ICAO

Doc 9303

机读旅行证件

第八版, 2021年

第 11部分：机读旅行证件安全机制

经秘书长批准并由其授权出版

国际民用航空组织

ICAO

Doc 9303

机读旅行证件

第八版, 2021年

第 11部分：机读旅行证件安全机制

经秘书长批准并由其授权出版

国际民用航空组织

国际民用航空组织分别以中文、阿拉伯文、英文、法文、俄文和西班牙文版本出版
999 Robert-Bourassa Boulevard, Montréal, Quebec, Canada H3C 5H7

下载文件和获取额外信息，请登录 www.icao.int/Security/FAL/TRIP。

Doc 9303 号文件 一《机读旅行证件》
第 11 部分 一 机读旅行证件安全机制
订购编号：9303P11
ISBN 978-92-9265-525-9（印刷版）

修订

《产品和服务目录》的补篇中公布了各项修订；在国际民航组织网站 www.icao.int 上有本目录及其补篇。以下篇幅供记录修订之用。

修订和更正记录

修订		
编号	日期	换页人

更正		
编号	日期	换页人

本出版物中所用称谓和陈述材料之方式，并不代表国际民航组织对任何国家、领土、城市或地区或其当局的法律地位，或就其边境或疆界的划分，表达了任何意见。

目录

1. 范围

Doc 9303号文件第11部分明确了一些规范，使各国和供应商能够实施可提供非接触式集成电路访问的电子机读旅行证件（“eMRTDs”）的密码安全功能。规定了密码协议以：

- 防止非法浏览非接触式集成电路数据；
- 防止非接触式集成电路和阅读器之间通信被窃听；
- 依据第12部分描述的公钥基础设施（PKI）认证在非接触式集成电路上存储的数据；
- 提供非接触式集成电路自身的认证。

第八版Doc 9303号文件纳入了可选旅行记录、签证记录和附加生物特征应用程序（称为LDS2应用程序）的规范，作为电子机读旅行证件的可选扩展。Doc 9303号文件的这一部分包括必要的扩展访问控制协议，以保护各个LDS2应用程序的数据写入和读取。这些访问控制协议也可用于保护电子机读旅行证件应用程序中的第二生物特征。

对非接触式集成电路储存数据进行认证是实现将集成电路用于人工和/或自动查验的基本安全特征。因此这一措施是必要的。

执行一项协议是必要的，以防止非法浏览存储在非接触式集成电路上的数据，以及防止集成电路和终端之间的通信被窃听。

可以选择性地执行其他协议，允许签发国或者签发机构根据国家规章/需求确定一套必要的安全特征。

本部分应结合Doc 9303号文件的以下各个部分进行阅读：

- 第1部分 — 引言；
- 第10部分 — 在非接触式集成电路中存储生物特征和其他数据的逻辑数据结构（LDS）；和
- 第12部分 — 机读旅行证件的公钥基础设施。

2. 假定和符号

假定本文件的读者熟悉公钥密码技术和公钥基础设施所提供的概念和机制。

虽然使用公钥密码技术增加了一些实施电子机读旅行证件的复杂性，但这项技术是有附加价值的，因为它为一线的边境控制点提供了确定电子机读旅行证件真伪的另一种措施。假定使用这种技术并不是确定真伪的唯一措施，因此，不应该将其作为唯一的决定因素来依赖。

如果由于诸如证书被撤销或无效的签名核证等造成非接触式集成电路上的数据不能使用，或者如果非接触式集成电路被有意留作空白（见Doc 9303号文件第10部分第4.5.4节），电子机读旅行证件未必就失效。在这些情况下，接收国可以依赖证件的其他安全特征达到验证的目的。

2.1 对电子机读旅行证件芯片和终端的要求

Doc 9303号文件的本部分规定了电子机读旅行证件芯片（或相当的集成电路）和终端（或查验系统）的实施要求。尽管电子机读旅行证件芯片必须符合Doc 9303号文件第1部分所述的术语的要求，对终端的要求可被视作指导，即只有在终端符合这些要求的情况下，才能确保电子机读旅行证件芯片和终端的互操作性，否则与电子机读旅行证件芯片的交互将会失败，或者电子机读旅行证件芯片的性能不稳定。总之，电子机读旅行证件芯片不必强制实施与终端相关的要求，除非电子机读旅行证件芯片的安全性受到直接影响。

2.2 符号

下列符号用于以一种独立算法方式表示密码学元件：

- 用对称密钥K加密明文S：**E**（K,S）；
- 用对称密钥K解密密文C：**D**（K,C）；
- 用**H**(*m*)表示计算消息m的散列值的运算。
- 用对称密钥K计算消息M的消息认证码：**MAC**（*K*,*M*）；
- 基于非对称密钥对（SK, PK）和（SK', PK'）以及域参数D的密钥协商：**KA**（SK, PK', D）/**KA**（SK', PK, D）；
- 从共享秘密S派生密钥：**KDF**（*S*）；
- 用密钥SK_{IFD}对消息m进行签名表示为 s = Sign(SK_{IFD}, *m*)；
- 使用公钥PK_{IFD}和消息m验证生成的签名s：Verify(PK_{IFD}, *s*, *m*)；
- 计算公钥PK的压缩值表示：Comp(*PK*)。

3. 保护电子数据的安全

除了通过数字签名和芯片访问控制进行被动认证外，签发国或签发机构还可以选择其他安全措施，采用更加复杂的方式保护非接触式集成电路及其数据。

访问电子机读旅行证件包括下列步骤：

1. 获得电子机读旅行证件非接触式集成电路的访问权限（第4节）
2. 数据认证（第5节）
3. 芯片认证（第6节）
4. 其他访问控制机制（第7节）
5. 读取数据（见Doc 9303号文件第10部分）。

不同步骤有可供使用的不同协议。电子机读旅行证件的具体配置由签发国或签发机构选定。可以适当地结合表1给出的选择，根据签发者的要求实现其他的安全保障。

不同样式电子机读旅行证件的查验程序在附录J中作了说明。

表1　保护电子数据的安全（简表）

方法	非接触式集成电路	查验系统	优点	说明
基线安全方法				
被动认证（第5.1节）	m	m	证明SO_D和LDS的内容真实且未被修改。	不能防止完全拷贝或者芯片替换。 不能防止未经授权的访问。 不能防止非法浏览。
高级安全方法				
传统MRZ（OCR-B）和基于芯片的MRZ（LDS）的比较	n/a	o	证明芯片内容和物理的eMRTD匹配。	增加（少许的）复杂性。 不能防止对芯片和传统证件的完全拷贝。
主动认证（第6.1节）	o	o	防止拷贝SO_D，并证明它是从真实的芯片中读取的。 证明非接触式集成电路没有被替换。	不能防止未经授权的访问。 增加复杂性。 需要为LDS2进行芯片认证
芯片认证（第6.2节）	o/c	o		

方法	非接触式集成电路	查验系统	优点	说明
基本访问控制（BAC）（第4.3节）	c（另见第4.1节）	m（另见第4.1节）	防止非法浏览和误用。 防止窃听电子机读旅行证件和查验系统之间的通信（当用来建立加密的会话信道时）。	不能防止完全拷贝或集成电路替换（还需要复制传统证件）。 增加复杂性。至少BAC或PACE之一应得到电子机读旅行证件的支持。LDS2需要PACE。PACE可比BAC更好地防窃听。还可参见附录A。
口令认证连接确立协议（PACE）（第4.4节）	r/c（另见第4.1节）	m（另见第4.1节）		
终端认证（第7.1节）	o/c	o	防止对敏感数据进行非授权的访问。 防止非法浏览敏感数据。	需要附加的密钥管理。 不能防止完全拷贝或集成电路替换（还需要复制传统证件）。 增加复杂性。LDS2需要终端认证。
数据加密（第7.2节）	o	o	保护其他生物特征的安全。 不需要芯片处理器。	需要复杂的解密密钥管理。 不能防止完全拷贝或集成电路替换。 增加复杂性。
m = 必要的，r = 建议的，o = 可选的，c = 有条件的，n/a = 不适用。				

注：关于实施BAC和PACE的非接触式集成电路配置的详情见第4节。

表1列出的高级安全方法的实施不影响国际民航组织合规性。

4. 访问非接触式集成电路

电子机读旅行证件增加非接触式集成电路而无访问控制会带来两种新的攻击可能性：

- 在未经授权的情况下对存储在非接触式集成电路中的数据进行电子阅读（非法浏览）；
- 非接触式集成电路和阅读器之间的未加密通信可在几米之内被窃听。

虽然有物理措施能够应对非法浏览（例如，护照封面内使用金属网作为防护），但解决不了窃听的问题。因此，签发国或签发机构应实施一种芯片访问控制机制，即电子机读旅行证件持有人知道存储在非接触式集成电路中的数据在被安全读取的一种访问控制机制。这种芯片访问控制机制能够防止非法浏览，也能够防止窃听。

受芯片访问控制机制保护的非接触式集成电路拒绝对其内容进行访问，除非查验系统能够证明对非接触式集成电路的访问是被授权的。这种证明在密码协议中给出，从中查验系统证明其知道源于实际证件的信息。

查验系统必须首先得到这方面的信息才能够阅读非接触式集成电路。信息必须从电子机读旅行证件（例如从机读区）中以光学/目视的方式获得。在不能进行机器阅读信息时，查验员还必须能够以人工方式将该信息输入到查验系统。

假定实际证件上的信息不能从未检视的证件上得到（例如，因为它们要通过以光学方式阅读机读区获得），可以被认为电子机读旅行证件是在知情的情况下送交查验的。由于信道加密，要对通信进行窃听需要付出巨大的努力。

本节规定了两种芯片访问控制机制：

- 基本访问控制（BAC，第4.3节），完全基于对称密码；
- 口令认证连接确立协议（PACE，第4.4节），使用非对称密码以提供熵更高的会话密钥。

另见附录A关于会话密钥熵的其他信息。

4.1　合规的配置

下列配置符合本规范：

- 仅实施BAC的电子机读旅行证件芯片；
- 实施PACE和BAC的电子机读旅行证件芯片；
- 从2018年1月起，仅实施PACE的电子机读旅行证件芯片。

注：未来，BAC可能不再受人们青睐。在这种情况下，PACE将成为默认访问控制机制。

合规的查验系统必须支持所有合规的电子机读旅行证件配置。如果电子机读旅行证件支持PACE和BAC两种安全机制，则查验系统在同一会话中应使用PACE或者BAC，但不应同时使用。

注 1：Doc 9303号文件的先前版本允许电子机读旅行证件芯片实施无芯片访问控制（“普通电子机读旅行证件”）。这在第八版中已被弃用。尽管如此，合规查验系统必须支持没有芯片访问控制的电子机读旅行证件。

注 2：为了访问LDS2应用程序，集成电路必须要求执行PACE。

4.2 芯片访问过程

认证查验系统的芯片访问过程包括如下步骤。

1. **读取EF.CardAccess** **（必要的）**

   如果电子机读旅行证件支持PACE，电子机读旅行证件芯片在文档EF.CardAccess中必须提供用于PACE的参数。

   如果EF.CardAccess可用，查验系统应读取文档EF.CardAccess（参见第9.2.11节）以确定电子机读旅行证件芯片支持的参数（即对称密码、密钥协商算法、域参数和映射）。查验系统可选择这些参数中的任何一个。

   如果文档EF.CardAccess不可用，或者不包括PACE参数，查验系统应该利用基本访问控制读取电子机读旅行证件（跳至步骤4）。

2. **读取EF.DIR** **（可选的）**

   查验系统可以读取EF.DIR（如果存在）以检索电子机读旅行证件芯片上存在的应用程序列表。

3. **PACE** **（有条件的）**

   建议该步骤在电子机读旅行证件芯片支持PACE机制的情况下进行。如果打算访问LDS2应用程序，则需要此步骤。

   - 查验系统应该根据MRZ（机读区）派生密钥K_{π}。如果查验系统知道CAN（卡访问号），则可使用CAN代替MRZ。
   - 电子机读旅行证件应将MRZ作为PACE的口令。除MRZ外，它也可以用CAN作为口令。
   - 查验系统和电子机读旅行证件芯片使用K_{π}互相认证，并派生会话密钥KS_{Enc}和KS_{MAC}。应使用第4.4节所述的PACE协议。

   如果成功，电子机读旅行证件芯片执行下列操作：

   - 启动安全通讯。
   - 允许访问不敏感数据（例如，电子机读旅行证件应用程序的EF.DG1、EF.DG2、EF.DG14、EF.DG15等和证件安全对象。“敏感数据”的定义参见Doc 9303号文件第1部分）。
   - 限制访问权限以要求安全通讯。

