ISSN1000-9825 Journal of Software 软件学报 2000,11(3):346~362

# 电子商务安全协议的逻辑验证*

陈庆锋[1,2] 王驹[3] 白硕[1] 张师超[2] 隋立颖[1]

[1](国家智能计算机研究开发中心 北京 100080)

[2](广西师范大学数学与计算机科学系 桂林 541005)

[3](中国科学院软件研究所 北京 100080)

E-mail: bai@ncic.ac.cn

**摘要** 作者在以前所做的工作中,已对电子商务安全(secure electronic transactions,简称SET)中抽取的片段进行了证明,也对SET中可能存在的问题进行了初步探讨.该文在此基础上,对SET的整个业务流程进行了严格的逻辑验证,通过形式化逻辑方法的验证,发现了SET协议中存在的一些问题,并对如何解决这些问题进行了初步的探讨.

**关键词** 信息安全,逻辑验证,防抵赖性,可追踪性.

**中图法分类号** TP309

近年来,越来越多的组织和个人通过国际互联网Internet发送和接收信息,而且据统计数字表明,随着社会经济的发展,它仍在呈上升趋势.对于这些在网上传输的信息,大家都关注着同一个问题——信息安全.针对这一问题,产生了很多保障信息安全的防范措施.

目前世界上比较流行的有关信息安全领域的研究大体可以分为如下几类.首先是比较传统的做法,即用数学手段对加密算法进行改进,达到所需要的目标,它具有严密、不易被攻破的特性,但它同时也比较抽象、难懂,且无法防范通过非法手段获取信息的行为.对于一些金融业务,这是一个不可忽视的环节.其次是一些新科技的应用,例如日本的高科技研究所信息科学系的Kenichi HAYASHI,Eiji OKAMOTO和Masahiro MAMBO[1]根据每个人使用鼠标的习惯,设计了一个模式识别系统,用鼠标就可以辨认出每个人的身份,但它只能作用于访问控制,而无法解决信息在传递过程中的安全问题.美国军方海军实验室的LiWu Chang和Ira S. Moskowitz[2]采用声音隐藏技术把要传输的信息隐藏于宿主信息中通过各种电子媒体如电话线、E-mail、打包的方式传递,根据这一特点,它更适合于通信领域.还有一种在金融上比较广泛研究的Agent技术[3],通过采用一些安全的Agent自动分析信息,并代替雇主进行交易.这些方法各有优点,但它们都要使用额外的外设,在金融业务中,这会增加成本开支,而且一旦设备崩溃,就会造成无法挽回的后果.安全协议的出现,在一定程度上弥补了这些不足之处,但协议本身相当复杂,它是否无懈可击,无法凭直观来检测,因此产生了形式化逻辑验证方法[4,5],其中文献[6]中对NDL逻辑框架的扩展,已经成功地验证了电子商务安全(secure electronic transactions,简称SET)[7]中的部分流程.总的看来,形式化逻辑验证方法以其严谨、简洁的特点在安全协议的验证上起着重要作用.

本文在文献[6,8]工作的基础上,在第1节中对SET协议的整个付费流程进行验证.第2节初步探讨了在证明中发现的问题.最后一节对本文进行总结.

---

* 本文研究得到国家863高科技项目基金(No. 863 306-ZD 10-02)资助.作者陈庆锋,1971年生,助理工程师,主要研究领域为信息安全,电子商务.王驹,1950年生,博士,研究员,主要研究领域为数理逻辑,计算机理论.白硕,1956年生,博士,研究员,博士生导师,主要研究领域为人工智能,计算机语言学,Internet/Intranet应用软件.张师超,1962年生,博士,教授,主要研究领域为人工智能,数据库技术.隋立颖,女,1973年生,硕士,主要研究领域为Internet/Intranet应用软件,计算机理论.

本文通讯联系人:白硕,北京 100080,国家智能计算机研究开发中心

本文1998-04-03收到原稿,1999-03-22收到修改稿

## 1 SET 协议的逻辑验证

在文献[6]中我们只列举了 SET 的 3 个范例,用扩展后的 NDL 逻辑框架对它们进行了逻辑验证.下面将给出 SET 付费业务流程的 5 个阶段的全面证明.

首先我们给出两个定理*:

**定理 3.** $Know(x,CertS(y))\rightarrow Know(x,Spb(y))$.

证明:

(1) $Know(x,CertS(y))$ [前提]

(2) $Know(x,Sign(CA,\langle y,Spb(y)\rangle))$ (1)[定义]

(3) $Know(x,\langle\langle y,Spb(y)\rangle,S(H(\langle y,Spb(y)\rangle),Spv(CA))\rangle)$ (2)[定义]

(4) $Know(x,\langle y,Spb(y)\rangle)$ (3)[6-1]

(5) $Know(x,Spb(y))$ (4)[6-1]

(5)即为所求. □

**定理 4.** $Know(x,CertK(y))\rightarrow Know(x,Kpb(y))$.

证法同定理 3.

### 1.1 SET 协议中的持卡人注册流程验证

已知:

$P=\{Know(CA,CertS(CA)),Know(CA,CertK(CA)),Know(C,Acct(C)),Know(C,Spb(CARoot)),$
$Know(C,Kpb(CARoot))\},$

$\alpha=Generate(C,InitReq)\circ Send(C,CA,InitReq)\circ Generate(CA,InitRes)\circ$
$Send(CA,C,Sign(CA,InitRes))\circ Send(CA,C,CertS(CA))\circ Send(CA,C,CertK(CA))\circ$
$Verify(C,CertS(CA),\langle X_2,\ldots,X_{n-1},CARoot\rangle)\circ Verify(C,CertK(CA),\langle X_2,\ldots,X_{n-1},CARoot\rangle)\circ$
$Generate(C,RegFormReq)\circ Generate(C,k_1)\circ Send(C,CA,E(RegFormReq,k_1))\circ$
$Send(C,CA,S(\langle k_1,Acct(C)\rangle,Kpb(CA)))\circ Generate(CA,RegForm)\circ$
$Send(CA,C,CertS(CA))\circ Send(CA,C,Sign(CA,RegForm))\circ$
$Verify(C,CertS(CA),\langle X_2,\ldots,X_{n-1},CARoot\rangle)\circ Generate(C,\langle Spb(C),Spv(C)\rangle)\circ$
$Generate(C,Filled\text{-}RegForm)\circ Generate(C,CertReq)\circ Generate(C,k_2)\circ$
$Generate(C,k_3)\circ Send(C,CA,E(Sign(C,\langle CertReq,k_2,Spb(C)\rangle),k_3))\circ$
$Send(C,CA,S(\langle k_3,Acct(C)\rangle,Kpb(CA)))\circ Legal(CA,CertReq)\circ$
$Generate(CA,CertS(C))\circ Generate(CA,CertRes)\circ$
$Send(CA,C,E(Sign(CA,CertRes),k_2))\circ Send(CA,C,CertS(CA))\circ$
$Send(CA,C,CertS(C))\circ Verify(C,CertS(CA),\langle X_2,\ldots,X_{n-1},CARoot\rangle)\circ$
$Verify(C,CertS(C),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle),$

$Q=\{Auth(C,X_2,\langle CA,Spb(CA)\rangle),Auth(C,X_2,\langle CA,Kpb(CA)\rangle),Auth(C,CA,InitRes),$
$Auth(C,X_2,\langle CA,Spb(CA)\rangle),Auth(C,CA,RegForm),$
$Auth(CA,C,\langle CertReq,k_2,Spb(C)\rangle);Auth(C,X_2,\langle CA,Spb(CA)\rangle),$
$Auth(C,CA,\langle C,Spb(C)\rangle),Auth(C,CA,CertRes)\}.$

求证:$P\vdash_{\alpha}Q$

证明:

(1) $Know(CA,CertS(CA))$ [前提]

(2) $Know(CA,CertK(CA))$ [前提]

(3) $Know(C,Acct(C))$ [前提]

---

* 由于是文献[8]中定理 1、2 的补充结论,作者为引用方便计,在编号上顺延下来.

