转移进行合并.

在建立了每个功能子模块的 EFSM 模型后,本文给出这些子模块的 EFSM 模型复合过程.

假设 TCM 的功能子模块 A 与 B ,其依赖关系为 B 依赖于 A ,EFSM 模型分别为 EFSMA 与 EFSMB,则它们的复合过程为

1. 将 EFSMA 与 EFSMB 的变量集合 V_A 与 V_B 进行合并.
2. 从 A 与 B 复合后的依赖关系图中进一步去除数据流依赖关系造成的依赖冗余.由于数据流依赖关系对于命令层测试非常重要,因此这种由于数据流依赖关系造成的依赖

冗余并没有在合成依赖关系图时去除,但这种冗余对 EFSM 的复合是没有意义的,因此根据传递依赖的性质,需要去除数据流上的冗余依赖关系集合 R 。

3. 获得影响 EFSM 复合的依赖关系集合 $\xi = \Omega - R - S$ 。其中 Ω 为算法 1 得到的依赖关系集合, R 为步 2 需要去除的冗余的数据流依赖关系集合, S 为 A, B 各自内部的依赖关系组成的集合。

4. 检查 ξ 中的每条依赖关系。若命令 x 是子模块 A 的内部命令, y 是子模块 B 的内部命令,且 x 与 y 存在依赖关系 $d \in \xi$,则检查 EFSMA 中 x 执行后进入的所有状态与

EFSMB 中 y 执行的所有前驱状态是否能够复合,如果这些状态中,变量 V 的取值没有冲突,则将它们复合为一个状态。

由于篇幅所限,本文仅以 Owner 管理子模块及密钥管理子模块为例展示从依赖关系图获取 EFSM 模型的结果。由 Owner 管理子模块与密钥管理子模块命令依赖关系图,并根据本文给出的建立子模块 EFSM 模型的方法,可得它们的 EFSM 模型,如图 6、图 7 所示。