   查验系统必须使用EF.DG14或EF.CardSecurity验证EF.CardAccess内容的真实性，和使用EF.CardSecurity验证EF.DIR（如果存在并读取）内容的真实性。

   注：如果电子机读旅行证件芯片上不存在LDS2应用程序，则EF.CardSecurity可能不包含EF.DIR的安全副本。

4. **基本访问控制** **（有条件的）**

如果电子机读旅行证件实施芯片访问控制，并且未使用PACE，这一步骤是必要的。如果成功实施了PACE，或者电子机读旅行证件未实施芯片访问控制，则这一步骤可跳过。

必须在执行基本访问控制之前，选择电子机读旅行证件应用程序。

- 查验系统应该从机读区派生出证件基本访问密钥（K_{Enc}和K_{MAC}）。
- 查验系统和电子机读旅行证件芯片使用证件基本访问密钥相互认证，并导出会话密钥KS_{Enc}和KS_{MAC}。

如果成功，电子机读旅行证件芯片实施下列操作：

- 启动安全通讯。
- 允许访问不敏感数据（例如，电子机读旅行证件应用程序的EF.DG1、EF.DG2、EF.DG14、EF.DG15等和证件安全对象）。
- 限制访问权限以要求安全通讯。

注：作为芯片访问程序的结果，当前DF可以是主文件（如果使用PACE）或电子机读旅行证件应用程序（如果使用 BAC）。

4.3 基本访问控制

4.3.1 协议规范

认证和密钥建立是根据[ISO/IEC 11770-2]密钥建立机制6，利用分组密码3DES [FIPS 46-3]，通过三轮询问-应答协议实现的。密码校验和是根据[ISO/IEC 9797-1]消息认证码算法3计算，并附加到加密文本上。必须使用第4.3.3节中描述的运算方式。被交换的随机数必须为8个字节，被交换的密钥材料为16个字节。接口设备（即查验系统）和非接触式集成电路不得使用特异标识符作为随机数。

具体地讲，接口设备（IFD）和集成电路（IC）应实施下列步骤：

1) IFD发送命令GET CHALLENGE，请求询问RND.IC。IC生成并返回随机数RND.IC。

2) IFD执行下列操作：

   a) 生成一个随机数RND.IFD和密钥组件K.IFD。

   b) 生成S = RND.IFD || RND.IC || K.IFD。

   c) 计算密文E_{IFD} = **E**（$K_{Enc,}$ S）。

d) 计算校验和M_{IFD} = **MAC**（$K_{MAC,}$ E_{IFD}）。

e) 发送命令EXTERNAL AUTHENTICATE，该指令利用数据E_{IFD} || M_{IFD} 实现相互认证的功能

3) IC执行下列操作：

a) 检验密文E_{IFD}的校验和M_{IFD}。

b) 破译密文E_{IFD}。

c) 从S中提取RND.IC，并检查IFD是否返回了正确值。

d) 生成密钥组件K.IC。

e) 生成R = RND.IC || RND.IFD || K.IC。

f) 计算密文E_{IC} = **E**（K_{Enc}, R）。

g) 计算校验和M_{IC} = **MAC**（K_{MAC}, E_{IC}）。

h) 使用数据E_{IC} || M_{IC}发送响应。

4) IFD执行下列操作：

a) 检查密文E_{IC}的校验和M_{IC} 。

b) 破译密文E_{IC}。

c) 从R中提取RND.IFD，并检查IC是否返回了正确值。

5) IFD和IC以（K.IC异或K.IFD）作为共享秘密，使用第9.7.1节和第9.7.4节中描述的密钥派生机制导出会话密钥KS_{Enc}和KS_{MAC}。

4.3.2 查验过程

当实施基本访问控制的电子机读旅行证件提供给查验系统时，使用以光学或视读方式读取的信息派生证件基本访问密钥（K_{Enc}和K_{MAC}），以便访问非接触式集成电路，并且在电子机读旅行证件非接触式集成电路与查验系统之间建立安全通讯信道。

安全信道建立后，支持基本访问控制的电子机读旅行证件非接触式集成电路必须对非授权的阅读企图，即未应用安全通讯的情况下发送的阅读企图（包括对逻辑数据结构中的（被保护的）文件的选择），作出“没有满足安全状态”（0x6982）的响应。如果集成电路在安全信道接收到普通命令SELECT，即未应用安全通讯，集成电路应中断安全信道。如果在安全信道建立之前或者安全信道中断之后发出普通命令SELECT，则集成电路返回0x6982和0x900，此二者都是国际民航组织的合规响应。

为认证查验系统，必须实施下述步骤：

1) 查验系统读取“MRZ_information”。“MRZ_information”由证件号码、出生日期和到期日串联构成，并包括它们各自的校验位（Doc 9303号文件第4部分、第5部分或第6部分分别对TD3、TD1和TD2这三种尺寸的证件的校验位做了描述）。“MRZ_information”通过使用OCR-B阅读器从机读区读取，或者通过手工输入。在这种情况下，手工输入的信息必须与MRZ中显示的信息保持一致。“MRZ_information”的SHA-1散列值中最重要的16个字节被用作密钥种子，通过第9.7.2节描述的密钥派生机制导出证件基本访问密钥。

2) 查验系统和电子机读旅行证件非接触式集成电路相互认证和导出会话密钥。必须使用上文描述的认证和密钥建立协议。

3) 在成功执行认证协议后，IFD和IC以（K.IC 异或K.IFD）作为共享秘密，通过使用第9.7.1节和第9.7.4节描述的密钥导出机制导出会话密钥KS_{Enc}和KS_{MAC}。随后的通信必须通过第9.8节中描述的安全通讯加以保护。

4.3.3 密码规范

4.3.3.1 询问和应答的加密

使用[ISO/IEC 11568-2]中零初始化向量（即0x00 00 00 00 00 00 00 00）的双密钥3DES CBC模式来计算E_{IFD}和E_{IC}。在执行EXTERNAL AUTHENTICATE 命令时，禁止对输入数据进行填充。

4.3.3.2 询问和应答的认证

使用[ISO/IEC 9797-1]中的消息认证码算法3计算密码校验和M_{IFD}和M_{IC}，其中用到分组密码DES、零初始化向量（8字节）和[ISO/IEC 9797-1]中的填充方法2。消息认证码长度必须为8字节。

4.3.4 应用协议数据单元

使用具有相互认证功能的命令GET CHALLENGE和EXTERNAL AUTHENTICATE进行基本访问控制。命令按[ISO/IEC 7816-4]中的规定编码。

4.3.4.1 获得询问

命令		
CLA		上下文特定
INS	0x84	GET CHALLENGE
P1/P2	0x0000	—
数据		缺失
响应		
数据	随机数	
状态字节	0x9000	常规处理 成功生成并发送随机数。
	其他	取决于操作系统的错误 不能传送随机数。

4.3.4.2 外部认证

命令			
CLA		上下文特定	
INS	0x82	EXTERNAL AUTHENTICATE	
P1/P2	0x0000	—	
数据		命令数据E_{IFD} || M_{IFD}	必要的
响应			
数据		响应数据E_{IC} || M_{IC}	必要的
状态字节	0x9000	常规处理 已成功执行协议。	
	其他	取决于操作系统的错误 协议失败。	

4.4 PACE

PACE是口令认证的Diffie-Hellman密钥协商协议，提供eMRTD芯片和查验系统的安全通信和基于口令的认证（即eMRTD芯片和查验系统共享同一个口令π）。

PACE在弱（短）口令的基础上确立eMRTD芯片和查验系统之间的安全通讯。在主文件中确立安全环境。协议使eMRTD芯片能够核验查验系统访问存储数据的权限并具有下列特性：

- 不受口令强度影响，提供强会话密钥。
- 用来认证查验系统的口令的熵可能非常低（例如，通常六位数是足够的）。

PACE使用的密钥K_π是由密钥派生函数**KDF$_\pi$**（参见第9.7.3节）从口令派生出来的。对于有全球互操作性的eMRTD，可用下列两种口令和相应的密钥：

- MRZ：由K_π = **KDF$_\pi$**（MRZ）定义的密钥K_π 是必要的。类似于BAC，该密钥是从机读区（MRZ）派生而来，即密钥由证件号码、出生日期和到期日派生而来。
- CAN：由K_π = **KDF$_\pi$**（CAN）定义的密钥K_π是可选的。该密钥由卡访问号（CAN）派生而来。CAN是印在证件的数字，必须随机或者伪随机选择（例如，使用加密性强的伪随机功能）。Doc 9303号文件第4部分、第5部分和第6部分对CAN域作了规范。

注：与MRZ（证件号码、出生日期、到期日）相反，CAN的优点是便于人工键入。

作为协议执行的一部分，**PACE**支持不同的映射：

- 基于Diffie-Hellman密钥协商的通用映射；
- 基于域元素到密码群的嵌入映射的合成映射；
- 芯片认证映射扩展通用映射，并将芯片认证纳入PACE协议。

如果芯片支持芯片认证映射，则芯片也必须支持通用映射或合成映射中的一种以及芯片认证。这意味着对于支持PACE的查验系统来说，仅支持通用映射和集成映射是必要的。支持芯片认证映射是可选的。

4.4.1　协议规范

查验系统从文档EF.CardAccess（参见第9.2.11节）读取eMRTD芯片支持的PACE参数，选取要使用的参数，随后执行协议。

应使用下列命令：

- Doc 9303号文件第10部分规定的READ BINARY；
- 第4.4.4.1节中规定的MSE:Set AT（具有设定认证模板功能的MANAGE SECURITY ENVIRONMENT命令）；

- 查验系统和eMRTD芯片应使用第4.4.4.2节中规定的一系列通用认证命令实施下列步骤：

  1) eMRTD芯片随机均匀选取随机数s，将该随机数加密为z = **E**（K_π,s），其中K_π = **KDF**$_\pi$ (π)是从共享口令π中派生来的，并将密文z发送给查验系统。

  2) 查验系统在共享口令π的帮助下，恢复纯文本s = **D**（K_π,z）。

  3) eMRTD芯片和查验系统两者都执行下列步骤：

     a) 交换随机数映射必需的其他数据：

        i) 对于通用映射，eMRTD芯片和查验系统交换临时密钥公钥。

        ii) 对于合成映射，查验系统向eMRTD芯片发送另一个随机数。

     b) 按照第4.4.3.3节描述，计算临时域参数D = **Map**（D_{IC},s,…）。

     c) 在临时域参数的基础上进行匿名Diffie-Hellman密钥协商（参见第9.6节），生成共享秘密K = **KA**（$SK_{DH,IC}$, $PK_{DH,IFD}$,D）= **KA**（$SK_{DH,IFD}$,$PK_{DH,IC}$,D）。

     d) 在Diffie-Hellman密钥协商过程中，集成电路和查验系统应验证两个公钥$PK_{DH,IC}$和$PK_{DH,IFD}$是不同的。

     e) 按照第9.7.1节描述，导出会话密钥KS_{MAC} = **KDF**$_{\mathrm{MAC}}$(K)和KS_{Enc} = **KDF**$_{\mathrm{Enc}}$(K)。

     f) 按照第4.4.3.4节描述，交换并核验认证令牌T_{IFD} = **MAC**（KS_{MAC},$PK_{DH,IC}$）和T_{IC} = **MAC**（KS_{MAC},$PK_{DH,IFD}$）。

  4) 在一定条件下，电子机读旅行证件芯片计算芯片认证数据CA_{IC}，将数据加密A_{IC} = **E**（KS_{Enc}, CA_{IC}），并将加密数据发送给终端（参见第4.4.3.5.1节）。终端解密A_{IC}，通过使用经恢复的芯片认证数据CA_{IC}来证实芯片的真实性（参见第4.4.3.5.2节）。

协议的简化版本另见图1。

芯片表示为集成电路		**查验系统表示为IFD**
静态域参数D_{IC} 选择随机数s 计算$z = \mathbf{E}(K_\pi,s)$		
	— z →	
		计算$s = \mathbf{D}(K_\pi,z)$
	←**Map**的其他数据 →	
$D = \mathbf{Map}(D_{IC},s,\ldots)$ 选择随机临时密钥对 $(SK_{DH,IC}, PK_{DH,IC},D)$		$D = \mathbf{Map}(D_{IC},s,\ldots)$ 选择随机临时密钥对 $(SK_{DH,IFD},PK_{DH,IFD},D)$
	← $PK_{DH,IC}, PK_{DH,IFD}$→	
检查$PK_{DH,IC} \neq PK_{DH,IFD}$ $K = \mathbf{KA}(SK_{DH,IC}, PK_{DH,IFD},D)$ 计算$T_{IC} =$ $\mathbf{MAC}(KS_{MAC},PK_{DH,IFD})$ ［计算CA_{IC}并按 $A_{IC} = \mathbf{E}(KS_{Enc}, CA_{IC})$加密］		检查$PK_{DH,IC} \neq PK_{DH,IFD}$ $K = \mathbf{KA}(SK_{DH,IFD},PK_{DH,IC},D)$ 计算$T_{IFD} =$ $\mathbf{MAC}(KS_{MAC},PK_{DH,IC})$
	← T_{IC}, T_{IFD} → [— A_{IC} →]	
核证T_{IFD}		核证T_{IC} ［解密A_{IC}并认证芯片］