(4)　$Know(C,Spb(CARoot))$　[前提]

(5)　$Know(C,Kpb(CARoot))$　[前提]

(6)　$Generate(C,InitReq)$　[动作]

(7)　$Know(C,InitReq)$　(6)[R-2]

(8)　$Send(C,CA,InitReq)$　[动作]

(9)　$Know(CA,InitReq)$　(8)[R-1]

(10)　$Generate(CA,InitRes)$　[动作]

(11)　$Know(CA,InitRes)$　(10)[R-2]

(12)　$Know(CA,Spv(CA))$　[2-2]

(13)　$Know(CA,S(H(InitRes),Spv(CA)))$　(11)(12)[4-2]

(14)　$Know(CA,\langle InitRes,S(H(InitRes),Spv(CA))\rangle)$　(11)(13)[6-1]

(15)　$Know(CA,Sign(CA,InitRes))$　(14)[定义]

(16)　$Send(CA,C,Sign(CA,InitRes))$　[动作]

(17)　$Send(CA,C,CertS(CA))$　[动作]

(18)　$Send(CA,C,CertK(CA))$　[动作]

(19)　$Know(C,Sign(CA,InitRes))$　(16)[R-1]

(20)　$Know(C,CertS(CA))$　(17)[R-1]

(21)　$Know(C,CertK(CA))$　(18)[R-1]

(22)　$Verify(C,CertS(CA),\langle X_2,\ldots,X_{n-1},CARoot\rangle)$　(4)(20)[动作]

/*如果检验失败,$C$ 没有在 PKI 树中找到 $CARoot$,则持卡人停止注册

其中 $X_2$ 是发放 $CertS(CA)$ 的上一级 $CA$ */

(23)　$IsVerified(C,X_2,CertS(CA))$　(22)[R-6]

(24)　$Auth(C,X_2,\langle CA,Spb(CA)\rangle)$　(23)[7-1]

/* $X_2,\ldots,X_{n-1},CARoot$ 为 PKI 树的各级证书授权当局 */

(25)　$Verify(C,CertK(CA),\langle X_2,\ldots,X_{n-1},CARoot\rangle)$　(4)(21)[动作]

/*如果检验失败,$C$ 没有在 PKI 树中找到 $CARoot$,则持卡人停止注册 */

(26)　$IsVerified(C,X_2,CertK(CA))$　(25)[R-6]

(27)　$Auth(C,X_2,\langle CA,Kpb(CA)\rangle)$　(26)[7-2]

/* $X_2,\ldots,X_{n-1},CARoot$ 为 PKI 树的各级证书授权当局 */

(28)　$Know(C,Spb(CA))$　(20)[定理 3]

(29)　$Auth(C,CA,InitRes)$　(19)(28)[定理 2]

(30)　$Generate(C,RegFormReq)$　[动作]

(31)　$Know(C,RegFormReq)$　(30)[R-2]

(32)　$Generate(C,k_1)$　[动作]

(33)　$Know(C,k_1)$　(32)[R-2]

(34)　$Know(C,E(RegFormReq,k_1))$　(31)(33)[1-1]

(35)　$Know(C,\langle k_1,Acct(C)\rangle)$　(3)(34)[6-1]

(36)　$Know(C,Kpb(CA))$　(21)[定理 4]

(37)　$Know(C,S(\langle k_1,Acct(C)\rangle,Kpb(CA)))$　(35)(36)[1-2]

(38)　$Send(C,CA,E(RegFormReq,k_1))$　[动作]

(39)　$Send(C,CA,S(\langle k_1,Acct(C)\rangle,Kpb(CA)))$　[动作]

(40)　$Know(CA,E(RegFormReq,k_1))$　(38)[R-1]

(41)　$Know(CA,S(\langle k_1,Acct(C)\rangle,Kpb(CA)))$　(39)[R-1]

(42) $Know(CA,\langle k_1,Acct(C)\rangle)$ (41)[2-1][3-2]

(43) $Know(CA,k_1)$ (42)[6-1]

(44) $Know(CA,Acct(C))$ (42)[6-1]

(45) $Know(CA,RegFormReq)$ (40)(43)[3-1]

(46) $Generate(CA,RegForm)$ [动作]

(47) $Know(CA,RegForm)$ (46)[R-2]

(48) $Know(CA,H(RegForm))$ (47)[4-1]

(49) $Know(CA,S(H(Regform),Spv(CA)))$ (47)(12)[4-2]

(50) $Know(CA,Sign(CA,RegForm))$ (47)(49)[定义]

(51) $Send(CA,C,CertS(CA))$ [动作]

(52) $Send(CA,C,Sign(CA,RegForm))$ [动作]

(53) $Know(C,CertS(CA))$ (51)[R-1]

(54) $Know(C,Sign(CA,RegForm))$ (52)[R-1]

(55) $Verify\ (C,CertS(CA),\langle X_2,\ldots,X_{n-1},CARoot\rangle)$ (4)(53)[动作]

/* 如果检验失败,$C$ 没有在 PKI 树中找到 $CARoot$,则持卡人停止注册 */

(56) $IsVerified(C,X_2,CertS(CA))$ (55)[R-6]

(57) $Auth(C,X_2,\langle CA,Spb(CA)\rangle)$ (56)[7-1]

(58) $Know(C,Spb(CA))$ (53)[定理 3]

(59) $Auth(C,CA,RegForm)$ (54)(58)[定理 2]

(60) $Generate(C,\langle Spb(C),Spv(C)\rangle)$ [动作]

(61) $Know(C,Spb(C))$ (60)[R-2]

(62) $Know(C,Spv(C))$ (60)[R-2]

(63) $Generate\ (C,Filled\text{-}RegForm)$ [动作]

/* 持卡人 $C$ 向注册表填信息/$eg$,持卡人的名称,届满日期,帐单地址以及其他一些被金融机构用来识别是否为持卡人的信息 */

(64) $Generate\ (C,CertReq)$ [动作]

/* $CertReq$ 中包含了向注册表 $RegForm$ 中填入的信息 */

(65) $Know(C,CertReq)$ (64)[R-2]

(66) $Generate(C,k_2)$ [动作]

(67) $Know(C,k_2)$ (66)[R-2]

(68) $Know(C,\langle CertReq,k_2,Spb(C)\rangle)$ (65)(67)(61)[6-1]

(69) $Know(C,S(H\langle CertReq,k_2,Spb(C)\rangle,Spv(C)))$ (68)(62)[4-2]

(70) $Know(C,Sign(C,\langle CertReq,k_2,Spb(C)\rangle))$ (68)(69)[定义]

(71) $Generate(C,k_3)$ [动作]

(72) $Know(C,k_3)$ (71)[R-2]

(73) $Know(C,E(Sign(C,\langle CertReq,k_2,Spb(C)\rangle),k_3)$ (70)(72)[1-1]

(74) $Know(C,\langle k_3,Acct(C)\rangle)$ (3)(72)[6-1]

(75) $Know(C,S(\langle k_3,Acct(C)\rangle,Kpb(CA)))$ (74)(36)[1-2]

(76) $Send(C,CA,E(Sign(C,\langle\langle CertReq,k_2,Spb(C)\rangle,k_3\rangle))$ [动作]

(77) $Send(C,CA,S(\langle k_3,Acct(C)\rangle,Kpb(CA)))$ [动作]

(78) $Know(CA,E(Sign(C,\langle CertReq,k_2,Spb(C)\rangle),k_3))$ (76)[R-1]

(79) $Know(CA,S(\langle k_3,Acct(C)\rangle,Kpb(CA)))$ (77)[R-1]

(80) $Know(CA,\langle k_3,Acct(C)\rangle)$ (79)[2-1][3-2]

(81)　$Know(CA,k_1)$　(80)[6-1]

(82)　$Know(CA,Acct(C))$　(80)[6-1]

(83)　$Know(CA,Sign(C,\langle CertReq,k_2,Spb(C)\rangle))$　(78)(81)[3-1]

(84)　$Know(CA,\langle CertReq,k_2,Spb(C)\rangle)$　(83)[定义]

(85)　$Know(CA,CertReq)$　(84)[6-1]

(86)　$Know(CA,k_2)$　(84)[6-1]