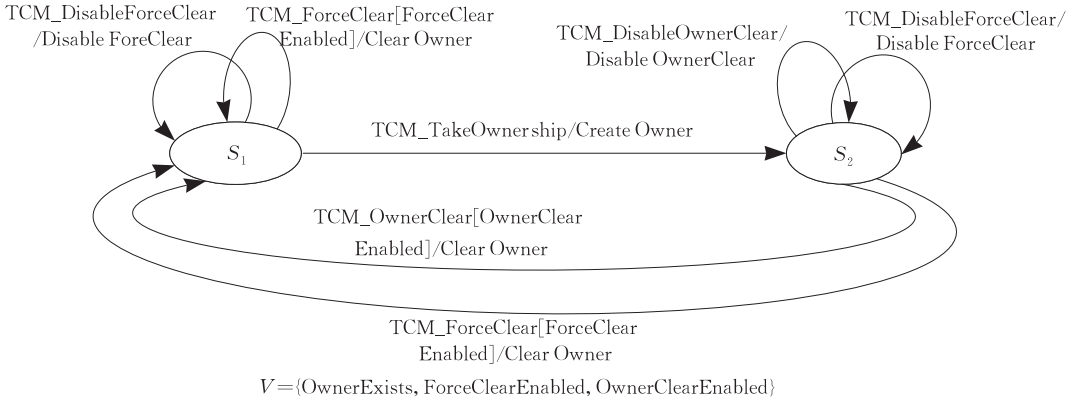


图 6 Owner 管理子模块 EFSM 图

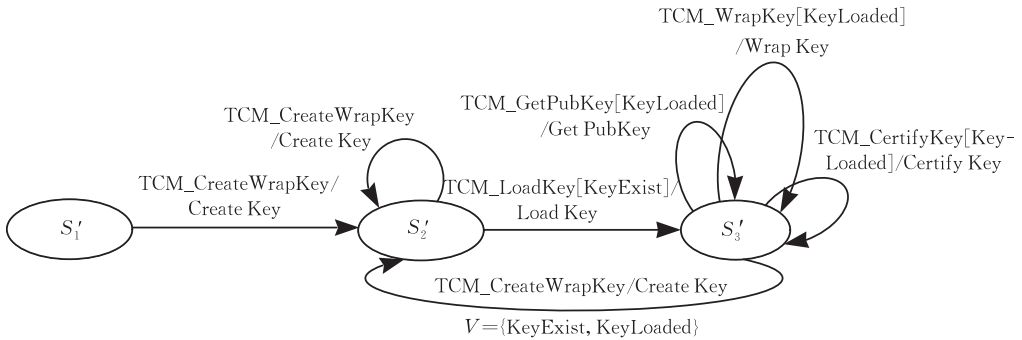


图 7 密钥管理子模块 EFSM 图

图中状态转移的格式为:状态转移 t [前置条件 P] / 迁移中的操作 op 。比如,图 6 的模型共有两个状态: S_1 表示无 Owner 状态, S_2 表示有 Owner 状态。状态转移共有 7 个。TCM_ForceClear [ForceClear Enabled] / Clear Owner 表示当模型处于无 Owner 状态时,如果 ForceClear 是 Enable 的,那么就可以执行状态转移事件 TCM_ForceClear,目的状态仍然为 S_1 ,并且要完成 Clear Owner 的操作。

建立这两个模块各自的 EFSM 模型后,根据本文给出的 EFSM 模型复合过程,将它们进行复合,可以得到复合后的 EFSM 模型,如图 8 所示。

4.3 获取测试向量

在本文的符合性测试框架下,可靠性与鲁棒性测试都是通过测试向量进行的。测试向量是由命令

的输入参数与命令执行环境共同构成的。命令的运行环境是指影响该命令运行的环境变量集合,属于隐式的输入参数。根据 4.1 节获得的命令依赖关系图,可以方便地获得测试向量,具体过程如算法 2 所示。

算法 2. 测试向量获取算法。

$GetVectors() \{$

输入:命令 A 对其他命令的依赖关系集合 $D = \{d_1, d_2, d_3, \dots\}$,命令 A 的输入参数集合 $I = \{i_1, i_2, i_3, \dots\}$

输出:测试向量 V

$V = I$; //将 V 初始化为 A 的输入参数集合 I

for each $d_i \in D$ do {

if ($IsControlDep(d_i)$) {

//若依赖关系 d_i 为控制流依赖关系,则获取 d_i

//所影响的环境变量 c ,并将该变量加入 V

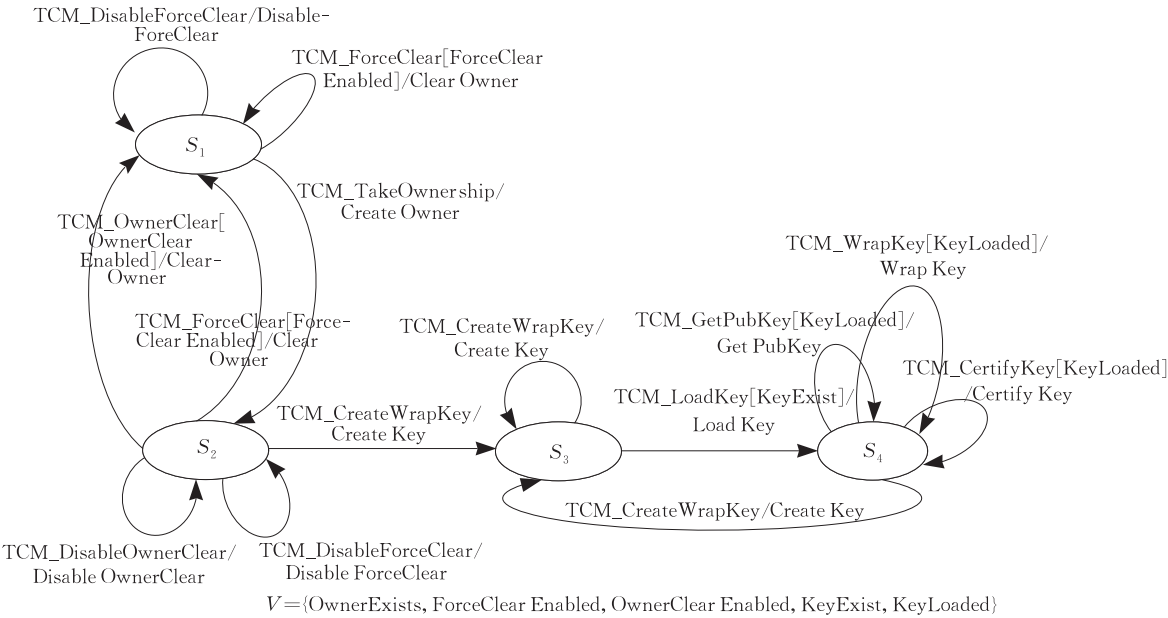


图 8 复合后的 EFSM 图

```
c=FindCV(di);
AddToVector(V,c);
} //end if
} //end for
return V; }
```