图1　PACE

4.4.2　安全状态

支持PACE的eMRTD芯片应对非授权的阅读企图（包括对逻辑数据结构中的（被保护的）文件的选择）作出“没有满足安全状态”（0x6982）的响应。

注：该规范比仅支持BAC的eMRTD的相应规范更具限制性。

如果成功进行了PACE，eMRTD便已核验了所使用的口令。使用派生会话密钥KS_{MAC}和KS_{Enc}启动安全通讯。

4.4.3　密码规范

本节包含规范的密码细节。

签发国或签发机构选择特定算法。查验系统必须支持下列各分节描述的所有组合，但芯片认证映射除外，它属于可选的范围。eMRTD芯片可支持一种以上算法组合。

注：一些算法不可用于芯片认证映射：鉴于安全原因，不再建议使用3DES。无法使用DH密钥交换协议变量来减少终端执行的变量数。

4.4.3.1 DH

关于DH的PACE，必须使用第9.6节和表2的相应的算法和格式。

表2 DH的算法和格式

客体标识符	映射	对称密码	密钥长度	安全通讯	认证令牌
id-PACE-DH-GM-3DES-CBC-CBC	通用	3DES	112	CBC / CBC	CBC
id-PACE-DH-GM-AES-CBC-CMAC-128	通用	AES	128	CBC / CMAC	CMAC
id-PACE-DH-GM-AES-CBC-CMAC-192	通用	AES	192	CBC / CMAC	CMAC
id-PACE-DH-GM-AES-CBC-CMAC-256	通用	AES	256	CBC / CMAC	CMAC
id-PACE-DH-IM-3DES-CBC-CBC	合成	3DES	112	CBC / CBC	CBC
id-PACE-DH-IM-AES-CBC-CMAC-128	合成	AES	128	CBC / CMAC	CMAC
id-PACE-DH-IM-AES-CBC-CMAC-192	合成	AES	192	CBC / CMAC	CMAC
id-PACE-DH-IM-AES-CBC-CMAC-256	合成	AES	256	CBC / CMAC	CMAC

4.4.3.2 ECDH

关于ECDH下的PACE，必须使用出自第9.6节和表3的相应的算法和格式。

应仅使用带有未压缩点的素数曲线。应该使用第9.5.1节中描述的标准化域参数。

表3　ECDH的算法和格式

客体标识符	映射	对称密码	密钥长度	安全通讯	认证令牌
id-PACE-ECDH-GM-3DES-CBC-CBC	通用	3DES	112	CBC / CBC	CBC
id-PACE-ECDH-GM-AES-CBC-CMAC-128	通用	AES	128	CBC / CMAC	CMAC
id-PACE-ECDH-GM-AES-CBC-CMAC-192	通用	AES	192	CBC / CMAC	CMAC
id-PACE-ECDH-GM-AES-CBC-CMAC-256	通用	AES	256	CBC / CMAC	CMAC
id-PACE-ECDH-IM-3DES-CBC-CBC	合成	3DES	112	CBC / CBC	CBC
id-PACE-ECDH-IM-AES-CBC-CMAC-128	合成	AES	128	CBC / CMAC	CMAC
id-PACE-ECDH-IM-AES-CBC-CMAC-192	合成	AES	192	CBC / CMAC	CMAC
id-PACE-ECDH-IM-AES-CBC-CMAC-256	合成	AES	256	CBC / CMAC	CMAC
id-PACE-ECDH-CAM-AES-CBC-CMAC-128	芯片认证	AES	128	CBC / CMAC	CMAC
id-PACE-ECDH-CAM-AES-CBC-CMAC-192		AES	192	CBC / CMAC	CMAC
id-PACE-ECDH-CAM-AES-CBC-CMAC-256		AES	256	CBC / CMAC	CMAC

4.4.3.3　加密和映射随机数

eMRTD芯片应随机均匀选取随机数*s*, 该随机数是长度为*l*的二进制字符串，*l* 是eMRTD芯片选择的相应分组密码**E**()分组长度的倍数。

- 随机数s应使用从口令π派生的密钥$K_\pi = KDF_\pi(\pi)$和设定为全零字符串的初始化向量，按照[ISO/IEC 10116]以CBC模式进行加密。
- 使用特定映射函数**Map**，将随机数s转换为随机发生器。
- 对于合成映射，应随机均匀选取另一个随机数*t* 作为长度为*k* 的二进制字符串，并以明文发送。在这种情况下，*k* 是相应分组密码**E**()的密钥长度，*l* 是使得$l \geq k$的分组密码**E**() 分组长度的最小倍数。

将随机数s或随机数*s*, *t* 映射入密码群，应采用下列映射的一种：

- 通用映射（第4.4.3.3.1节）；
- 合成映射（第4.4.3.3.2节）；
- 芯片认证映射（第4.4.3.3.3节）。

4.4.3.3.1 通用映射

ECDH

函数**Map**:G → Ĝ 被定义为Ĝ = s×G+H，其中，选择<G>中的H，使得$\log_G H$未知。H点应由匿名Diffie-Hellman密钥协商[TR-03111]进行计算,即 $H = \mathbf{KA}(SK_{Map,IC}, PK_{Map,IFD}, D_{IC}) = \mathbf{KA}(SK_{Map,IFD}, PK_{Map,IC}, D_{IC})$。

注：密钥协商算法ECKA通过使用兼容余因子乘法来避免小子群攻击。

DH

函数**Map**:g → ĝ被定义为ĝ =g^s×h，其中，选择<g>中的h，使得$\log_g h$未知。群元素h应按照 $h = \mathbf{KA}(SK_{Map,IC}, PK_{Map,IFD}, D_{IC}) = \mathbf{KA}(SK_{Map,IFD}, PK_{Map,IC}, D_{IC})$，由匿名Diffie-Hellman密钥协商进行计算。

注：必须使用[RFC 2631]描述的公钥验证方法来避免小子群攻击。

4.4.3.3.2 合成映射

ECDH

函数**Map**:G → Ĝ被定义为Ĝ = $f_G(\mathbf{R}_p(s,t))$，其中，$\mathbf{R}_p()$是将八位字符串映射到GF(p)元素的伪随机函数，$f_G()$是将GF(p)元素映射到<G>的函数。随机数t应由查验系统随机选择，并发送给eMRTD芯片。伪随机函数$\mathbf{R}_p()$的描述见下文。函数$f_G()$在[BCIMRT2010]中加以界定。详细描述见附录B。

DH

函数**Map**:g → ĝ被定义为ĝ= $f_g(\mathbf{R}_p(s,t))$，其中，$\mathbf{R}_p()$是将八位字符串映射到GF(p)元素的伪随机函数，$f_g()$是将GF(p)元素映射到<g>的函数。随机数t 应由查验系统随机选择，并发送给eMRTD芯片。伪随机函数$\mathbf{R}_p()$描述见下文。函数$f_g()$被定义为$f_g(x)=x^a \bmod p$，a = (p-1)/q是余因子。实施中必须检查ĝ ≠ 1。

伪随机数字映射

函数$\mathbf{R}_p(s,t)$是将（位长度l的）八位字符串s和（位长度k的）八位字符串t 映射到GF(p)的元素$\text{int}(x_1||x_2||...||x_n) \bmod p$的函数。函数$\mathbf{R}_p(s,t)$详细说明见下文图2。

按照[ISO/IEC 10116]，初始化向量=0，这一构造是基于CBC模式中的相应的分组密码**E**()构建的，其中，k是 **E**()的密钥长度（位）。必要时，结果k_i必须缩短为密钥长度k。应选择最小的值n ，使得$n*l \geq \log_2 p + 64$。

注：只有针对AES-192时才有必要缩短：使用k_i的1至24八位字节；其他的八位字节不使用。在DES的情况下，k被认为等于128位，$R(s,t)$的输出应为128位。

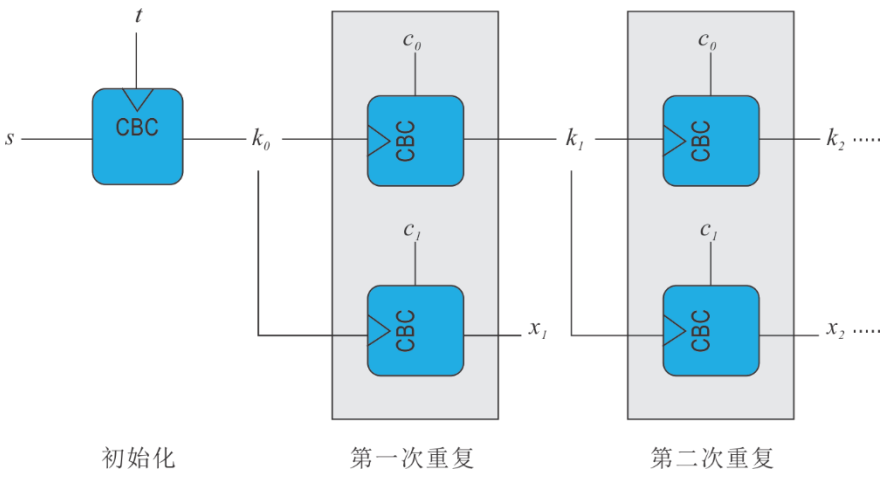

图2　伪随机数字映射

常数c_0和c_1定义如下：

- 关于3DES和AES-128 (l=128)：
  - c0=0xa668892a7c41e3ca739f40b057d85904
  - c1=0xa4e136ac725f738b01c1f60217c188ad
- 关于AES-192和AES-256 (l=256)：
  - c0=
    0xd463d65234124ef7897054986dca0a174e28df758cbaa03f240616414d5a1676
  - c1=
    0x54bd7255f0aaf831bec3423fcf39d69b6cbf066677d0faae5aadd99df8e53517

4.4.3.3.3　芯片认证映射

PACE-CAM的映射阶段和PACE-GM的映射阶段相同（参见第4.4.3.3.1节）。

4.4.3.4　认证令牌

使用认证码和密钥协商派生的密钥KS_{MAC}在公钥数据对象（参见9.4）和接收到的临时公钥（即排除域参数，参见第9.4.5节）的基础上计算认证令牌，其中公钥数据对象包含MSE:Set AT（参见第4.4.4.1节）中所示的客体标识符。

注：消息认证码插入进行填充，即不进行特定应用的填充。

3DES

按照[ISO/IEC 9797-1]中带有分组密码DES和IV=0的MAC算法3/填充方法2，以零售模式使用3DES［FIPS 46-3］。

AES

在8字节长度的消息认证码中使用AES［FIPS 197］的CMAC模式[SP 800-38B]。

4.4.3.5 加密芯片认证数据

eMRTD芯片必须根据第6.2节所述提供静态密钥对SK_{IC}, PK_{IC}。对于利用芯片认证映射的PACE来说，加密芯片认证数据是必要的。

4.4.3.5.1 由eMRTD芯片生成

芯片认证数据应按照$CA_{IC} = (SK_{IC})^{-1} * SK_{Map,IC} \bmod p$进行计算，其中，$SK_{IC}$ 是芯片的静态私钥，$SK_{Map,IC}$ 是芯片在PACE的映射阶段用来计算H的临时私钥（参见第4.4.3.3.1节），p 是所使用的密码群的阶。应使用密钥协商派生的密钥KS_{Enc}对芯片认证数据进行加密，从而生成加密芯片认证数据$A_{IC} = \mathbf{E}(KS_{Enc}, CA_{IC})$。

注：在eMRTD芯片的个性化过程中可提前计算$(SK_{IC})^{-1}$，并将其安全存储在芯片中，避免运行时的模逆运算。

4.4.3.5.2 终端核验

终端应对A_{IC} 解密以恢复CA_{IC} 并核证$PK_{Map,IC} = \mathbf{KA}(CA_{IC}, PK_{IC}, D_{IC})$, 其中$PK_{IC}$ 是eMRTD芯片的静态公钥。