(87)　$Know(CA,Spb(C))$　(84)[6-1]

(88)　$Auth(CA,C,\langle CertReq,k_2,Spb(C)\rangle)$　(83)(87)[定理 2]

(89)　$Legal\ (CA,CertReq)$　[动作]

/*CA 用持卡人知道的 $Acct(C)$和 $CertReq$ 中注册表的信息检验 $CertReq$ 的合法性*/

(90)　$Generate(CA,CertS(C))$　[动作]

(91)　$Know(CA,CertS(C))$　(90)[R-2]

(92)　$Generate(CA,CertRes)$　[动作]

(93)　$Know(CA,CertRes)$　(92)[动作]

(94)　$Know(CA,S(H(CertRes),Spv(CA)))$　(93)[2-2][4-2]

(95)　$Know(CA,Sign(CA,CertRes))$　(93)(94)[定义]

(96)　$Know(CA,E(Sign(CA,CertRes),k_2))$　(86)(95)[1-1]

(97)　$Send(CA,C,E(Sign(CA,CertRes),k_2))$　[动作]

(98)　$Send(CA,C,CertS(CA))$　[动作]

(99)　$Send(CA,C,CertS(C))$　[动作]

(100)　$Know(C,E(Sign(CA,CertRes),k_2))$　(97)[R-1]

(101)　$Know(C,CertS(CA))$　(98)[R-1]

(102)　$Know(C,CertS(C))$　(99)[R-1]

(103)　$Verify\ (C,CertS(CA),\langle X_2,\ldots,X_{n-1},CARoot\rangle)$　(4)(101)[动作]

/*如果检验失败,$C$ 没有在 PKI 树中找到 $CARoot$,则持卡人停止注册*/

(104)　$IsVerified(C,X_2,CertS(CA))$　(103)[R-6]

(105)　$Auth(C,X_2,\langle CA,Spb(CA)\rangle)$　(104)[7-1]

(106)　$Verify\ (C,CertS(C),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)$　(4)(102)[动作]

/*如果检验失败,$C$ 没有在 PKI 树中找到 $CARoot$,则持卡人停止注册*/

(107)　$IsVerified(C,CA,CertS(C))$　(106)[R-6]

(108)　$Auth(C,CA,\langle C,Spb(C)\rangle)$　(107)[7-1]

(109)　$Know(C,Sign(CA,CertRes))$　(67)(100)[3-1]

(110)　$Know(C,Spb(CA))$　(102)[定理 3]

(111)　$Auth(C,CA,CertRes)$　(109)(110)[定理 2]

(112)　"持卡人存储证书 $CertS(C)$以及 $CA$ 的回答中的相关信息,用于以后的电子商务"

### 1.2　SET 协议中的商家注册流程验证

已知:

$P=\{Know(CA,CertS(CA)),Know(CA,CertK(CA)),Know(M,Acct(M)),Know(M,Spb(CARoot))\}$,

$\alpha=Generate(M,InitReq)\circ Send(M,CA,InitReq)\circ Generate(CA,RegForm)\circ$

$Send(CA,M,Sign(CA,RegForm))\circ Send(CA,M,CertS(CA))\circ Send(CA,M,CertK(CA))\circ$

$Verify(M,CertS(CA),\langle X_2,\ldots,X_{n-1},CARoot\rangle)\circ$

$Verify(M,CertK(CA),\langle X_2,\ldots,X_{n-1},CARoot\rangle)\circ Generate(M,\langle Sph(M),Spv(M)\rangle)\circ$

$Generate(M,\langle Kpb(M),Kpv(M)\rangle)\circ Generate(M,Filled\text{-}RegForm)\circ$

$Generate(M,CertReq)\circ Generate(M,k_1)\circ$

$Send(M,CA,E(Sign(M,\langle CertReq,Spb(M),Kpb(M)\rangle),k_1))\circ$

$Send(M,CA,S(\langle Acct(M),k_1\rangle,Spb(CA)))\circ Legal(CA,CerReq)\circ$

$Generate(CA,CertS(M))\circ Generate(CA,CertK(M))\circ Generate(CA,CertRes)\circ$

$Send(CA,M,Sign(CA,CertRes))\circ Send(CA,M,CertS(M))\circ Send(CA,M,CertK(M))\circ$

$Send(CA,M,CertS(CA))\circ Verify(M,CertS(CA),\langle X_2,\ldots,X_{n-1},CARoot\rangle)\circ$

$Verify(M,CertK(M),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)\circ$

$Verify(M,CertS(M),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)$,

$Q=\{Auth(M,X_2,\langle CA,Spb(CA)\rangle),Auth(M,CA,RegForm);$

$Auth(CA,M,\langle CertReq,Spb(M),Kpb(M)\rangle);Auth(M,CA,\langle M,Spb(M)\rangle)$,

$Auth(M,CA,\langle M,Kpb(M)\rangle),Auth(M,X_2,\langle CA,Spb(CA)\rangle),Auth(M,CA,CertRes)\}$.

求证:$P\mid\sim Q$

证明:

| | | |
|---|---|---|
| (1) | $Know(CA,CertS(CA))$ | [前提] |
| (2) | $Know(CA,CertK(CA))$ | [前提] |
| (3) | $Know(M,Acct(M))$ | [前提] |
| (4) | $Know(M,Spb(CARoot))$ | [前提] |
| (5) | $Generate(M,InitReq)$ | [动作] |
| (6) | $Know(M,InitReq)$ | (5)[R-2] |
| (7) | $Send(M,CA,InitReq)$ | [动作] |
| (8) | $Know(CA,InitReq)$ | (7)[R-1] |
| (9) | $Generate(CA,RegForm)$ | [动作] |
| (10) | $Know(CA,RegForm)$ | (9)[R-2] |
| (11) | $Know(CA,S(H(RegForm),Spv(CA)))$ | (10)[2-2][4-2] |
| (12) | $Know(CA,Sign(CA,RegForm))$ | (10)(11)[定义] |
| (13) | $Send(CA,M,Sign(CA,RegForm))$ | [动作] |
| (14) | $Send(CA,M,CertS(CA))$ | [动作] |
| (15) | $Send(CA,M,CertK(CA))$ | [动作] |
| (16) | $Know(M,Sign(CA,RegForm))$ | (13)[R-1] |
| (17) | $Know(M,CertS(CA))$ | (14)[R-1] |
| (18) | $Know(M,CertK(CA))$ | (15)[R-1] |
| (19) | $Verify(M,CerS(CA),\langle X_2,\ldots,X_{n-1},CARoot\rangle)$ | (17)(4)[动作] |
| | /* 如果检验失败,$M$ 没有在 PKI 树中找到 $CARoot$,则商家停止注册 */ | |
| (20) | $I:Verified(M,X_2,CertS(CA))$ | (19)[R-6] |
| (21) | $Auth(M,X_2,\langle CA,Spb(CA)\rangle)$ | (20)[7-1] |
| (22) | $Verify(M,CerK(CA),\langle X_2,\ldots,X_{n-1},CARoot\rangle)$ | (18)(4)[动作] |
| (23) | $I:Verified(M,X_2,CertK(CA))$ | (22)[R-6] |
| (24) | $Auth(M,X_2,\langle CA,Kpb(CA)\rangle)$ | (23)[7-1] |
| (25) | $Know(M,Spb(CA))$ | (17)[定理 3] |
| (26) | $Auth(M,CA,RegForm)$ | (16)(25)[定理 2] |
| (27) | $Generate(M,\langle Spb(M),Spv(M)\rangle)$ | [动作] |
| (28) | $Generate(M,\langle Kpb(M),Kpv(M)\rangle)$ | [动作] |
| (29) | $Know(M,Spb(M))$ | (27)[R-2] |

(30) $Know(M,Spv(M))$ (27)[R-2]

(31) $Know(M,Kpb(M))$ (28)[R-2]

(32) $Know(M,Kpv(M))$ (28)[R-2]

(33) $Generate(M,Filled\text{-}RegForm)$ [动作]

/*商家向注册表填入信息(eg,姓名,地址,身份证号)*/

(34) $Generate(M,CertReq)$ [动作]