根据算法 2 可获得命令 A 的测试向量 $V = \{i_1, i_2, i_3, \dots, c_1, c_2, c_3, \dots\}$, 其中 i_1, i_2, i_3, \dots 表示命令 A 所需的输入参数变量, c_1, c_2, c_3, \dots 表示隐式影响命令 A 执行的环境变量, 两者共同构成测试向量 V.

5 测试用例生成

5.1 功能性测试用例

基于 EFSM 模型的用例自动生成算法已经有不少成果^[8-13]. 本文采用 Bourhfir 提出的算法^[8]进行测试用例的生成. 测试用例的生成分为两步: 第 1 步通过算法生成抽象测试用例, 这些抽象测试用例是不能执行的; 第 2 步将抽象测试用例具体化为可执行的测试用例, 在这一步中需要填入具体的测试数据. 生成测试用例所遵循的原则有两种: (1) 基于需求的用例筛选方法; (2) 随机筛选方法. 基于需求的筛选方法指的是根据用户提供的限制条件对产生的测试用例进行选择, 如用户指定只生成与某个状态相关的用例. 随机筛选方法是指按照一定的概率分布, 筛选出最合适的测试用例, 这种概率分布一般是遵循一定的使用模型 (usage model) 的. 本文采纳的是基于需求的用例筛选方法, 用户可以根据需求, 选取部分 EFSM 模型进行测试用例的生成.

5.2 可靠性与鲁棒性测试用例

从一个测试向量 V 产生测试用例, 可以使用等价类划分法来有效地约减测试空间. 首先将 V 中的每个变量的取值范围划分为若干等价类, 然后按照以下的原则来获取测试用例.

- (1) 设计一个新的测试用例, 使其尽可能多地覆盖尚未覆盖的有效等价类; 重复这一步骤, 直到所有的有效等价类都被覆盖为止;
- (2) 设计一个新的测试用例, 使其仅覆盖一个无效等价类, 重复这一步骤, 直到所有的无效等价类都被覆盖为止.

通过方法 (1) 产生的测试用例即为可靠性测试用例, 测试的是正确的输入参数情况下, TCM 命令的输出是否符合规范. 而方法 (2) 产生的测试用例即为鲁棒性测试用例, 测试的是非法输入下, TCM 命令的输出是否符合规范.

6 实验过程及结果分析

6.1 实验过程

为了验证本文提出的形式化方法的有效性, 我们对 TCM 规范中的 Owner 管理子模块与密钥管理子模块按照上述方法进行了形式化建模, 并生成了测试用例. 最后根据测试用例编写测试脚本, 用于 TCM 产品的符合性测试中.