注：必须与芯片认证映射结合进行被动认证。只有在对各个安全对象进行成功验证后，eMRTD芯片才可被视作真实的。

4.4.3.5.3 填充

应根据[ISO/IEC 9797-1]“填充方法2”对要加密的数据进行填充。

4.4.3.5.4 AES

应根据[ISO/IEC 10116]，$IV=E(KS_{Enc},-1)$，以CBC模式使用AES［19］，其中-1是长度128的字符串，所有位设为1。

4.4.4　应用协议数据单元

下列命令序列应用于实施PACE：

1. MSE:Set AT
2. GENERAL AUTHENTICATE

4.4.4.1　MSE:Set AT

命令MSE:Set AT用于选择并初始化PACE协议。MSE:Set AT对PACE的使用由PACE对象标识符（参见第4.4.3节和第9.2.3节）指示，该标识符包含为带有标记0x80的加密机制参考，参见下表。

命令			
CLA		上下文特定	
INS	0x22	管理安全环境	
P1/P2	0xC1A4	设定用于相互认证的模版	
数据	0x80	*密码机制引用* 要选择的协议客体标识符（只是值，标识符0x06省略）。	必要的
	0x83	*公钥/密钥引用* 所使用的口令由这个数据对象的以下数值显示： 0x01: MRZ_information 0x02: CAN	必要的
	0x84	*私钥引用/计算会话密钥的引用* 如果域参数不确定，即一个以上的域参数集可用于PACE，该数据对象是必要的，以指明要使用的域参数标识符。	有条件的
	0x7F4C	*证书持有者授权模板* 如果终端请求将用于终端认证的认证机构参考作为 PACE 的一部分返回，则该数据对象（在 Doc 9303-12 中定义）必须存在（参见第 4.4.5 节）。 该数据对象中包含的对象标识符应设置为id-IS（参见Doc 9303-10）。 任意数据模板中的访问位均应由终端设置为 1。	有条件的
响应			
数据	–	缺失	
状态字节	0x9000	*正常处理* 已选择协议并初始化。	

	0x6A80	命令数据域中的参数不正确 算法不支持或者初始化失败
	0x6A88	未发现所指数据 所指数据（即口令或域参数）不可用
	其他	运行系统相关错误 协议初始化失败

注 1：一些运行系统只在密钥用于选定目的时，才接受选择一个不可用的密钥并返回一个差错。

注 2：对于MSE:Set命令，IC应该忽略带有未为此命令指定的标签的数据对象。终端不应该包含带有IC不知道的标签的数据对象。

4.4.4.2 GENERAL AUTHENTICATE

使用一连串GENERAL AUTHENTICATE命令来执行PACE协议。

命令			
CLA		上下文特定	
INS	0x86	GENERAL AUTHENTICATE	
P1/P2	0x0000	默认密钥和协议	
数据	0x7C	动态认证数据 协议特定数据对象	必要的
响应			
数据	0x7C	动态认证数据 第4.4.5节所描述的协议特定数据对象	必要的
状态字节	0x9000	正常处理 协议（步骤）成功	
	0x6300	认证失败 协议（步骤）失败	
	0x6A80	命令数据域的参数不正确 所提供的数据无效	
	其他	运行系统相关错误 协议（步骤）失败	

4.4.4.3　命令链

命令链必须用于GENERAL AUTHENTICATE命令将命令序列与协议执行连接起来。除非芯片明确指明，否则命令链不得用于其他目的。关于命令链的详细信息，见[ISO/IEC 7816-4]。

4.4.5　交换的数据

协议特定数据对象应在一连串GENERAL AUTHENTICATE命令中，同表4所示的带有上下文特定标识符的压缩于动态认证数据对象中的协议特定命令和响应数据相交换（见第4.4.4.2节）：

表4　PACE交换的数据

步骤	说明	协议命令数据		协议响应数据	
1.	加密随机数	-	缺失[1]	0x80	加密随机数
2.	映射随机数	0x81	映射数据	0x82	映射数据
3.	执行密钥协商	0x83	临时公钥	0x84	临时公钥
4.	相互认证	0x85	认证令牌	0x86	认证令牌
				0x87	认证机构参考（有条件的）
				0x88	认证机构参考（有条件的）
				0x8A	加密芯片认证数据（有条件的）

如果数据对象0x7F4C在PACE设置期间传输到IC（参见第4.4.4.1节）并且IC支持终端认证，则必须存在认证机构参考。在这种情况下，数据对象0x87应包含最新的认证机构参考。数据对象0x88可能包含先前的认证机构参考。

如果使用芯片认证映射，必须有加密芯片认证数据（参见第4.4.3.5节），但在其他情况下不得有该数据。

4.4.5.1　加密随机数

加密随机数（参见第4.4.3.3节）应被编码为八位字节串。

1. 这暗示动态认证数据对象空白。

4.4.5.2 映射数据

交换的数据专用于所使用的映射：

4.4.5.2.1 通用映射

临时公钥（参见第4.4.3.3节和第9.4.4节）应被编码为椭圆曲线点（椭圆曲线密钥交换协议）或者无符号整数（密钥交换协议）。

4.4.5.2.2 合成映射

随机数*t* 应被编码为八位字节串。

注：为进行合成映射，上下文特定数据对象0x82应为空。

4.4.5.2.3 芯片认证映射

映射数据的编码和通用映射相同（参见第4.4.5.2.1节）。

4.4.5.3 公钥

公钥应按照第9.4.5节描述进行编码。

4.4.5.4 认证令牌

认证令牌（参见第4.4.3.4节）应被编码为八位字节串。

4.4.5.5 认证机构参考

认证机构参考(CAR)数据对象应按照Doc 9303-12中的规定进行编码。

4.4.5.6 加密芯片认证数据

在加密前应使用[TR-03111]中规定的函数FE2OS()将芯片认证数据编码为八位字节串。注意FE2OS()要求以同样数量的八位字节作为该群的素数阶进行编码，即可能包括前导0x00's。加密芯片认证数据应被编码为八位字节串。

5. 数据认证

除了LDS数据组外，非接触式集成电路上还包含一个证件安全对象（SO_D）。该对象由签发国或签发机构数字签名，并包括LDS内容的散列表示（见Doc 9303号文件第10部分）。

配备有各国证件签名者公钥的查验系统，或者已从eMRTD上读取了证件签名者证书（C_{DS}）的查验系统，将能核实证件安全对象（SO_D）。这样，通过证件安全对象（SO_D）的内容，LDS的内容就会得到认证。

这种验证机制不需要eMRTD中非接触式集成电路的处理能力。因此，它被称作对非接触式集成电路内容的“被动认证”。

被动认证可证明SO_D和LDS的内容是真实且未被修改的。它不能防止对非接触式集成电路内容的完全拷贝或芯片替换。

因此，被动认证系统应该辅之以对eMRTD的附加物理查验。

5.1 被动认证

5.1.1 查验过程

查验过程执行下列步骤：

1. 查验系统应从非接触式集成电路上读取证件安全对象（SO_D）（证件安全对象必须包括证件签名者证书（C_{DS}），另见Doc 9303号文件第10部分）。

2. 查验系统应建立并验证根据Doc 9303号文件第12部分用于证件安全对象（SO_D）签名的从信任锚到证件签名者证书的核验路径。

3. 查验系统应使用经核验的证件签名者公钥核验证件安全对象（SO_D）的签名。

4. 查验系统可从非接触式集成电路上读取相关的数据组。

5. 查验系统应通过将数据组的内容散列混编并将结果与证件安全对象（SO_D）中相对应的散列值进行比较，确保数据组的内容是真实和未被修改的。

下列其他检查被视作最佳做法：

1. 查验系统或查验员应检查证件签名者证书中是否有DocumentTypeExtension。

    - 如果有，查验系统应检查确认DocumentTypeExtension、数据组1的证件类型和可视MRZ的证件类型的一致性（分别见Doc 9303号文件第12部分、Doc 9303号文件第10部分和Doc 9303号文件第3部分）。

- 如果没有，查验系统应检查确认证件签名证书的KeyUsage设定为digitalSignature，检查确认证件签名证书不包含ExtendedKeyUsage-Extension（见Doc 9303号文件第12部分）。

2. 查验系统或查验员应从以下几方面检查国家代码的一致性：

   - Subject-field，以及证件签名者证书的SubjectAltName；
   - Subject-field，以及信任锚（国家签名认证机构证书）的SubjectAltName；
   - 从非接触式集成电路读取的数据组1；
   - 可视MRZ。

   此外，查验系统或查验员还可将数据组1的内容与可视MRZ进行对比（分别见Doc 9303号文件第12部分、第10部分和第3部分）。

3. 查验系统应核实证件签名证书中的密钥使用期包括eMRTD的签发日期（见Doc 9303号文件第12部分）。

现在可使用生物特征信息，和提供eMRTD的人一起进行生物特征核验。

5.1.2 LDS2应用的附加查验过程

电子机读旅行证件签发后写入的数据不受证件安全对象的保护，证件安全对象由证件签发人签署。为了验证签发后写入的数据的真实性，查验系统必须对每个写入的数据对象执行以下步骤：

1. 查验系统应依照Doc 9303-12建立并验证从信任锚到用于签署数据对象的签署者证书的认证路径。查验系统可以使用事先已知的证书和从芯片检索的证书来建立路径（参见Doc 9303-10）。

2. 查验系统应使用经过验证的签名者公钥来验证数据对象的签名。

注：对于真实性与接收国或接收机构的查验过程无关的数据对象可以跳过此程序。

6. 非接触式集成电路认证

签发国或签发机构可选择保护其eMRTD的芯片不被替换。

可使用下述机制来核实芯片的真实性。

1. 第6.1节定义的主动认证。对主动认证的支持通过EF.DG15的存在来表示。如果可用，终端可读取并核验EF.DG15并执行主动认证。

2. 第6.2节定义的芯片认证。对芯片认证的支持通过EF.DG14/EF.CardSecurity中的相应安全信息SecurityInfos的存在来表示。如果可用，终端可读取并核证EF.DG14/EF.CardSecurity并执行芯片认证。

3. 第4.4节定义的利用芯片认证映射的PACE（PACE-CAM）。支持通过EF.CardAccess内相应的PACEInfo结构的存在来表示。如果芯片访问程序成功进行了利用芯片认证映射的PACE，终端可进行下列步骤认证芯片：

    - 读取和核验EF.CardSecurity
    - 使用来自EF.CardSecurity的公钥，以及作为具有芯片认证映射对应关系的PACE一部分所收到的映射数据和芯片认证数据，以认证芯片（第4.4.3.5.2节）。

6.1　主动认证

主动认证通过用仅集成电路知晓的私钥对IFD（查验系统）发出的询问进行签名，来认证非接触式集成电路。

为此，非接触式集成电路包含有其自身的主动认证密钥对（KPr_{AA}和KPu_{AA}）。数据组15（公钥（KPu_{AA}）信息）的散列表示被存储在证件安全对象（SO_D）中，因此可由签发者的数字签名进行认证。相应的私钥（KPr_{AA}）被存储在非接触式集成电路的安全存储器中。

通过对视读MRZ进行认证（利用证件安全对象（SO_D）中的散列的MRZ）和使用eMRTD主动认证密钥对（KPr_{AA}和KPu_{AA}）进行的询问-应答，查验系统核验所读取的证件安全对象（SO_D）是从存储在真实eMRTD中的真实非接触式集成电路读取的。

主动认证需要eMRTD非接触式集成电路具备处理能力。

6.1.1　协议规范

使用[ISO/IEC 7816-4] INTERNAL AUTHENTICATE 命令进行主动认证。

如果在建立安全通讯后执行主动认证，必须根据第9.8节所述，所有命令和响应作为安全通讯应用协议数据单元进行发送。

具体地讲，IFD（查验系统）和集成电路（eMRTD的非接触式集成电路）执行下列步骤：

1. IFD生成一个随机数RND.IFD，并使用INTERNAL AUTHENTICATE命令将随机数发送到集成电路。
2. 集成电路执行下列操作：
    a) 生成消息M；
    b) 计算h(M)；
    c) 计算签名σ，并发送响应至IFD。

3. IFD核验INTERNAL AUTHENTICATE命令的响应，并检查集成电路是否返回了正确值。

6.1.2 密码规范

6.1.2.1 随机数

输入必须为一个8字节的随机数（RND.IFD）。

注：随机数不得重复使用，例如，用于BAC/PACE的随机数不得再次用于主动认证。

6.1.2.2 公钥加密算法

根据[ISO/IEC 9796-2]中的数字签名方案1，在使用基于整数分解的机制时，集成电路应计算签名。

在下文中，k代表生成签名的密钥长度，L_h代表签名生成过程中使用的散列函数H输出的长度。如果签名生成过程中使用SHA-1，则必须使用尾部字段选项1（t设定为1），否则必须使用尾部字段选项2（t设定为2）。