/*CertReq 中包含了向注册表 RegForm 中填入的信息*/

(35) $Know(M,CertReq)$ (34)[R-2]

(36) $Know(M,\langle CertReq,Spb(M),Kpb(M)\rangle)$ (29)(31)(35)[6-1]

(37) $Know(M,S(H(CertReq,Spb(M),Kpb(M)),Spv(M)))$ (36)(29)[4-2]

(38) $Know(M,Sign(M,\langle CertReq,Spb(M),Kpb(M)\rangle))$ (36)(37)[定义]

(39) $Generate(M,k_1)$ [动作]

(40) $Know(M,k_1)$ (39)[R-2]

(41) $Know(M,E(Sign(M,\langle CertReq,Spb(M),Kpb(M)\rangle),k_1))$ (38)(40)[1-1]

(42) $Know(M,\langle Acct(M),k_1\rangle)$ (3)(40)[6-1]

(43) $Know(M,E(\langle Acct(M),k_1\rangle,Spb(CA)))$ (42)(25)[1-1]

(44) $Send(M,CA,E(Sign(M,\langle CertReq,Spb(M),Kpb(M)\rangle),k_1))$ [动作]

(45) $Send(M,CA,S(\langle Acct(M),k_1\rangle,Spb(CA)))$ [动作]

(46) $Know(CA,E(Sign(M,\langle CertReq,Spb(M),Kpb(M)\rangle),k_1))$ (44)[R-1]

(47) $Know(CA,S(\langle Acct(M),k_1\rangle,Spb(CA)))$ (45)[R-1]

(48) $Know(CA,\langle Acct(M),k_1\rangle)$ (47)[2-2][3-2]

(49) $Know(CA,Acct(M))$ (48)[6-1]

(50) $Know(CA,k_1)$ (48)[6-1]

(51) $Know(CA,Sign(M,\langle CertReq,Spb(M),Kpb(M)\rangle))$ (46)(50)[3-1]

(52) $Know(CA,\langle CertReq,Spb(M),Kpb(M)\rangle)$ (51)[定义]

(53) $Know(CA,Spb(M))$ (52)[6-1]

(54) $Auth(M,CA,\langle CertReq,Spb(M),Kpb(M)\rangle)$ (51)(53)[定理 2]

(55) $Know(CA,CertReq)$ (52)[6-1]

(56) $Legal(CA,CerReq)$ [动作]

/*CA 用已知商家的信息检验 CertReq 中注册表的合法性.若不合法,则 CA 终止商家注册过程*/

(57) $Generate(CA,CertS(M))$ [动作]

(58) $Generate(CA,CertK(M))$ [动作]

(59) $Know(CA,CertS(M))$ (57)[R-2]

(60) $Know(CA,CertK(M))$ (58)[R-2]

(61) $Generate(CA,CertRes)$ [动作]

(62) $Know(CA,CertRes)$ (61)[R-2]

(63) $Know(CA,S(H(CertRes),Spv(CA)))$ (62)[2-2][4-2]

(64) $Know(CA,Sign(CA,CertRes))$ (62)(63)[定义]

(65) $Send(CA,M,Sign(CA,CertRes))$ [动作]

(66) $Send(CA,M,CertS(M))$ [动作]

(67) $Send(CA,M,CertK(M))$ [动作]

(68) $Send(CA,M,CertS(CA))$ [动作]

(69) $Know(M,Sign(CA,CertRes))$ (65)[R-1]

(70) $Know(M,CertS(M))$ (66)[R-1]

(71) $Know(M,CertK(M))$ (67)[R-1]

(72) $Know(M,CertS(CA))$ (68)[R-1]

(73) $Verify(M,CertS(CA),\langle X_2,\ldots,X_{n-1},CARoot\rangle)$ (3)(72)[动作]

/*如果检验失败,$M$ 没有在 PKI 树中找到 $CARoot$,则商家停止注册*/

(74) $Isverified(M,X_2,CertS(CA))$ (73)[R-6]

(75) $Auth(M,X_2,\langle CA,Spb(CA)\rangle)$ (74)[7-1]

(76) $Verify(M,CertS(M),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)$ (3)(70)[动作]

/*如果检验失败,$M$ 没有在 PKI 树中找到 $CARoot$,则商家停止注册*/

(77) $Isverified(M,CA,CertS(M))$ (76)[R-6]

(78) $Auth(M,CA,\langle M,Spb(M)\rangle)$ (77)[7-1]

(79) $Verify(M,CertK(M),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)$ (3)(71)[动作]

/*如果检验失败,$M$ 没有在 PKI 树中找到 $CARoot$,则商家停止注册*/

(80) $Isverified(M,CA,CertK(M))$ (76)[R-6]

(81) $Auth(M,CA,\langle M,Kpb(M)\rangle)$ (80)[7-2]

(82) $Know(M,Spb(CA))$ (72)[定理 3]

(83) $Auth(M,CA,CertRes)$ (69)(82)[定理 2]

(84) "商家存储证书 $CertS(M)$,$CertK(M)$以及 CA 的响应中的相关信息,用于以后的电子商务"

### 1.3 SET 协议中的购买请求流程验证

已知:

$P=\{Know(M,CertS(M)),Know(M,CertK(P)),Know(C,Acct(C)),$
$Know(C,CertS(C)),Know(C,Spb(CARoot)),Know(M,Spb(CARoot))\},$

$a=Generate(C,InitReq)\circ Send(C,M,InitReq)\circ Generate(M,InitRes)\circ$
$Send(M,C,Sign(M,InitReq))\circ Send(M,C,CertS(M))\circ Send(M,C,CertK(P))\circ$
$Verify(C,\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)\circ Verify(C,\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)\circ$
$Generate(C,OI)\circ Generate(C,PI)\circ Generate(C,k_1)\circ$
$Send(C,M,E(\langle PI,H(OI),So(C,\langle H(OI),H(PI)\rangle)\rangle,k_1))\circ$
$Send(C,M,S(\langle Acct(C),k_1\rangle,Kpb(P)))\circ$
$Send(C,M,\langle OI,H(PI),So(C,\langle H(OI),H(PI)\rangle)\rangle)\circ$
$Send(C,M,CertS(C))\circ Verify(M,CertS(C),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)\circ$
$Generate(M,PurchRes)\circ Send(M,C,Sign(M,PurchRes))\circ Send(M,C,CertS(M))\circ$
$Verify(C,CertS(M),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle),$

$Q=\{Auth(C,CA,\langle M,Spb(M)\rangle),Auth(C,CA,\langle P,Spb(P)\rangle),Auth(C,M,InitReq),$
$Auth(M,CA,\langle C,Spb(C)\rangle),Auth(M,C,\langle H(OI),H(PI)\rangle);Auth(C,CA,\langle M,Spb(M)\rangle),$
$Auth(C,CA,PurchRes)\}.$

求证:$P\vdash_a Q$

证明:

(1) $Know(M,CertS(M))$ [前提]

(2) $Know(M,CertK(P))$ [前提]

(3) $Know(C,Acct(C))$ [前提]

(4) $Know(C,CertS(C))$ [前提]

(5) $Generate(C,InitReq)$ [动作]

(6) $Know(C,InitReq)$ (5)[R-2]

(7) $Send(C,M,InitReq)$ [动作]

(8) $Know(M,InitReq)$ (7)[R-1]

(9) $Generate(M,InitRes)$ [动作]

(10) $Know(M,InitReq)$ (9)[R-2]

(11) $Know(M,S(H(InitReq),Spv(M)))$ (10)[2-2][4-2]

(12) $Know(M,Sign(M,InitReq))$ (10)(11)[定义]

(13) $Send(M,C,Sign(M,InitReq))$ [动作]

(14) $Send(M,C,CertS(M))$ [动作]

(15) $Send(M,C,CertK(P))$ [动作]

(16) $Know(C,Sign(M,InitReq))$ (13)[R-1]

(17) $Know(C,CertS(M))$ (14)[R-1]

(18) $Know(C,CertK(P))$ (15)[R-1]

(19) $Know(C,Spb(CARoot))$ [前提]

(20) $Verify\ (C,CertS(M),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)$ (17)(19)[动作]