根据 4.2 节 EFSM 的建模方法以及复合方法, Owner 管理子模块与密钥管理子模块的复合 EFSM

模型如图 8 所示。同时,根据算法 2 中测试向量获取算法,我们从两个子模块中共获得 10 个测试向量,这些测试向量都由输入参数与环境变量组成。

我们对 EFSM 模型采取基于需求的用例筛选法获取测试用例集合,由于这两个子模块复合后的 EFSM 模型中,状态与迁移的数目不大,因此我们选

择对状态与迁移进行全部覆盖的方式。而对测试向量我们采取 5.2 节给出的等价类划分法获取测试用例集合。

最后根据这些测试用例编写测试脚本,并将测试脚本输入测试平台进行实际的 TCM 测试。具体测试的实施流程如图 9 所示。

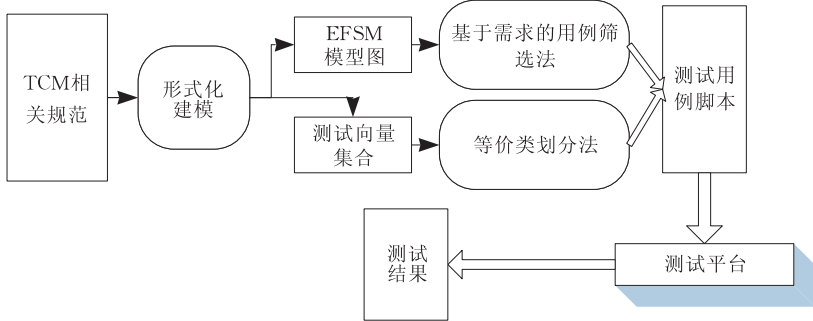


图 9 测试实施流程

6.2 结果分析

我们将通过形式化方法得到的 Owner 管理子模块与密钥管理子模块的测试脚本用于检验市场上两款主流 TCM 产品,并通过测试结果的分析,表明本文形式化方法的有效性。两款 TCM 产品的测试结果如表 1 所示。

表 1 TCM 产品测试通过率

	用例数	产品 1 的通过率/%	产品 2 的通过率/%
功能性测试	92	79.3	64.1
可靠性测试	35	88.6	85.7
鲁棒性测试	89	84.2	70.8

结果表明,两款产品在可靠性与鲁棒性测试中都有较好表现。而在功能性测试中,由于 TCM 规范的复杂性,两款产品的实现与标准不符的问题较多。

为了表明本文提出的形式化建模方法在符合性测试中起的作用,下面将对比形式化建模方法与非形式化建模方法的测试情况。

文献[7]是我们早期的 TCM 符合性测试工作,采用的测试框架与本文相同,但是其功能性测试中 EFSM 模型的建立完全凭借的是测试人员对标准的熟悉程度,没有采用有效的形式化建模方法。而在可靠性与鲁棒性测试中,其测试向量的提取也完全根据经验从命令依赖关系图中获取。

首先比较两种方案产生的有效测试用例数。因为我们采用了相同的测试用例生成算法^[8],所以用例生成的效率以及有效性是相同的。然而由于 EFSM 模型的建立方法不同,导致两种方案产生的有效测试用例数相差很大。并且随着用户选择的进

行测试的功能子模块的增多,这种差距会逐渐增大,如图 10 所示。其原因在于,文献[7]中 EFSM 模型的建立完全凭借经验,并且在逐个建立子模块的 EFSM 模型后,没有进行 EFSM 模型的复合,这就导致了采用基于需求进行用例筛选的算法^[8]提取测试用例时,功能子模块之间的一些有效测试用例被遗漏。而本文提出的从获取依赖关系图到建立 EFSM 模型以及复合 EFSM 模型等方法可以有效地避免符合性测试中功能性测试用例的遗漏,从而更好地发现产品与规范的不符问题。

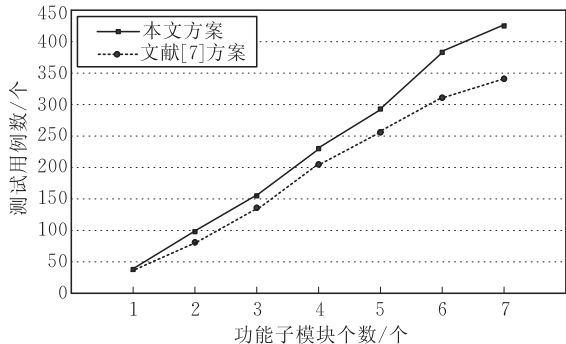


图 10 功能性测试用例数对比图

两种方案在功能性测试用例获取上的差异同样在获取可靠性与鲁棒性测试用例上存在,如表 2 所示。表 2 为 Owner 管理子模块与密钥管理子模块在两种方案下所产生的可靠性与鲁棒性测试用例数的对比。由于本文定义了命令之间控制流与数据流的依赖关系,并给出了测试向量获取算法(算法 2),因此得到的测试向量的变量集合更加完整,从而根据 5.2 节给出的测试用例获取方法,能够得到更多的

有效测试用例.