尾部字段的以下值应用于选项 2：

散列函数	SHA-224	SHA-256	SHA-384	SHA-512
尾部字段	0x38CC	0x34CC	0x36CC	0x35CC

出于互操作性原因，仅支持 SHA-1、SHA-224、SHA-256、SHA-384 和 SHA-512 作为使用 RSA 进行主动认证的散列函数。

要签名的消息M应是M1和M2的串联，其中M1必须是eMRTD生成的长度为c – 4的随机数（RND.IC），其中c（签名容量）由$c = k - L_h - (8 \times t) – 4$得出，而M2是由查验系统生成的RND.IFD。

签名计算结果必须是一个不含不可恢复的消息部分M2的签名.。

eMRTD应该实施［ISO/IEC 9796-2］B.6段所述的签名生成方案，而不应该使用［ISO/IEC 9796-2］B.4段所述的签名生成方案。eMRTD将不实施其他签名生成方案。

查验系统将实施［ISO/IEC 9796-2］B.6段所述的签名生成方案，并且应该实施［ISO/IEC 9796-2］B.4段所述的签名生成方案。

6.1.2.3　椭圆曲线数字签名算法

椭圆曲线数字签名算法，应根据[TR-03111]使用明文签名格式。仅应使用带有未压缩点的素数曲线。应使用输出长度与在使用中的椭圆曲线数字签名算法长度相同或者较短的散列算法。仅支持SHA-224、SHA-256、SHA-384或SHA-512作为散列函数。不应使用RIPEMD-160和SHA-1。

要签名的消息M是查验系统提供的随机数RND.IFD。

6.1.3　应用协议数据单元

根据[ISO/IEC 7816-4]的规定，通过一次调用INTERNAL AUTHENTICATE 命令执行主动认证。

命令			
CLA		上下文特定	
INS	0x88	INTERNAL AUTHENTICATE	
P1/P2	0x0000	—	
数据		RND.IFD	必要的
响应			
数据		集成电路生成的签名σ	必要的
状态字节	0x9000	正常处理 已成功执行协议。	
	其他	操作系统相关错误 协议失败。	

6.1.4　主动认证密钥

主动认证密钥对（KPr_{AA}和KPu_{AA}）应以安全方式生成。

主动认证公钥（KPu_{AA}）和主动认证私钥（KPr_{AA}）都被存储在eMRTD的非接触式集成电路中。此后，任何密钥管理都不适用于这些密钥。

注：应注意在安全通讯的主动认证中，使用超过1848位（如果使用3DES标准的安全通讯）/1792位（如果使用AES标准的安全通讯）的密钥长度时，扩展长度的应用协议数据单元必须得到eMRTD芯片和查验系统的支持。

签发国或签发机构应选择适当的密钥长度，保护eMRTD在其整个寿命期间不受攻击。应考虑到适当的密码分类。

6.1.5 主动认证公钥信息

主动认证公钥存储在LDS数据组15中。数据结构的格式（证件持有人公钥信息）规定参见[RFC 5280]中第9.1节。所有的安全对象必须以特异编码规则（DER）格式产生，以保持其中签名的完整性。

```
ActiveAuthenticationPublicKeyInfo ::= SubjectPublicKeyInfo
```

6.1.6 查验过程

当把带有数据组15的eMRTD提供给查验系统时，可使用主动认证机制确保数据是从真实的非接触式集成电路上读取的，确保非接触式集成电路和实际证件匹配。

查验系统和非接触式集成电路执行下列步骤：

1. 从eMRTD读取整个MRZ（如果还没有作为BAC的一部分被阅读的话），并与DG1中的MRZ数值比对。因为DG1的真实性和完整性已经通过被动认证进行验证，相似性可确保MRZ是真实的且未被修改。
2. 被动认证还证明了DG15的真实性和完整性。这可以确保主动认证公钥（KPu_{AA}）是真实的且未被修改。
3. 为确保证件安全对象（SO_D）不是复制的，查验系统如上文所述，在同eMRTD非接触式集成电路的询问-应答协议中使用eMRTD主动认证密钥对（KPr_{AA}和KPu_{AA}）。

在成功地执行询问 — 应答协议后，就可以证明证件安全对象（SO_D）是属于实际证件的，非接触式集成电路是真实的，而且非接触式集成电路和实际证件是匹配的。

6.2 芯片认证

芯片认证协议是临时静态的Diffie-Hellman密钥协商协议，提供电子机读旅行证件芯片的安全通信和单向认证。

和主动认证的主要差别在于：

- 阻止询问语义，因为该协议产生的副本是不可转移的。
- 除了电子机读旅行证件芯片的认证，该协议还提供强会话密钥。

关于询问语义的细节描述见附录c。

静态芯片认证密钥对必须存储在电子机读旅行证件芯片上。

- 私钥应被安全存储在电子机读旅行证件芯片的存储器内。
- 公钥存储在ChipAuthenticationPublicKeyInfo （芯片认证公钥信息）结构的 SubjectPublicKeyInfo （主体公钥信息）中（见第9.2.6节）。

通过使用新的会话密钥进行安全通讯，该协议对电子机读旅行证件芯片本身和存储数据进行了隐式认证。

如果IC支持芯片认证，IC可以支持主文件中的芯片认证和/或可以支持电子机读旅行证件应用程序中的芯片认证。如果芯片认证与LDS2应用中的数据组访问结合使用，IC必须支持主文件中的芯片认证。

注：如果需要与欧盟扩展访问控制[TR-03110]兼容，则IC必须支持电子机读旅行证件应用中的芯片认证。

6.2.1　协议规范

下列步骤由终端和电子机读旅行证件芯片执行。

1. 电子机读旅行证件芯片向终端发送其静态Diffie-Hellman公钥PK_{IC}和域参数D_{IC}。
2. 终端生成临时Diffie-Hellman密钥对（$SK_{DH,IFD}$, $PK_{DH,IFD}$, D_{IC}）并将临时公钥$PK_{DH,IFD}$发送给电子机读旅行证件芯片。
3. 电子机读旅行证件芯片和终端两者都进行下列计算：
   a) 共享秘密K = **KA**$(SK_{IC}, PK_{DH,IFD}, D_{IC})$ = **KA**$(SK_{DH,IFD}, PK_{IC}, D_{IC})$
   b) 根据K派生出用于安全通讯的会话密钥KS_{MAC} = **KDF**$_{MAC}(K)$ and KS_{Enc} = **KDF**$_{Enc}(K)$ 。

图3给出了简化版本：

IC（芯片）		**IFD（查验系统）**
静态密钥对$(SK_{IC}, PK_{IC}, D_{IC})$		
	— PK_{IC}, D_{IC} →	
		选择随机临时密钥对 $(SK_{DH,IFD}, PK_{DH,IFD}, D_{IC})$
	← $PK_{DH,IFD}$ —	
K = **KA**$(SK_{IC}, PK_{DH,IFD}, D_{IC})$		K = **KA**$(SK_{DH,IFD}, PK_{IC}, D_{IC})$

图3　芯片认证

为核验PK_{IC}的真实性，终端应执行被动认证。

6.2.2　安全状态

如果芯片认证成功执行，则可以使用派生的会话密钥KS_{MAC}和KS_{Enc}重启安全通讯。否则，使用之前建立的会话密钥（PACE或BAC）继续进行安全通讯。

注：被动认证必须和芯片认证结合起来执行。只有在对相应的SO_D进行成功认证后，电子机读旅行证件芯片才可被认为是真实的。

6.2.3 密码规范

具体算法由签发国或签发机构选择。查验系统必须支持下列各段所述的所有组合。电子机读旅行证件芯片可支持一个以上算法组合。

6.2.3.1 基于DH的芯片认证

基于DH的芯片认证，必须使用第9.6节和表5的相应算法和格式。关于公钥，必须使用PKCS#3 [PKCS#3]而不是X9.42 [X9.42]。

表5 基于DH的芯片认证的客体标识符

客体标识符	对称密码	密钥长度	安全通讯
id-CA-DH-3DES-CBC-CBC	3DES	112	CBC / CBC
id-CA-DH-AES-CBC-CMAC-128	AES	128	CBC / CMAC
id-CA-DH-AES-CBC-CMAC-192	AES	192	CBC / CMAC
id-CA-DH-AES-CBC-CMAC-256	AES	256	CBC / CMAC

6.2.3.2 基于ECDH的芯片认证

基于ECDH的芯片认证，必须使用第9.6节和表6的相应算法和格式。

表6 基于ECDH的芯片认证的客体标识符

客体标识符	对称密码	密钥长度	安全通讯
id-CA-ECDH-3DES-CBC-CBC	3DES	112	CBC / CBC
id-CA-ECDH-AES-CBC-CMAC-128	AES	128	CBC / CMAC
id-CA-ECDH-AES-CBC-CMAC-192	AES	192	CBC / CMAC
id-CA-ECDH-AES-CBC-CMAC-256	AES	256	CBC / CMAC

6.2.4　应用协议数据单元

基于使用的对称算法，芯片认证可有两种实现方法。

- 采用3DES安全通讯执行芯片认证应使用下列命令：

  1. MSE:Set KAT

- 采用AES安全通讯执行芯片认证应使用下列命令序列，该命令序列可用于采用3DES安全通讯执行的芯片认证：

  1. MSE:Set AT
  2. GENERAL AUTHENTICATE

6.2.4.1　使用MSE:Set KAT实现

注：MSE:Set KAT只可用于id-CA-DH-3DES-CBC-CBC和id-CA-ECDH-3DES-CBC-CBC，即安全通讯限于3DES。

命令			
CLA		上下文特定	
INS	0x22	管理安全环境	
P1/P2	0x41A6	设定用于计算的密钥协商模板	
数据	0x91	临时公钥 临时公钥$PK_{DH,IFD}$（参见第9.4.4节）编码为明文公钥值。	必要的
	0x84	私钥引用 如果私钥不明确，即用于芯片认证的密钥对有多个，该数据对象是必要的（参见第6.2节和第9.2.6节）。	有条件的
响应			
数据	–	缺失	
状态字节	0x9000	正常处理 成功进行密钥协商操作。已派生新的会话密钥。	
	0x6A80	命令数据域的参数不正确 临时公钥验证失败。	
	其他	操作系统相关错误 之前建立的会话密钥仍然有效。	

6.2.4.2 使用MSE:Set AT和GENERAL AUTHENTICATE实现

1. MSE:Set AT: 命令MSE:Set AT用于选择协议并对协议初始化。用于芯片认证的MSE:Set AT由芯片认证对象标识符（参见第6.2.3节和第9.2.7节）指示，该标识符包含为带有标记0x80的加密机制参考，请参见下表。

命令			
CLA		上下文特定	
INS	0x22	管理安全环境	
P1/P2	0x41A4	芯片认证： 设定用于内部认证的认证模版。	
数据	0x80	密码机制引用 要选择的协议的客体标识符（只是值，标识符0x06省略）。	必要的
	0x84	私钥引用 如果私钥不明确，即用于芯片认证的私钥有多个，该数据对象是必要的，用以指出要使用的私钥的标识符。	有条件的
响应			
数据	–	缺失	
状态字节	0x9000	正常处理 已选择协议并使之初始化。	
	0x6A80	命令数据域参数不正确 算法不支持或者初始化失败。	
	0x6A88	未找到引用数据 引用数据（即私钥）不可用。	
	other	操作系统相关错误 协议初始化失败。	

注：只有在密钥被用于特定目的时，一些操作系统才接受选择一个不可用的密钥并返回错误。

2. GENERAL AUTHENTICATE：使用命令GENERAL AUTHENTICATE实施芯片认证。

命令				
CLA		上下文特定		
INS	0x86	GENERAL AUTHENTICATE		
P1/P2	0x0000	默认密钥和协议		
数据	0x7C	动态认证数据 协议特定数据对象		必要的
		0x80	临时公钥	
响应				
数据	0x7C	动态认证数据 协议特定数据对象		必要的
状态字节	0x9000	正常处理 协议（步骤）成功。		
	0x6300	认证失败 协议（步骤）失败。		
	0x6A80	数据域参数不正确 所提供的数据无效。		
	0x6A88	未找到引用数据 引用数据（即私钥）不可用。		
	其他	操作系统相关错误 协议（步骤）失败。		