/* 如果检验失败,C 没有在 PKI 树中找到 CARoot,则持卡人终止购买请求 */

(21) $IsVerified(C,CA,CertS(M))$ (20)[R-6]

(22) $Auth(C,CA,\langle M,Spb(M)\rangle)$ (21)[7-1]

(23) $Verify\ (C,CertK(P),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)$ (18)(19)[动作]

/* 如果检验失败,C 没有在 PKI 树中找到 CARoot,则持卡人终止购买请求 */

(24) $IsVerified(C,CA,CertK(P))$ (23)[R-6]

(25) $Auth(C,CA,\langle P,Spb(P)\rangle)$ (24)[7-1]

(26) $Know(C,Spb(M))$ (17)[定理 3]

(27) $Auth(C,M,InitReq)$ (26)(16)[定理 2]

(28) $Generate(C,OI)$ [动作]

(29) $Generate(C,PI)$ [动作]

(30) $Know(C,OI)$ (28)[R-2]

(31) $Know(C,PI)$ (29)[R-2]

(32) $Know(C,H(OI))$ (30)[4-1]

(33) $Know(C,H(PI))$ (31)[4-1]

(34) $Know(C,\langle H(OI),H(PI)\rangle)$ (32)(33)[4-1]

(35) $Know(C,H(\langle H(OI),H(PI)\rangle))$ (34)[4-1]

(36) $Know(C,S(H(\langle H(OI),H(PI)\rangle),Spv(C)))$ (34)[2-2][4-2]

(37) $Know(C,So(C,\langle H(OI),H(PI)\rangle))$ (36)[定义]

(38) $Generate(C,k_1)$ [动作]

(39) $Know(C,k_1)$ (38)[R-2]

(40) $Know(C,\langle PI,H(OI),So(C,\langle H(OI),H(PI)\rangle)\rangle)$ (31)(32)(37)[6-1]

(41) $Know(C,E(\langle PI,H(OI),So(C,\langle H(OI),H(PI)\rangle)\rangle,k_1))$ (39)(40)[1-1]

(42) $Know(C,\langle Acct(C),k_1\rangle)$ (3)(39)[4-1]

(43) $Know(C,Kpb(P))$ (18)[定理 4]

(44) $Know(C,S(\langle Acct(C),k_1\rangle,Kpb(P)))$ (43)(42)[1-2]

(45) $Know(C,\langle OI,H(PI),So(C,\langle H(OI),H(PI)\rangle)\rangle)$ (30)(33)(37)[4-1]

(46) $Send(C,M,E(\langle PI,H(OI),So(C,\langle H(OI),H(PI)\rangle)\rangle,k_1))$ [动作]

(47) $Send(C,M,S(\langle Acct(C),k_1\rangle,Kpb(P)))$ [动作]

(48) $Send(C,M,\langle OI,H(PI),So(C,\langle H(OI),H(PI)\rangle)\rangle)$ [动作]

(49) $Send(C,M,CertS(C))$ [动作]

(50) $Know(M,E(\langle PI,H(OI),So(C,\langle H(OI),H(PI)\rangle)\rangle,k_1))$ (46)[R-1]

(51) $Know(M,S(\langle Acct(C),k_1\rangle,Kpb(P)))$ (47)[R-1]

(52) $Know(M,\langle OI,H(PI),So(C,\langle H(OI),H(PI)\rangle)\rangle)$ (48)[R-1]

(53) $Know(M,CertS(C))$ (49)[R-1]

(54) $Know(M,Spb(CARoot))$ [前提]

(55) $Verify(M,CertS(C),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)$ (53)(54)[动作]

/* 如果检验失败,$M$ 没有在 PKI 树中找到 $CARoot$,则商家终止购买请求 */

(56) $IsVerified(M,CA,CertS(C))$ (55)[R-6]

(57) $Auth(M,CA,\langle C,Spb(C)\rangle)$ (56)[7-1]

(58) $Know(M,OI)$ (52)[6-1]

(59) $Know(M,H(PI))$ (52)[6-1]

(60) $Know(M,So(C,\langle H(OI),H(PI)\rangle))$ (52)[6-1]

(61) $Know(M,H(OI))$ (58)[4-1]

(62) $Know(M,\langle H(OI),H(PI)\rangle)$ (59)(61)[4-1]

(63) $Know(M,Spb(C))$ (53)[R-6]

(64) $Auth(M,C,\langle H(OI),H(PI)\rangle)$ (60)(63)[定理 1]

(65) "商家 $M$ 处理请求(包括将付费指令 PI 提交付费网关 $P$ 认证)"

(66) $Generate(M,PurchRes)$ [动作]

(67) $Know(M,PurchRes)$ (66)[R-2]

(68) $Know(M,S(H(PurchRes),Spv(M)))$ (67)[2-2][4-2]

(69) $Know(M,Sign(M,PurchRes))$ (67)(68)[定义]

(70) $Send(M,C,Sign(M,PurchRes))$ [动作]

(71) $Send(M,C,CertS(M))$ [动作]

(72) $Know(C,Sign(M,PurchRes))$ (70)[R-1]

(73) $Know(C,CertS(M))$ (71)[R-1]

(74) "商家 $M$ 认证事务,然后履行业务要求(e.g. 向持卡人 $C$ 发送货物)"

(75) $Verify(C,CertS(M),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)$ (73)[2-5][动作]

/* 如果检验失败,$C$ 没有在 PKI 树中找到 $CARoot$,则持卡人终止购买请求 */

(76) $IsVerified(C,CA,CertS(M))$ (75)[R-6]

(77) $Auth(C,CA,\langle M,Spb(M)\rangle)$ (76)[7-1]

(78) $Know(C,Spb(M))$ (73)[定理 3]

(79) $Auth(C,M,PurchRes)$ (72)(78)[定理 2]

(80) "持卡人 $C$ 存储购买回答 $PurchRes$"

### 1.4 SET 协议中的付费认证流程验证

已知:

$$P=\{Know(P,Kpb(P)),Know(P,Spb(P)),Know(M,Kpb(P)),Know(P,Spb(CARoot)),$$
$$Know(M,Spb(CARoot)),Know(P,Acct(C)),Know(P,CertS(P)),Know(M,CertS(C)),$$
$$Know(M,CertS(M)),\ Know(M,CertK(M)),$$
$$Know(M,E(\langle PI,H(OI),SO(C,\langle H(OI),H(PI)\rangle)\rangle,k_1)),$$
$$Know(M,S(\langle Acct(C),k_1\rangle,Kpb(P)))\},$$

$\alpha=Generate(M,AuthReq)\circ Generate(M,k_2)\circ Send(M,P,E(Sign(M,AuthReq),k_2))\circ$
$Send(M,P,S(k_2,Kpb(P)))\circ Send(M,P,E(\langle PI,H(OI),So(C,\langle H(OI),H(PI)\rangle)\rangle,k_1))\circ$
$Send(M,P,\langle S(Acct(C),k_1\rangle,Kpb(P)))\circ Send(M,P,CertS(C))\circ Send(M,P,CertS(M))\circ$
$Send(M,P,CertK(M))\circ Verify(P,CertS(C),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)\circ$
$Verify(P,CertS(M),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)\circ$
$Verify(P,CertK(M),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)\circ Generate(P,AuthRes)\circ$
$Generate(P,k_3)\circ Generate(P,CapToken)\circ Send(P,M,S(\langle k_3,Acct(C)\rangle,Kpb(M)))\circ$
$Send(P,M,E(Sign(P,AuthRes),k_3))\circ Send(P,M,S(\langle k_4,Acct(C)\rangle,Kpb(P)))\circ$
$Send(P,M,E(Sign(P,CapToken),k_4))\circ Send(P,M,CertS(P))\circ$
$Verify(M,CertS(P),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)$,

$Q=\{Auth(P,M,AuthReq),Auth(P,CA,\langle C,Spb(C)\rangle),Auth(P,CA,\langle M,Spb(M)\rangle),$
$Auth(P,CA,\langle M,Kpb(M)\rangle),Auth(P,C,\langle H(OI),H(PI)\rangle),$
$Auth(M,CA,\langle P,Spb(P)\rangle),Auth(M,P,AuthRes)\}$.