表 2 可靠性与鲁棒性测试用例数对比		
	本文方案用例数	文献[7]方案用例数
可靠性测试	35	22
鲁棒性测试	89	67

为了证明本文形式化建模方法产生的更多测试用例是有效的,我们比较本文与文献[7]的错误检测率,也就是未通过测试的测试用例数与总测试用例数的比值,如图 11、图 12 所示. 无论在产品 1 还是产品 2 的测试结果中,本文测试方法的错误检测率都比文献[7]高出很多,甚至在产品 2 的功能性测试中高出了近 25%. 可以看出,采用形式化的建模方法能够更加有效地发现产品与规范的不符问题. 因此本文提出的形式化建模方法对 TCM 的符合性测试有着重要意义.

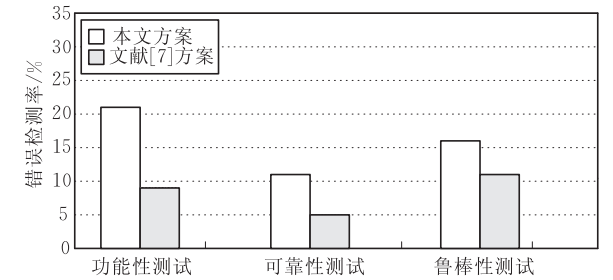


图 11 产品 1 两种方案错误检测率对比

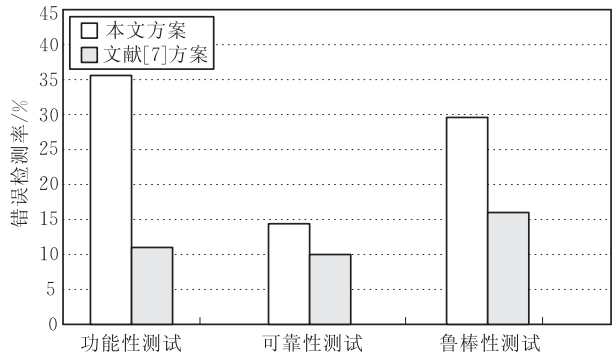


图 12 产品 2 两种方案错误检测率对比

与其他非形式化的可信芯片符合性测试工作^[5]相比,形式化的测试方案使得可信芯片(TPM/TCM)的测试不必花费大量的精力编写测试用例,而只需要关注可信芯片的形式化模型,并且能够通过对测试范围进行有效分析来获得很好的量化测试结果. 同时,形式化的测试方法可以引入自动测试用例生成,避免非形式化方法中手工编写测试用例时人为造成的各种错误. 最后,形式化方法的运用也使得可信芯片的符合性测试具有良好的可扩展性. 当标准进行了升级后,形式化的测试只需要对模型进行相

应的修改,然后自动生成测试用例,而不需繁琐的手动修改或增添测试用例.

7 结论与展望

本文首先给出了 TCM 符合性测试的框架,将测试工作分为命令层与功能层,并将测试划分成了功能性测试、安全性测试、可靠性测试与鲁棒性测试. 接着定义了 TCM 命令间的依赖关系,并给出了依赖关系图的获取算法. 基于 TCM 命令间的依赖关系图,本文又给出了用于功能性测试的 EFSM 模型建立方法及用于可靠性与鲁棒性测试的测试向量获取算法. 然后介绍了本文的测试用例提取方法. 最后通过对两款 TCM 产品的 Owner 管理子模块与密钥管理子模块的符合性测试实验,以及实验结果分析表明本文的形式化建模方法是有效的,并对 TCM 符合性测试具有重要意义.

目前,我们已经有了一个运行测试脚本的半自动化的 TCM 符合性测试平台来支持本文的形式化测试方法. 但是,我们对 TCM 标准的建模仍在进行中. 将来的工作将包含以下方面的内容:(1)测试用例的自动化生成. 测试用例的提取应尽量减少人工的干预,这有利于保证符合性测试的公正性,因此具有重要研究意义;(2)测试结果的可追溯性研究. 测试结果不应该仅仅表现为测试的通过率,我们期望通过测试结果的可追溯性研究,为 TCM 产品的标准符合程度做出更详细而有用的回答.