注：在安全对象中提供芯片支持的芯片认证公钥（见第9.2.11节）。如果支持多个公钥，终端必须在MSE:Set AT内选择相应的芯片私钥。

6.2.4.3　临时公钥

临时公钥（参见第9.4.5节）应被编码为椭圆曲线点（ECDH）或无符号整数（DH）。

7. 其他访问控制机制

为确保全球互通性，存储在非接触式集成电路上的个人数据，至少应包括MRZ和持证人的数字存储人脸图像。当eMRTD被打开供查验时，这两项内容都可以直观地看到（阅读）。

除了数字存储的人脸图像是全球互用的主要生物特征外，国际民航组织已经认可使用数字存储的指纹和/或虹膜图像。为供本国或双边使用，各国可以选择存储模板和/或选择限制访问或加密该数据，具体由各国自定。

对访问这些比较敏感的个人数据应该施加更多的限制。这可以通过两种途径实现：扩展访问控制或数据加密。第7.1节将终端认证指定为用于扩展访问控制的可互操作机制。如果不需要互操作性，则可以使用其他机制。

7.1 终端认证

终端认证机制是有条件的。LDS2 应用程序需要实施。终端认证可用于保护电子机读旅行证件应用程序中的二级生物特征。

终端认证协议是一个两步质询-响应协议，提供终端的明确单边认证。该协议基于 [TR-03110] 中指定的扩展访问控制。如果集成电路支持此协议，则它必须支持芯片认证或具有芯片认证映射的 PACE。

该协议使集成电路能够验证终端是否有权访问敏感数据。由于终端随后可能会访问敏感数据，因此必须适当保护所有进一步的通信。因此，终端认证还对终端选择的暂用公钥进行认证，该公钥用于设置带有芯片认证的安全消息传递或带有芯片认证映射的 PACE。集成电路必须将终端的访问权限绑定到由终端经过验证的暂用公钥建立的安全消息传递。

集成电路可以支持主文件和/或电子机读旅行证件应用程序中的终端认证。如果终端认证用于保护电子机读旅行证件应用以外的其他应用中的数据组，则集成电路必须支持主文件中的终端认证。

注：如果需要与欧盟扩展访问控制[TR-03110]兼容，集成电路必须支持电子机读旅行证件应用程序中的终端认证。

7.1.2 协议规范

以下步骤由终端和集成电路执行：

1. 终端向集成电路发送证书链。该链以可通过存储在芯片上的 CVCA 公钥验证的证书开始，并以终端证书结束。

2. 集成电路验证证书，提取终端公钥 PK_{IFD}。

3. 集成电路随机选择一个询问 r_{IC} 发送给终端。

4. 终端以签名 s_{IFD} = **Sign**(SK_{IFD}, $ID_{IC}||r_{IC}||$**Comp**($PK_{DH,IFD}$))进行响应。

5. 集成电路检查 **Verify**(PK_{IFD} , s_{IFD}, ID_{IC} ||r_{IC}||**Comp**(*PKDH,IFD*)) = *true*。

注：密钥 $PK_{DH,IFD}$ 是在芯片认证或带芯片认证映射的 PACE 期间生成的。如果生成了多个密钥（例如，在带芯片认证映射的 PACE 之后执行芯片认证），则必须使用最新的密钥。

在本协议中，ID_{IC} 是集成电路的标识符：

- 如果使用 BAC，ID_{IC} 是包含在机读区中电子机读旅行证件的证件号，包括校验位。
- 如果使用 PACE，则使用集成电路的临时 PACE 公钥计算 ID_{IC}，即 ID_{IC} = **Comp**($PK_{DH,IC}$)。

注：在 MF 中执行终端认证之前，需要成功执行 PACE 协议。

简化版如下图所示：

集成电路（芯片）		**IFD（查验系统）**
随机选择 r_{IC}		
	— r_{IC} →	
		s_{IFD} = **Sign**(SK_{IFD},ID_{IC}||r_{IC}||**Comp**($PK_{DH,IFD}$))
	← s_{IFD} —-	
Verify(PK_{IFD} , s_{IFD}, ID_{IC}||r_{IC}||**Comp**($PK_{DH,IFD}$)) = *true*		

图 4.　终端认证

7.1.3　安全状态

如果终端认证成功执行，集成电路应根据被认证终端的有效授权授予对存储的敏感数据的访问权限。如果有效授权未授予对 LDS2 应用程序中的任何数据的访问权限，则选择此应用程序必须被集成电路拒绝。

然而，集成电路应限制终端对由经过认证的临时公钥建立的安全消息传递的访问权限，即由终端提供的临时公钥作为芯片认证或带有芯片认证映射的 PACE 的一部分。集成电路不得在同一个会话中接受一次以上的终端认证（参见第 9.8.1 节和第 9.8.3 节关于“会话”的定义）。

注 1：只要由经过验证的临时公钥建立的安全消息传递处于活动状态，访问权限就是有效的，因此选择或取消选择应用程序不会影响安全状态。

注 2：安全消息传递不受终端认证的影响。即使终端认证失败，电子机读旅行证件芯片也应保留安全消息（除非发生安全消息错误）。

7.1.4 密码规范

7.1.4.1 RSA 终端认证

对于使用 RSA 的终端认证，必须使用以下算法和格式。

7.1.4.1.1 签名算法

应使用表 7 中指定的 RSA [RFC-3447], [PKCS#1]。

表 7. 使用 RSA 进行终端认证的对象标识符

OID	签名	散列值	参数
id-TA-RSA-PSS-SHA-256	RSASSA-PSS	SHA-256	默认值
id-TA-RSA-PSS-SHA-512	RSASSA-PSS	SHA-512	默认值

与 RSA-PSS 一起使用的默认参数定义如下：

- 散列算法：根据表 7 选择散列算法。
- 掩码生成算法：MGF1 [RFC-3447], [PKCS#1]使用选定的散列算法。
- Salt Length：所选散列算法输出的八位字节长度。
- 尾部字段： 0xBC

7.1.4.1.2 公钥格式

应使用 Doc9303-12 号文件中描述的 TLV 格式[ISO/IEC 7816-8]。

- 对象标识符应取自表 7。
- 模数的位长应为 2 048 或 3 072。
- 指数的位长最多为 32。

7.1.4.1.3 公钥压缩

终端的压缩临时公钥 **Comp**($PK_{DH,IFD}$)被定义为 DH 公共值的 SHA-1 散列值，即固定长度 20 的八位字节串。

7.1.4.2 ECDSA 终端认证

对于使用 ECDSA 的终端认证，必须使用以下算法和格式。

7.1.4.2.1 签名算法

应使用表 8 中指定的带有普通签名格式[TR-03111]的 ECDSA。

表 8.　使用 ECDSA 进行终端认证的对象标识符

OID	**签名**	**散列值**
id-TA-ECDSA-SHA-224	ECDSA	SHA-224
id-TA-ECDSA-SHA-256	ECDSA	SHA-256
id-TA-ECDSA-SHA-384	ECDSA	SHA-384
id-TA-ECDSA-SHA-512	ECDSA	SHA-512

7.1.4.2.2　公钥格式

应使用 Doc9303-12 号文件中描述的 TLV 格式[ISO/IEC 7816-8]。

- 对象标识符应取自表 8。
- 曲线的位长应为 224、256、320、384 或 512。
- 域参数应符合[TR-03111]。

7.1.4.2.3　公钥压缩

终端的压缩临时公钥 **Comp**($PK_{DH,IFD}$)被定义为 ECDH 公共点的 x 坐标，即一个固定长度⌈log256p⌉的八位字节串。

7.1.4.3　证书验证

为了验证终端证书，必须为集成电路提供一个证书链，该证书链始于存储在集成电路上的信任点。这些信任点或多或少是集成电路的 CVCA 的最新公钥。

7.1.4.3.1　集成电路信任点的初始状态

在生产或（预）个人化阶段，初始信任点应安全地存储在集成电路的内存中。

（预）个人化代理应：

- 将集成电路的当前日期设置为（预）个人化的日期；和
- 使用最近生效日期作为信任点来个人化 CVCA 密钥。

（预）个人化代理可以另外个人化以前的 CVCA 密钥作为信任点。

7.1.4.3.2　链接证书

由于 CVCA 使用的密钥会随时间变化，因此必须生成 CVCA 链接证书。CVCA 链接证书必须用以前的 CVCA 密钥签名，即具有最近生效日期的 CVCA 密钥。集成电路需要根据收到的有效链接证书在内部更新其信任点。

集成电路必须能够存储最多两个信任点。

注：由于 CVCA 链接证书的调度（参见 Doc9303-12），集成电路上最多需要存储两个信任点。

7.1.4.3.3 当前日期

集成电路必须接受过期的 CVCA 链接证书，但不能接受过期的 DV 和终端证书。为了确定证书是否过期，集成电路应使用其当前日期。

当前日期：如果集成电路没有内部时钟，则集成电路的当前日期应近似如下所述。该日期由集成电路使用有效 CVCA 链路证书、DV 证书或准确终端证书中包含的最新证书生效日期自主近似认定。

准确的终端证书：如果颁发的证件验证器(DV)被集成电路信任以生成具有正确证书生效日期的终端证书，则终端证书是准确的。国内 DV 签发的 CVCA 链路证书、DV 证书和终端证书，集成电路应认为是准确的。其他证书不得被认为是准确的。

终端可以将 CVCA 链接证书、DV 证书和终端证书发送到集成电路以更新当前日期和存储在集成电路上的信任点，即使终端不打算或不能继续进行终端认证。

注：集成电路仅验证显然是最近的证书（即关于近似当前日期），除非集成电路装有内部时钟。

7.1.4.3.4 一般验证程序

证书验证过程包括三个步骤：

1. **证书验证**：签名必须是有效的，除非证书是 CVCA 链接证书，否则证书不得过期。如果验证失败，程序应中止。

   注：CVCA 链接证书过期的情况只能发生在集成电路的时间源超过上述近似当前日期的情况。

2. **内部状态更新**：为验证 DV 证书，必须更新当前日期，必须导入公钥和属性（包括相关证书扩展），必须启用新的信任点，必须禁用过期的信任点。

3. **清理**：芯片应为每个应用程序提供最多两个启用的信任点。如果在内部状态更新后应用程序的两个以上信任点仍然启用，则应禁用具有最近生效日期的信任点。

更新当前日期的操作和启用和禁用信任点的操作必须作为原子操作来实现。

启用信任点：新的信任点应添加到信任点列表。

禁用信任点：过期的信任点不得用于验证 DV 证书。如果集成电路的当前日期可能超过信任点的到期日期，例如，集成电路使用内部时钟，则过期信任点必须保持不可用于 CVCA 链接证书的验证。在成功导入后续链接证书后，可以删除禁用的信任点。