求证:$P\vdash_\alpha Q$

证明:

| | | |
|---|---|---|
| (1) | $Know(P,CertS(P))$ | [前提] |
| (2) | $Know(M,Kpb(P))$ | [前提] |
| (3) | $Know(M,E(\langle PI,H(OI),So(C,\langle H(OI),H(PI)\rangle)\rangle,k_1))$ | [前提] |
| (4) | $Know(M,S(\langle Acct(C),k_1\rangle,Kpb(P)))$ | [前提] |
| (5) | $Know(M,CertS(C))$ | [前提] |
| (6) | $Know(M,CertS(M))$ | [前提] |
| (7) | $Know(M,CertK(C))$ | [前提] |
| (8) | $Generate(M,AuthReq)$ | [动作] |
| (9) | $Know(M,AuthReq)$ | (8)[R-2] |
| (10) | $Know(M,S(H(AuthReq),Spv(M)))$ | (9)(2-2)[4-2] |
| (11) | $Know(M,Sign(M,AuthReq))$ | (9)(10)[定义] |
| (12) | $Generate(M,k_2)$ | [动作] |
| (13) | $Know(M,k_2)$ | (12)[R-2] |
| (14) | $Know(M,E(Sign(M,AuthReq),k_2))$ | (11)(13)[1-1] |
| (15) | $Know(M,S(k_2,Kpb(p)))$ | (2)(13)[1-2] |
| (16) | $Send(M,P,E(Sign(M,AuthReq),k_2))$ | [动作] |
| (17) | $Send(M,P,S(k_2,Kpb(P)))$ | [动作] |
| (18) | $Send(M,P,E(\langle PI,H(OI),So(C,\langle H(OI),H(PI)\rangle)\rangle,k_1))$ | [动作] |
| (19) | $Send(M,P,S(\langle Acct(C),k_1\rangle,Kpb(P)))$ | [动作] |
| (20) | $Send(M,P,CertS(C))$ | [动作] |
| (21) | $Send(M,P,CertS(M))$ | [动作] |
| (22) | $Send(M,P,CertK(M))$ | [动作] |
| (23) | $Know(P,E(Sign(M,AuthReq),k_2))$ | (16)[R-1] |
| (24) | $Know(P,S(k_2,Kpb(P)))$ | (17)[R-2] |
| (25) | $Know(P,E(\langle PI,H(OI),So(C,\langle H(OI),H(PI)\rangle)\rangle,k_1))$ | (18)[R-1] |
| (26) | $Know(P,S(\langle Acct(C),k_1\rangle,Kpb(P)))$ | (19)[R-1] |
| (27) | $Know(P,CertS(C))$ | (20)[R-1] |
| (28) | $Know(P,CertS(M))$ | (21)[R-1] |

(29) $Know(P,CertK(M))$ (22)[R-1]

(30) $Know(P,k_2)$ (24)[2-1][3-2]

(31) $Know(P,Sign(M,AuthReq))$ (23)(30)[R-1]

(32) $Know(P,Spb(M))$ (28)[定理 3]

(33) $Auth(P,M,AuthReq)$ (31)(32)[定理 2]

(34) $Verify(P,CertS(C),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)$ (27)[2-5][动作]

/ * 如果检验失败,$P$ 没有在 PKI 树中找到 $CARoot$,则付费网关终止付费认证 * /

(35) $IsVerified(P,CA,CertS(C))$ (34)[R-6]

(36) $Auth(P,CA,\langle C,Spb(C)\rangle)$ (35)[7-1]

(37) $Verify(P,CertS(M),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)$ (28)[2-5][动作]

/ * 如果检验失败,$P$ 没有在 PKI 树中找到 $CARoot$,则付费网关终止付费认证 * /

(38) $IsVerified(P,CA,CertS(M))$ (37)[R-6]

(39) $Auth(P,CA,\langle M,Spb(M)\rangle)$ (38)[7-1]

(40) $Verify(P,CertK(M),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)$ (29)[2-5][动作]

/ * 如果检验失败,$P$ 没有在 PKI 树中找到 $CARoot$,则付费网关终止付费认证 * /

(41) $IsVerified(P,CA,CertK(M))$ (40)[R-6]

(42) $Auth(P,CA,\langle M,Kpb(M)\rangle)$ (41)[7-1]

(43) $Know(P,\langle Acct(C),k_1\rangle)$ (26)[2-1][3-2]

(44) $Know(P,Acct(C))$ (43)[4-1]

(45) $Know(P,k_1)$ (43)[6-1]

(46) $Know(P,\langle PI,H(OI),So(C,\langle H(OI),H(PI)\rangle)\rangle)$ (25)(45)[3-1]

(47) $Know(P,PI)$ (46)[6-1]

(48) $Know(P,H(OI))$ (46)[6-1]

(49) $Know(P,So(C,\langle H(OI),H(PI)\rangle))$ (46)[6-1]

(50) $Know(P,H(PI))$ (47)[4-1]

(51) $Know(P,\langle H(OI),H(PI)\rangle)$ (48)(50)[4-1]

(52) $Know(P,Spb(C))$ (27)[定理 3]

(53) $Auth(P,C,\langle H(OI),H(PI)\rangle)$ (52)(49)(51)[定理 1]

(54) "付费网关 $P$ 确认商家 $M$ 的认证请求 $AuthReq$ 与持卡人 $C$ 的付费指令 PI 之间的一致"

(55) "付费网关 $P$ 把认证请求 $AuthReq$ 通过金融网传给持卡人的金融机构"

(56) $Generate(P,AuthRes)$ [动作]

(57) $Know(P,AuthRes)$ (56)[R-2]

(58) $Know(P,S(H(AuthRes),Spv(P)))$ (57)[2-2][4-2]

(59) $Know(P,Sign(P,AuthRes))$ (57)(58)[定义]

(60) $Generate(P,k_3)$ [动作]

(61) $Know(P,k_3)$ (60)[R-2]

(62) $Know(P,E(Sign(P,AuthRes),k_3))$ (59)(61)[1-1]

(63) $Know(P,S(k_3,Kpb(M)))$ (32)(61)[1-2]

(64) $Know(P,S(\langle k_3,Acct(C)\rangle,Kpb(M)))$ (32)(44)(61)[1-2]

(65) $Generate(P,CapToken)$ [动作]

(66) $Know(P,CapToken)$ (65)[R-2]

(67) $Know(P,S(H(Captoken),Spv(P)))$ (66)[2-2][4-2]

(68) $Know(P,Sign(P,Captoken))$ (66)(67)[定义]

(69) $Generate(P,k_4)$ [动作]

(70) $Know(P,k_4)$ (69)[R-2]

(71) $Know(P,E(Sign(P,CapToken),k_4))$ (68)(70)[1-1]

(72) $Know(P,S(\langle k_4,Acct(C)\rangle,Kpb(P)))$ (70)(44)[2-4][1-2]

(73) $Send(P,M,S(\langle k_3,Acct(C)\rangle,Kpb(M)))$ [动作]

(74) $Send(P,M,E(Sign(P,AuthRes),k_3))$ [动作]

(75) $Send(P,M,S(\langle k_4,Acct(C)\rangle,Kpb(P)))$ [动作]

(76) $Send(P,M,E(Sign(P,CapToken),k_4))$ [动作]

(77) $Send(P,M,CertS(P))$ [动作]

(78) $Know(M,S(\langle k_3,Acct(C)\rangle,Kpb(M)))$ (73)[R-1]

(79) $Know(M,E(Sign(P,AuthRes),k_3))$ (74)[R-1]

(80) $Know(M,S(\langle k_4,Acct(C)\rangle,Kpb(P)))$ (75)[R-1]

(81) $Know(M,E(Sign(P,CapToken),k_4))$ (76)[R-1]

(82) $Know(M,CertS(P))$ (77)[R-1]

(83) $Verify(M,CertS(P),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)$ (82)[2-5][动作]

/*如果检验失败,$M$ 没有在 PKI 树中找到 $CARoot$,则商家终止付费认证*/

(84) $IsVerified(M,CA,CertS(P))$ (83)[R-5]