参 考 文 献

[1] Trusted Computing Group. TPM Main Part 1, Design Principles. Specification Version 1. 2, Revision 62. 2, October, 2003

[2] State Password Administration Committee in China. Functionality and Interface Specification of Cryptographic Support Platform for Trusted Computing. December, 2007 (in Chinese)
(中国国家密码管理局. 可信计算密码支撑平台功能与接口规范. 2007 年 12 月)

[3] Atmel Corporation. AT97SC3201 Security Target. Version 2. 3. 21, February, 2005

[4] Atmel Corporation. Trusted Platform Module AT97SC3201 Summary. Version 1. 2. June, 2005

[5] Ahmad-Reza S, Marcel S, Christian S et al. TCG Inside?: A note on TPM specification compliance//Proceedings of the 1st ACM Workshop on Scalable Trusted Computing. New York: ACM Press, 2006: 47-56

[6] Zhan Jing, Zhang Huan-Guo. Research on TPM based on state machine theory. Journal of Wuhan University, 2008, 33(10): 1067-1069(in Chinese)
(詹静, 张焕国. 基于状态机理论的可信平台模块测试研究. 武汉大学学报, 2008, 33(10): 1067-1069)

[7] Li Hao, Feng Deng-Guo. Compliant testing method of trusted cryptography module. Journal of Wuhan University, 2009, 55(1): 31-34(in Chinese)
(李昊, 冯登国. 可信密码模块符合性测试方法与实施. 武汉大学学报, 2009, 55(1): 31-34)

[8] Bourhfir C, Dssouli R. Automatic executable test case generation for EFSM specified protocols//Proceedings of the 10th IFIP International Workshop on Testing of Communicating Systems. Cheju Island, Korea, 1997: 75-90

[9] Yi Guo-Hong, Lu Yan-Sheng. Equivalence testing based on EFSM. Computer Science, 2007, 34(1): 281-284(in Chi-

nese)
(易国洪, 卢炎生. 基于 EFSM 模型的等价类测试. 计算机科学, 2007, 34(1): 281-284)

[10] Bourhfir C, Dssouli R, Aboulhamid E. Automatic test generation for EFSM-based systems. The Journal of Real-Time Systems, 1989, 1(1): 27-60

[11] Petrenko A, Boroday S, Groz R. Confirming configurations in EFSM testing. IEEE Transactions on Software Engineering, 2004, 30(1): 29-42

[12] Bourhfir C, Aboulhamid E. Test cases selection from SDL specifications. Computer Networks, 2001, 35(6): 693-708

[13] Cheng Kwang-Ting. Automatic generation of functional vectors using the extended finite state machine model. ACM Transactions on Design Automation of Electronic Systems, 1996, 1(1): 57-79



LI Hao, born in 1983, Ph. D. candidate. His major research interests include network and system security, trusted computing.

HU Hao, born in 1981, Ph. D. candidate. His major research interests include network and system security, trusted computing.

CHEN Xiao-Feng, born in 1980, Ph. D. candidate. His major research interests include information and network security.

Background

The work is part of the project “Testing and Evaluating System and Tools for Trusted Computing Platform”, which is supported by the National High Technology Research and Development Program (863 Program) of China under grant No. 2007AA01Z412, and the National Key Technology R&D Program of China under grant No. 2008BAH22B06.

With the development of trusted computing, the test and evaluation for trusted computing platform becomes more and more important recently. The Trusted Cryptography Module (TCM), which is specified and made by China, acts as root of trust and is the basis for all trusted computing services and applications. Hence, the testing and evaluating for TCM is one of the most important researches in trusted computing. The recent researches such as the compliant test of Trusted

Platform Module (TPM) by Ruhr-University Bochum and the compliant test of TCM by Wuhan University present the importance of compliant test to TPM\TCM. However the methods they use in compliant test are not effective and less of persuasion. In this paper, the authors propose a formal method for compliant test of TCM, which uses EFSM and testing vectors to build the formal model of TCM’s standard. Since the modeling method presents the arithmetic for getting the dependency graph of the TCM’s commands, the process of building EFSM and the way of getting testing vectors, the test case set can be built effectively. Finally, the results of the experiments show that the proposed method in this paper is more effective in getting test cases and can bring better fault inspection rate.