7.1.4.3.5　验证程序示例

以下验证过程作为示例提供，可用于验证证书链。对于每个收到的证书，集成电路执行以下步骤：

1. 集成电路验证证书上的签名。如果签名不正确，则验证失败。

2. 如果证书不是 CVCA 链接证书，则将证书到期日期与集成电路的当前日期进行比较。如果到期日期在当前日期之前，则验证失败。

3. 接受证书有效，则导入证书中包含的公钥和属性（包括相关的证书扩展）。

    - 对于 CVCA、DV 和准确终端证书：将证书生效日期与集成电路的当前日期进行比较。如果当前日期早于生效日期，则将当前日期更新为生效日期。

    - 对于 CVCA 链接证书：新的 CVCA 公钥被添加到安全存储在集成电路内存中的信任点列表。然后启用新的信任点。

    - 对于 DV 和终端证书：临时导入新的 DV 或终端公钥，分别用于后续的证书验证或终端认证。

4. 安全存储在集成电路内存中的过期信任点被禁用以验证 DV 证书，并可从信任点列表删除。

7.1.4.3.6　有效认证

每个证书应载有一个证书持有人认证模板（见 Doc9303-12），并可载有认证扩展（见 Doc9303-12，第 7.2.2.6 节）。

- 证书持有人认证模板标识终端类型（本规范仅考虑检验系统，但其他规范可能使用不同的终端类型）。

- 证书持有人授权模板和授权扩展确定颁发证书颁发机构分配的证书持有人的相关授权。

为了确定证书持有人的有效授权，集成电路必须计算终端证书、引用的 DV 证书和引用的 CVCA 证书中包含的相关认证的按位布尔“and”。

有效授权应由集成电路解释如下：

- 有效角色是 CVCA：

    — 此链接证书由国家 CVCA 颁发。

    — 集成电路必须更新其内部信任点，即公钥和有效授权。

    — 证书颁发者是可信的时间来源，集成电路必须使用证书生效日期更新其当前日期。

— 集成电路不得授予 CVCA 访问敏感数据的权限（即应忽略有效授权）。

- 有效角色是 DV：

  — 由国家 CVCA 为授权的 DV 颁发的证书。

  — 证书颁发者是可信的时间来源，集成电路必须使用证书生效日期更新其当前日期。

  — 集成电路不得授予 DV 访问敏感数据的权限（即应忽略有效授权）。

- 有效角色是终端：

  — 证书由国内或国外 DV 签发。

  — 如果证书是准确的终端证书（参见第 7.1.4.3.3 节），则签发者是可信的时间来源，集成电路必须使用证书生效日期更新其当前日期。

  — 集成电路必须根据有效授权授予已验证终端访问敏感数据的权限。

注：证书持有人授权模板和认证扩展可以包含未分配给访问权限的位（RFU 位）。在评估访问权限期间，集成电路必须忽略这些位。

7.1.4.3.7 公钥导入

由证书验证程序导入的公钥永久或临时存储在集成电路上。

如果集成电路已经知道证书持有人参考，集成电路应该拒绝导入公钥。

永久导入：包含在 CVCA 链接证书中的公钥应由集成电路永久导入，并且必须安全地存储在集成电路的内存。永久导入的公钥及其元数据应满足以下条件：

- 它可以在到期后被后续永久导入的公钥覆盖。
- 它必须被具有相同证书持有人参考的后续永久导入公钥覆盖，或必须拒绝导入。
- 不得被临时导入的公钥覆盖。

启用和禁用永久导入的公钥必须是原子操作。

临时导入：包含在 DV 和终端证书中的公钥应由集成电路临时导入。临时导入的公钥及其元数据应满足以下条件：

- 在集成电路断电后，它不得选择或可用。
- 在后续加密操作成功完成之前，它必须保持可用（即 PSO：验证证书或外部认证）。

- 它可能会被后续临时导入的公钥覆盖。

终端不得使用任何临时导入的公钥，而是最近导入的公钥。

导入的元数据：对于每个永久或临时导入的公钥，必须存储证书中包含的以下附加数据（参见 Doc9303-12）：

- 证书持有人参考
- 证书持有人授权（有效角色和有效授权）
- 证书生效日期
- 证书失效日期
- 证书扩展（如适用）

第 7.1.4.3.6 节描述了有效角色（CVCA、DV 或终端）的计算和证书持有人的有效授权。

注：存储数据的格式取决于操作系统，这超出了本规范的范围。

7.1.5　应用协议数据单元

以下命令序列应与安全消息一起使用以实现终端认证：

- MSE:Set DST
- PSO:Verify Certificate
- MSE:Set AT
- Get Challenge
- External Authenticate

对每个要验证的 CV 证书（CVCA 链接证书、DV 证书、终端证书）重复步骤 1 和 2。

7.1.5.1 MSE:Set DST

命令 MSE:Set DST 用于设置证书验证。

指令			
CLA		上下文特定	
INS	0x22	管理安全环境	
P1/P2	0x81B6	设置数字签名模板进行验证。	
数据	0x83	公钥引用 要设置的公钥的 ISO 8859-1 编码名称。	必要的

响应		
数据	–	缺失
状态字节	0x9000	正常操作 已为给定目的选择了密钥。
	0x6A88	未找到参考数据 选择失败，因为公钥不可用。
	other	操作系统相关错误 未选择密钥。

注：某些操作系统接受选择不可用的公钥并仅在公钥用于选定目的时返回错误。

7.1.5.2 PSO:Verify Certificate

指令 PSO:Verify Certificate 用于验证和导入证书。

指令			
CLA		上下文特定	
INS	0x2A	进行安全操作	
P1/P2	0x00BE	验证自述证书。	
数据	0x7F4E	证书正文 要验证的证书正文。	必要的
	0x5F37	签名 要验证的证书的签名。	必要的

响应		
数据	–	缺失
状态字节	0x9000	常规处理 证书已成功验证并已导入公钥。
	其他	操作系统相关错误 无法导入公钥（例如，证书未被接受）。

7.1.5.3　MSE:Set AT

使用 MSE:Set AT 进行终端认证由 P1/P2 设置为 0x81A4 表示，见下表。

指令			
CLA		上下文特定	
INS	0x22	管理安全环境	
P1/P2	0x81A4	终端认证：	
数据	0x83	公钥/私钥的参考 该数据对象用于通过其 ISO 8859-1 编码名称选择终端的公钥。	必要的

响应		
数据	–	缺失
状态字节	0x9000 0x6A80 0x6A88 其他	常规处理 协议已被选择和初始化。 指令数据字段中的参数不正确 算法不受支持或初始化失败。 未找到参考数据 引用的数据不可用。 操作系统相关错误 协议初始化失败。

注：某些操作系统接受选择不可用的公钥并仅在公钥用于选定目的时返回错误。

7.1.5.4　Get Challenge

指令			
CLA		上下文特定	
INS	0x84	Get Challenge	
P1/P2	0x0000		
数据	–	缺失	
Le	0x08		必要的

响应		
数据	r_{IC}	8 字节的随机性
状态字节	0x9000 其他	常规处理 操作系统相关错误

7.1.5.5 External Authenticate

指令			
CLA		上下文特定	
INS	0x82	外部认证	
P1/P2	0x0000	密钥和算法隐含已知。	
数据		终端生成的签名。	必要的

响应		
数据	–	缺失
状态字节	0x9000	常规处理 认证成功。将根据相应验证证书的有效授权授予对数据组的访问权限。
	0x6300	警告 签名认证失败。
	0x6982	安全状态不满足 认证失败，因为终端当前的认证级别不允许使用终端认证（例如，终端认证已经执行等）。
	其他	操作系统相关错误 认证失败。

7.2 其他生物特征的加密

为限制访问其他生物特征，还可以对它们进行加密。为能够解密加密的数据，查验系统必须配备解密密钥。加密/解密算法和使用的密钥由执行国自行确定，不在本文件的范围之内。

对其他生物特征如何进行保护，取决于国家内部规范，或由共享该信息的几个国家双边商定的规范。

8. 查验系统

为了支持所需要的功能和eMRTD实现的定义选项，查验系统必须满足某些先决条件。

8.1 基本访问控制

支持BAC的查验系统必须满足下述先决条件：

1. 查验系统应能从实际证件获取MRZ从而派生出证件基本访问密钥（K_{Enc}和K_{MAC}）。

2. 当把带有BAC的eMRTD提供给查验系统，包括对带有安全通讯的通信信道进行加密时，该系统的软件需支持第4.3节中描述的协议。

8.2　口令认证连接确立协议

支持PACE的查验系统必须满足下列先决条件：

1. 查验系统可从实际证件获取MRZ和/或CAN。

2. 当把带有BAC的eMRTD提供给查验系统，包括对带有安全通讯的通信信道进行加密时，该系统的软件需支持第4.4节中描述的协议。

8.3　被动认证

为了对存储在eMRTD非接触式集成电路的数据进行被动认证，查验系统需要了解签发国或签发机构的密钥信息：

1. 对于每个签发国或签发机构来说，国家签名CA证书或从证书上提取的相关信息应安全存储于查验系统。

2. 或者，对于每个签发国或签发机构来说，证件签名者证书（C_{DS}）或从证书上提取的相关信息应安全存储于查验系统。

在使用签发国或签发机构的国家签名CA公钥之前，接收国或机构必须信任该密钥。

在使用证件签名者证书（C_{DS}）验证证件安全对象SO_D之前，查验系统应使用国家签名CA公钥核验其数字签名。

此外，查验系统应可获取验证过的证书撤销信息。

8.4　主动认证

查验系统对主动认证的支持是可选的。

如果查验系统支持主动认证，要求查验系统具备读取可视MRZ的能力。

如果查验系统支持主动认证，查验系统软件应支持第6.1节中描述的主动认证协议。

8.5 芯片认证

查验系统对芯片认证的支持是可选的。

如果查验系统支持芯片认证，要求查验系统具备读取可视MRZ的能力。

如果查验系统支持芯片认证，查验系统软件应支持第6.2节所述的芯片认证协议。

8.6 终端认证

查验系统对终端认证的支持是可选的。

如果查验系统支持终端认证，则要求查验系统具有安全存储查验系统私钥的能力。查验系统必须定期访问其DV以更新终端证书。

如果查验系统支持终端认证，查验系统的软件应支持第7.1节所述的终端认证协议。

8.7 其他生物特征的解密

如何保护可选其他生物特征取决于国家内部规范，或共享这一信息的国家双边商定的规范。

9. 通用规范

9.1 抽象语法标记（ASN.1）结构

数据结构SubjectPublic KeyInfo和AlgorithmIdentifier定义如下：

```
SubjectPublicKeyInfo ::= SEQUENCE {
   algorithm      AlgorithmIdentifier,
   subjectPublicKey     BIT STRING
}
AlgorithmIdentifier ::= SEQUENCE {
  algorithm OBJECT IDENTIFIER,
  parameters ANY DEFINED BY algorithm OPTIONAL
}
```

parameters的具体信息可见[X9.42]和[TR-03111]。

9.2 关于支持的协议和支持的应用的信息

ASN.1数据结构SecurityInfos应由电子机读旅行证件芯片提供，以表明支持的安全协议。数据结构具体说明如下：

```
SecurityInfos  ::= SET OF SecurityInfo

  SecurityInfo ::= SEQUENCE {
  protocol  OBJECT IDENTIFIER,
  requiredData ANY DEFINED BY protocol,
  optionalData ANY DEFINED BY protocol OPTIONAL
}
```

SecurityInfo数据结构包含的元素有下列含义：

- 客体标识符protocol指定支持的协议。
- 开放型requiredData包含协议特定的强制性数据。
- 开放型optionalData包含协议特定的选择性数据。

PACE的安全信息

为表明对支持PACE，SecurityInfos可包含下列项：

- 必须至少有一个使用标准化域参数的PACEInfo。
- 对于支持的每一个具体域参数集，必须存在PACEDomainParameterInfo。

AA（主动认证）安全信息

如果电子机读旅行证件芯片采用ECDSA（椭圆曲线数字签名算法）签名算法进行主动认证，SecurityInfos必须包括SecurityInfos项：

- ActiveAuthenticationInfo

CA（芯片认证）安全信息

为表明支持芯片认证，SecurityInfos可包含下列项：

- 必须至少有一个使用具体域参数的ChipAuthenticationInfo和相应的ChipAuthenticationPublicKeyInfo。

终端认证的安全信息

指出支持终端认证 SecurityInfos 可以包含下列项：

- 至少应有一个 TerminalAuthenticationInfo 存在。

目前应用的安全信息

Doc 9303-10 号文件第 3.11.2 节建议使用透明的基本文件 EF.DIR 来指示支持的应用程序。如果存在任何 LDS2 应用程序，则该文件是必需的。由于 EF.DIR 没有签名，因此可以被篡改，例如，为了向 IFD 隐藏现有的应用程序，如果存在任何 LDS2 应用程序，则提供 EF.DIR 的安全副本作为 SecurityInfo。

其他协议的安全信息

SecurityInfos 可包含表明支持其他协议或提供其他信息的附加项。查验系统可放弃任何未知项。

9.2.1 PACEInfo

该数据结构提供关于实现PACE的详细信息。

- 客体标识符protocol应指定要使用的算法（即密钥协商，对称密码和MAC）。
- version指定协议版本。本规范仅支持版本2。
- 整数parameterId用于表示域参数标识符。如果电子机读旅行证件芯片使用标准化域参数（参见第9.5.1节），并为PACE提供多个具体域参数，或者protocol是*-CAM-* OID（客体标识符）的一个，则必须使用整数parameterId。如利用芯片认证映射进行PACE，parameterID（参数标识符）还表示所使用的芯片认证密钥的ID，即芯片必须提供keyID的ChipAuthenticationPublicKeyInfo，其中keyID等同于该结构中的parameterID。

```
PACEInfo ::= SEQUENCE {
   Protocol      OBJECT IDENTIFIER (
                 id-PACE-DH-GM-3DES-CBC-CBC |
                 id-PACE-DH-GM-AES-CBC-CMAC-128 |
                 id-PACE-DH-GM-AES-CBC-CMAC-192 |
                 id-PACE-DH-GM-AES-CBC-CMAC-256 |
                 id-PACE-ECDH-GM-3DES-CBC-CBC |
                 id-PACE-ECDH-GM-AES-CBC-CMAC-128 |
                 id-PACE-ECDH-GM-AES-CBC-CMAC-192 |
                 id-PACE-ECDH-GM-AES-CBC-CMAC-256 |
                 id-PACE-DH-IM-3DES-CBC-CBC |
                 id-PACE-DH-IM-AES-CBC-CMAC-128 |
                 id-PACE-DH-IM-AES-CBC-CMAC-192 |
                 id-PACE-DH-IM-AES-CBC-CMAC-256 |
                 id-PACE-ECDH-IM-3DES-CBC-CBC |
                 id-PACE-ECDH-IM-AES-CBC-CMAC-128 |
                 id-PACE-ECDH-IM-AES-CBC-CMAC-192 |
                 id-PACE-ECDH-IM-AES-CBC-CMAC-256
                 id-PACE-ECDH-CAM-AES-CBC-CMAC-128 |
                 id-PACE-ECDH-CAM-AES-CBC-CMAC-192 |
                 id-PACE-ECDH-CAM-AES-CBC-CMAC-256),
   version       INTEGER, -- MUST be 2
   parameterId   INTEGER OPTIONAL
}
```