(85) $Auth(M,CA,\langle P,Spb(P)\rangle)$ (84)[7-1]

(86) $Know(M,\langle k_3,Acct(C)\rangle)$ (78)[2-1][3-2]

(87) $Know(M,k_3)$ (87)[6-1]

(88) $Know(M,Sign(P,AuthRes))$ (79)(86)[3-1]

(89) $Know(M,Spb(P))$ (82)[定理 3]

(90) $Auth(M,P,AuthRes)$ (88)(89)[定理 2]

(91) "商家 $M$ 为将来的获得处理存储加密的获得令牌 $CapToken$ 和信封"

(92) "商家 $M$ 完成购买请求处理"

### 1.5 SET 协议中的付费获得流程验证

已知:

$$P=\{Know(M,S(\langle k_4,Acct(C)\rangle,Kpb(P))),Know(M,E(Sign(P,CapToken),k_4)),$$
$$Know(M,CertS(M)),Know(M,CertK(M)),Know(M,Kpb(P)),$$
$$Know(P,CertS(P)),Know(M,Spb(P))\},$$

$$a=Generate(M,CapReq)\circ Send(M,P,S(k_5,Kpb(P)))\circ$$
$$Send(M,P,E(Sign(M,CapReq),k_5))\circ Send(M,P,S(\langle k_4,Acct(C)\rangle,Kpb(P)))\circ$$
$$Send(M,P,E(Sign(P,CapToken),k_4)))\circ Send(M,P,CertS(M))\circ$$
$$Send(M,P,CertK(M))\circ Verify(P,CertS(M),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)\circ$$
$$Verify(P,CertK(M),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)\circ Generate(P,CapRes)\circ$$
$$Generate(P,k_6)\circ Send(P,M,S(k_6,Kpb(M)))\circ Send(P,M,E(Sign(P,CapRes),k_6))\circ$$
$$Send(P,M,CertS(P))\circ Verify(M,CertS(P),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle),$$

$$Q=\{Auth(P,CA,\langle M,Spb(M)\rangle),Auth(P,CA,\langle M,Kpb(M)\rangle),Auth(M,CA,\langle P,Spb(P)\rangle),$$
$$Auth(M,P,CapRes)\}.$$

求证:$P\vdash_a Q$

证明:

(1) $Know(M,S(\langle k_4,Acct(C)\rangle,Kpb(P)))$ [前提]

(2) $Know(M,E(Sign(P,CapToken),k_4))$ [前提]

(3) $Know(M,CertS(M))$ [前提]
(4) $Know(M,CertK(M))$ [前提]
(5) $Know(M,Kpb(P))$ [前提]
(6) $Know(P,CertS(P))$ [前提]
(7) $Know(M,Spb(P))$ [前提]
(8) $Generate(M,CapReq)$ [动作]
(9) $Know(M,CapReq)$ (8)[R-2]
(10) $Know(M,S(CapReq,Spv(M)))$ (9)[2-2][4-2]
(11) $Know(M,Sign(M,CapReq))$ (9)(10)[定义]
(12) $Generate(M,k_5)$ [动作]
(13) $Know(M,k_5)$ (12)[R-2]
(14) $Know(M,E(Sign(M,CapReq),k_5))$ (11)(13)[1-1]
(15) $Know(M,S(k_5,Kpb(P)))$ (5)(13)[1-2]
(16) $Send(M,P,S(k_5,Kpb(P)))$ [动作]
(17) $Send(M,P,E(Sign(M,CapReq),k_5))$ [动作]
(18) $Send(M,P,S(\langle k_4,Acct(C)\rangle,Kpb(P)))$ [动作]
(19) $Send(M,P,E(Sign(P,CapToken),k_4)))$ [动作]
(20) $Send(M,P,CertS(M))$ [动作]
(21) $Send(M,P,CertK(M))$ [动作]
(22) $Know(P,S(k_5,Kpb(P)))$ (16)[R-1]
(23) $Know(P,E(Sign(M,CapReq),k_5))$ (17)[R-1]
(24) $Know(P,S(\langle k_4,Acct(C)\rangle,Kpb(P)))$ (18)[R-1]
(25) $Know(P,E(Sign(P,CapToken),k_4))$ (19)[R-1]
(26) $Know(P,CertS(M))$ (20)[R-1]
(27) $Know(P,CertK(M))$ (21)[R-1]
(28) $Verify(P,CertS(M),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)$ (26)[2-5][动作]
/* 如果检验失败,$P$ 没有在 PKI 树中找到 $CARoot$,则付费网关拒绝付费 */
(29) $ISVerified(P,CA,CertS(M))$ (28)[R-6]
(30) $Auth(P,CA,\langle M,Spb(M)\rangle)$ (29)[7-1]
(31) $Verify(P,CertK(M),\langle CA,X_2,\ldots,X_{n-1},CARoot\rangle)$ (27)[2-5][动作]
/* 如果检验失败,$P$ 没有在 PKI 树中找到 $CARoot$,则付费网关拒绝付费 */
(32) $ISVerified(P,CA,CertK(M))$ (31)[R-6]
(33) $Auth(P,CA,\langle M,Kpb(M)\rangle)$ (32)[7-2]
(34) $Know(P,Spb(M))$ (26)[定理 3]
(35) $Know(P,Kpb(M))$ (27)[定理 4]
(36) $Know(P,k_5)$ (22)[2-1][3-2]
(37) $Know(P,Sign(M,CapReq))$ (23)(36)[3-1]
(38) $Auth(P,M,CapReq)$ (34)(37)[定理 2]
(39) $Know(P,\langle k_4,Acct(C)\rangle)$ (24)[2-1][3-2]
(40) $Know(P,k_4)$ (39)[4-1]
(41) $Know(P,Acct(C))$ (39)[6-1]
(42) $Know(P,Sign(P,CapToken))$ (25)(40)[3-1]
(43) "付费网关 $P$ 确信商家 $M$ 的获得请求 $CapReq$ 和获得令牌 $CapToken$ 之间的一致性"

(44)　"付费网关 $P$ 通过金融网向持卡人 $C$ 的金融机构发送获得请求"

(45)　$Generate(P,CapRes)$　[动作]

(46)　　$Know(P,CapRes)$　(45)[R-2]

(47)　　$Know(P,S(H(CapRes),Spv(P)))$　(46)[2-2][4-2]

(48)　　$Know(P,Sign(P,CapRes))$　(46)(47)[定义]

(49)　$Generate(P,k_6)$　[动作]

(50)　　$Know(P,k_6)$　(49)[R-2]

(51)　　$Know(P,E(Sign(P,CapRes),k_6))$　(48)(50)[1-1]

(52)　　$Know(P,S(k_6,Kpb(M)))$　(35)(50)[1-2]

(53)　$Send(P,M,S(k_6,Kpb(M)))$　[动作]

(54)　$Send(P,M,E(Sign(P,CapRes),k_6))$　[动作]

(55)　$Send(P,M,CertS(P))$　[动作]

(56)　　$Know(M,S(k_6,Kpb(M)))$　(53)[R-1]

(57)　　$Know(M,E(Sign(P,CapRes),k_6))$　(54)[R-1]

(58)　　$Know(M,CertS(P))$　(55)[R-1]

(59)　$Verify(M,CertyS(P),(CA,X_2,\ldots,X_{n-1},CARoot))$　(58)[2-5][动作]

　　/* 如果检验失败,$M$ 没有在 PKI 树中找到 $CARoot$,则商家 $M$ 拒绝获得付费 */

(60)　　$IsVerified(P,CA,CertS(P))$　(59)[R-6]

(61)　　$Auth(M,CA,(P,Spb(P)))$　(60)[7-1]

(62)　　$Know(M,k_6)$　(56)[2-1][3-2]

(63)　　$Know(M,Sign(P,CapRes))$　(57)(62)[3-1]

(64)　　$Know(M,Spb(P))$　(58)[定理 3]

(65)　　$Auth(M,P,CapRes)$　(63)(64)[定理 2]

(66)　"商家 $M$ 存储获得回答,使它与从请求者获得的付费一致"

以上用文献[1]中扩展的 NDL 逻辑框架对 SET 的付费业务流程进行了全面的验证.从整个证明过程可以看出,NDL 在验证安全协议的安全性质上的重要作用.我们通过把一些复杂的过程抽象成简单的逻辑符号,对 SET 进行了严格的证明.但也应该看到,在证明中假设各参与方都是合法用户,排除了他们非法获得或泄露消息的可能性,对此如果没有相应的防范措施,一旦发生意外,要追究责任时,各方都可以抵赖,拒绝承担责任.采取什么样的方法来摆脱这种状况是我们迫切需要解决的问题.

### 1.6　验证逻辑的 Prolog 程序实现

由于逻辑推导所涉及的东西大多是符号和规则,此推理过程若要由人来完成,那么无论在人力上,还是在时间上均是一个极大的浪费.但逻辑推理具有易在机器上验证的特点,因此我们编制了基于 NDL 逻辑框架的 Prolog"安全协议验证系统",其规则系统即为 NDL 的公理和规则,用户可以根据增加的需要输入前提,来达到验证安全协议是否存在漏洞的目的.

## 2　"$k$ out of $n$"问题

### 2.1　"$k$ out of $n$"的定义

"$k$ out of $n$"问题的含义是指,在协议执行过程中有 $n$ 个参与方,每个参与方都知道自己权限范围内的秘密信息.在正常情况下,参与各方只知道有限的信息,谁都无法获得超出自己权限的商业机密,即 SET 的付费流程是安全可靠的.但是,如果其中有 $k$ 方聚在一起,把他们所知道的秘密凑到一起形成一把完整的"钥匙",就可以非法窃取商业机密.

当金融机构发现问题,想追究他们的法律责任时,任何一方都可以抵赖.因为他们只用自己知道的合法的、

有限的信息,是无法知道这些超出自己权限范围的商业机密的.这就涉及到当今电子商务中一个很重要的问题,即如何在电子交易中实现"可防抵赖"和"可追踪"两种安全保障机制,从而防止该问题的发生.

注意,"$k$ out of $n$"问题是在信任 RSA,DES 算法的安全强度,而忽略算法的数学细节的前提下进行讨论的.

### 2.2 SET 中参与方概述

Issuer(发行方):它是一个金融机构,由它发行付费卡,确定持卡人帐号.它保证了按付费卡品牌和地方法律为认证的事务付费.

Acquirer(获得方):它是一个金融机构,确认商家的帐户,处理付费卡认证和付费.

Cardholder(持卡人):它的付费卡由 Issuer 发行,SET 保证在持卡人与商家交易的过程中,付费卡的帐目信息保持可靠.

Merchant(商家):它提供所要卖的商品或要求付费的服务,能接受付费卡的商家必须与 Acquirer 有关系.

Payment Gateway(付费网关):它是由 Acquirer 或处理商家付费信息(包括从持卡人得到的付费指令)的指定第三方操作的设备.

Third Parties(第三方):Issuer 和 Acquirers 有时会选择把付费卡业务分配给第三方处理.

### 2.3 对持卡人注册阶段的分析

$$Message\ C:\langle Spv(C),Spb(C)\rangle,k_1,k_2,k_3,Acct(C),$$

$$Message\ CA:\ Spv(CA),Spb(C),k_1,k_2,k_3,Acct(C).$$

通过该阶段各参与方所知道消息的分析和上边对它所做的逻辑认证可以看出,持卡人的帐户信息是能否保证持卡人注册阶段的安全性的关键环节.从帐户信息是否泄露的角度出发,安全性问题又可分为以下 3 个方面:

- $C$ 更改了密钥 $Spv(C)$,$k_1$,$k_2$ 和 $k_3$,但没有泄露帐户信息.
- $C$ 或 CA 向第三者泄露帐户信息,但没有更改密钥.
- $C$ 更改了密钥,且向第三者泄露了帐户信息.

如果发生第 1 种情况,因为密钥已经改变,$C$ 可以抵赖说他没有向 CA 申请过证书.如果发生第 2 种情况,因为无论是 $C$ 或 CA 泄露的消息,第三者都可以成功地冒充持卡人从 CA 获得合法的签名公钥证书,但是要追究责任,$C$ 和 CA 都可以抵赖,指责对方泄露了消息.对于第 3 种情况,它与第 1 种情况唯一不同之处在于,第三者也能冒充 $C$ 申请到合法的签名证书.因此,无论上面哪一种情况,该阶段的安全性都无法保障,那么为了能够保证电子商务中的安全性,整个 SET 付费流程中必须能够保证上面所提到的"防抵赖"性和各参与方活动的"可追踪"性两种安全机制的实施.

## 3 结束语

以上对 SET 协议进行了全面的逻辑验证,说明了 NDL 的重要性,从证明中也发现了 SET 中确实存在漏洞.该证明过程使整个 SET 的验证变得非常简洁,它同时也为用 Prolog 编程模拟该验证过程奠定了基础.本文的重点是给出整个 SET 流程的证明.只是简单地就对"持卡人注册"阶段进行的讨论指出了"$k$ out of $n$"问题的存在.我们将来的目标就是针对该问题进行分析,结合实际的金融业务中可能会发生的情况,通过实现"防抵赖"和"可追踪"两种安全机制来保障 SET 中的信息安全.我们将在以后的工作中对 SET 协议中存在的"$k$ out of $n$"问题进行详细的讨论.

## 参考文献

1 Kenichi HAYASHI, Eiji OKAMOTO, Masabiro MAMBO. Proposal of user identification scheme using mouse. In: Han Yong-fei, Okamoto Tatsuaki, Qing Si-han eds. Proceedings of the ICICS'97. LNCS 1334, Berlin: Springer-Verlag, 1997. 424~434

2 Chang Li-wu, Moskowitz T S. Critical analysis of security in voice hiding techniques. In: Han Yong-fei, Okamoto Tatsuaki, Qing Si-han eds. Proceedings of the ICICS'97. LNCS 1334, Berlin: Springer-Verlag, 1997. 203~216

3 Van der Merwe Jaco, Von Solms S H. Electronic commerce with secure intelligent trade agents. In: Han Yong-fei,

Okamoto Tatsuaki, Qing Si-han eds. Proceedings of the ICICS'97. LNCS 1334, Berlin: Springer-Verlag, 1997. 452～462

4 Kailar R. Accountability in electronic commerce protocols. Information and Communications, 1996,22(5):313～328

5 Kailar R, Gligor V. On belief evolution in authentication protocols. In: Proceedings of the 4th IEEE Computer Security Foundations Workshop. Los Alamitors, CA: IEEE Computer Society Press, 1991

6 Chen Qing-feng, Bai Shuo, Wang Ju *et al*. The secure electronic transactions protocol and its logical verification with non-monotomic dynamic logic. Journal of Software, 2000,11(2):240～250
(陈庆锋,白硕,王驹等.电子商务安全协议及其非单调动态逻辑验证.软件学报,2000,11(2):240～250)

7 SET Secure Electronic Transaction Specification, Book 1: Business Description. Version 1.0, 1997

8 Bai Shuo, Sui Li-ying, Chen Qing-feng *et al*. The verification logic for secure protocols. Journal of Software, 2000,11(2):213～221
(白硕,隋立颖,陈庆锋等.安全协议的验证逻辑.软件学报,2000,11(2):213～221)

## Logical Verification of Secure Electronic Transactions Protocol

CHEN Qing-feng[1,2] WANG Ju[3] BAI Shuo[1] ZHANG Shi-chao[2] SUI Li-ying[1]

[1](*National Research Center for Intelligent Computers Beijing* 100080)

[2](*Department of Mathemetics and Computer Science Guangxi Normal University Guilin* 541005)

[3](*Institute of Software The Chinese Academy of Sciences Beijing* 100080)

**Abstract** In the previous work, some segments of secure electronic transactions (SET) are verified, and some potential problems in SET are also discussed. Based on these, all transactions in SET are strictly logically verified in this paper. Though this formal logic verification, certain problems are found to do exist in SET. Solutions to these problems are also discussed.

**Key words** Information security, logical verification, accountability, traceable.