9.2.2 PACEDomainParameterInfo

如果电子机读旅行证件芯片提供PACE需要的具体域参数，则该数据结构是必要的，否则必须予以省略。

- 客体标识符protocol应指定域参数的类型（即DH或ECDH）。
- 序列domainParameter应包含域参数。
- 整数parameterId可用于表示本地域参数标识符。如果电子机读旅行证件芯片为PACE提供多个具体域参数，则必须使用整数parameterId。

```
PACEDomainParameterInfo ::= SEQUENCE {
   protocol           OBJECT IDENTIFIER(
                      id-PACE-DH-GM |
                      id-PACE-ECDH-GM |
                      id-PACE-DH-IM |
                      id-PACE-ECDH-IM
                      id-PACE-ECDH-CAM),
   domainParameter    AlgorithmIdentifier,
   parameterId        INTEGER OPTIONAL
}
```

注：电子机读旅行证件芯片可支持多个具体域参数集（即芯片可支持不同算法和/或密钥长度）。在这种情况下，必须在相应的PACEDomainParameterInfo中指定标识符。

PACEDomainParameterInfo中包含的域参数是不受保护的，可能不安全。使用不安全的PACE域参数将会泄露使用过的密码。电子机读旅行证件芯片必须支持第9.5.1节规定的至少一个标准化域参数集。查验系统不得使用电子机读旅行证件芯片提供的具体域参数，除非查验系统明确知道这些域参数是安全的。

临时公钥必须作为明文公钥值进行交换。关于编码的更多信息可参见第9.4.5节。

9.2.3 PACE客体标识符

PACE使用的客体标识符包含在bsi-de的子树中：

```
bsi-de OBJECT IDENTIFIER ::= {
   itu-t(0) identified-organization(4) etsi(0)
   reserved(127) etsi-identified-organization(0) 7
}
```

须使用下列客体标识符：

```
id-PACE OBJECT IDENTIFIER ::= {
   bsi-de protocols(2) smartcard(2) 4
}

id-PACE-DH-GM                     OBJECT IDENTIFIER ::= {id-PACE 1}
id-PACE-DH-GM-3DES-CBC-CBC        OBJECT IDENTIFIER ::= {id-PACE-DH-GM 1}
id-PACE-DH-GM-AES-CBC-CMAC-128    OBJECT IDENTIFIER ::= {id-PACE-DH-GM 2}
id-PACE-DH-GM-AES-CBC-CMAC-192    OBJECT IDENTIFIER ::= {id-PACE-DH-GM 3}
id-PACE-DH-GM-AES-CBC-CMAC-256    OBJECT IDENTIFIER ::= {id-PACE-DH-GM 4}

id-PACE-ECDH-GM                   OBJECT IDENTIFIER ::= {id-PACE 2}
id-PACE-ECDH-GM-3DES-CBC-CBC      OBJECT IDENTIFIER ::= {id-PACE-ECDH-GM 1}
id-PACE-ECDH-GM-AES-CBC-CMAC-128  OBJECT IDENTIFIER ::= {id-PACE-ECDH-GM 2}
id-PACE-ECDH-GM-AES-CBC-CMAC-192  OBJECT IDENTIFIER ::= {id-PACE-ECDH-GM 3}
id-PACE-ECDH-GM-AES-CBC-CMAC-256  OBJECT IDENTIFIER ::= {id-PACE-ECDH-GM 4}

id-PACE-DH-IM                     OBJECT IDENTIFIER ::= {id-PACE 3}
id-PACE-DH-IM-3DES-CBC-CBC        OBJECT IDENTIFIER ::= {id-PACE-DH-IM 1}
id-PACE-DH-IM-AES-CBC-CMAC-128    OBJECT IDENTIFIER ::= {id-PACE-DH-IM 2}
id-PACE-DH-IM-AES-CBC-CMAC-192    OBJECT IDENTIFIER ::= {id-PACE-DH-IM 3}
id-PACE-DH-IM-AES-CBC-CMAC-256    OBJECT IDENTIFIER ::= {id-PACE-DH-IM 4}

id-PACE-ECDH-IM                   OBJECT IDENTIFIER ::= {id-PACE 4}
id-PACE-ECDH-IM-3DES-CBC-CBC      OBJECT IDENTIFIER ::= {id-PACE-ECDH-IM 1}
id-PACE-ECDH-IM-AES-CBC-CMAC-128  OBJECT IDENTIFIER ::= {id-PACE-ECDH-IM 2}
id-PACE-ECDH-IM-AES-CBC-CMAC-192  OBJECT IDENTIFIER ::= {id-PACE-ECDH-IM 3}
id-PACE-ECDH-IM-AES-CBC-CMAC-256  OBJECT IDENTIFIER ::= {id-PACE-ECDH-IM 4}

id-PACE-ECDH-CAM                  OBJECT IDENTIFIER ::= {id-PACE 6}
id-PACE-ECDH-CAM-AES-CBC-CMAC-128 OBJECT IDENTIFIER ::= {id-PACE-ECDH-CAM 2}
id-PACE-ECDH-CAM-AES-CBC-CMAC-192 OBJECT IDENTIFIER ::= {id-PACE-ECDH-CAM 3}
id-PACE-ECDH-CAM-AES-CBC-CMAC-256 OBJECT IDENTIFIER ::= {id-PACE-ECDH-CAM 4}
```

9.2.4 *ActiveAuthenticationInfo*

如果电子机读旅行证件芯片采用ECDSA签名算法实现主动认证，电子机读旅行证件芯片逻辑数据结构数据组14中的SecurityInfos必须包含下列SecurityInfo项：

```
ActiveAuthenticationInfo ::= SEQUENCE {
   Protocol           OBJECT IDENTIFIER(id-icao-mrtd-security-aaProtocolObject),
   Version            INTEGER, -- MUST be 1
   signatureAlgorithm OBJECT IDENTIFIER
}

id-icao-mrtd-security-aaProtocolObject OBJECT IDENTIFIER ::=
                                    { id-icao-mrtd-security 5 }
```

关于signatureAlgorithm，应使用[TR-03111]中定义的客体标识符。

注：关于客体标识符id-icao-mrtd-security在Doc 9303号文件第10部分进行了定义。

9.2.5 ChipAuthenticationInfo

该数据结构提供关于实施芯片认证的详细信息。

- 客体标识符protocol应指定要使用的算法（即密钥协商，对称密码和和MAC）。
- 整数version（版本）应指定协议的版本。目前，本规范仅支持版本1。
- 整数keyId（密钥标识符）可用来表明本地密钥标识符。如果电子机读旅行证件芯片提供多个芯片认证公钥，则必须使用keyId。

```
ChipAuthenticationInfo ::= SEQUENCE {
   protocol  OBJECT IDENTIFIER(
             id-CA-DH-3DES-CBC-CBC |
             id-CA-DH-AES-CBC-CMAC-128 |
             id-CA-DH-AES-CBC-CMAC-192 |
             id-CA-DH-AES-CBC-CMAC-256 |
             id-CA-ECDH-3DES-CBC-CBC |
             id-CA-ECDH-AES-CBC-CMAC-128 |
             id-CA-ECDH-AES-CBC-CMAC-192 |
             id-CA-ECDH-AES-CBC-CMAC-256),
   version   INTEGER, -- MUST be 1
   keyId      INTEGER OPTIONAL
}
```

9.2.6 ChipAuthenticationPublicKeyInfo

该数据结构提供一个芯片认证公钥，或是利用电子机读旅行证件芯片的芯片认证映射的进行PACE的公钥。

- 客体标识符protocol应指定公钥的类型（即DH或ECDH）。
- 序列chipAuthenticationPublicKey应包含编码后的公钥。
- 整数keyId可用于表明本地密码标识符。如果电子机读旅行证件芯片提供多个芯片认证公钥，或者该密钥用于利用芯片认证映射PACE，则必须使用keyId。

```
ChipAuthenticationPublicKeyInfo ::= SEQUENCE {
   protocol                        OBJECT IDENTIFIER(id-PK-DH | id-PK-ECDH),
   chipAuthenticationPublicKey     SubjectPublicKeyInfo,
   keyId                           INTEGER OPTIONAL
}
```

注：电子机读旅行证件芯片可支持多个芯片认证密钥对（即芯片可支持不同算法和/或密钥长度）。在这种情况下，必须在相应的ChipAuthenticationInfo和ChipAuthenticationPublicKeyInfo中指定本地密钥标识符。

9.2.7 芯片认证客体标识符

应使用下列客体标识符：

```
id-PK OBJECT IDENTIFIER ::= {
   bsi-de protocols(2) smartcard(2) 1
}

id-PK-DH                      OBJECT IDENTIFIER ::= {id-PK 1}
id-PK-ECDH                    OBJECT IDENTIFIER ::= {id-PK 2}

id-CA OBJECT IDENTIFIER ::= {
   bsi-de protocols(2) smartcard(2) 3
}

id-CA-DH                      OBJECT IDENTIFIER ::= {id-CA 1}
id-CA-DH-3DES-CBC-CBC         OBJECT IDENTIFIER ::= {id-CA-DH 1}
id-CA-DH-AES-CBC-CMAC-128     OBJECT IDENTIFIER ::= {id-CA-DH 2}
id-CA-DH-AES-CBC-CMAC-192     OBJECT IDENTIFIER ::= {id-CA-DH 3}
id-CA-DH-AES-CBC-CMAC-256     OBJECT IDENTIFIER ::= {id-CA-DH 4}

id-CA-ECDH                    OBJECT IDENTIFIER ::= {id-CA 2}
id-CA-ECDH-3DES-CBC-CBC       OBJECT IDENTIFIER ::= {id-CA-ECDH 1}
id-CA-ECDH-AES-CBC-CMAC-128   OBJECT IDENTIFIER ::= {id-CA-ECDH 2}
id-CA-ECDH-AES-CBC-CMAC-192   OBJECT IDENTIFIER ::= {id-CA-ECDH 3}
id-CA-ECDH-AES-CBC-CMAC-256   OBJECT IDENTIFIER ::= {id-CA-ECDH 4}
```

9.2.8 TerminalAuthenticationInfo

此数据结构提供有关实施终端认证的详细信息。

- 对象标识符 `protocol` 应确定通用终端认证协议，因为特定的协议可能随时间有改变。
- 整数 `version` 应确定协议版本。目前，此规范支持第 1 版。请注意，[TR-03110] 的后续版本定义了该协议的第 2 版，这超出了本规范的范围。

```
TerminalAuthenticationInfo ::= SEQUENCE {
   protocol    OBJECT IDENTIFIER(id-TA),
   version     INTEGER      -- MUST be 1
  }
```

9.2.9 终端认证对象标识符

应使用以下对象标识符：

```
id-TA OBJECT IDENTIFIER ::= {
  bsi-de protocols(2) smartcard(2) 2
}

id-TA-RSA                OBJECT IDENTIFIER ::= {id-TA 1}
id-TA-RSA-PSS-SHA-256    OBJECT IDENTIFIER ::= {id-TA-RSA 4}
id-TA-RSA-PSS-SHA-512    OBJECT IDENTIFIER ::= {id-TA-RSA 6}

id-TA-ECDSA              OBJECT IDENTIFIER ::= {id-TA 2}
id-TA-ECDSA-SHA-224      OBJECT IDENTIFIER ::= {id-TA-ECDSA 2}
id-TA-ECDSA-SHA-256      OBJECT IDENTIFIER ::= {id-TA-ECDSA 3}
id-TA-ECDSA-SHA-384      OBJECT IDENTIFIER ::= {id-TA-ECDSA 4}
id-TA-ECDSA-SHA-512      OBJECT IDENTIFIER ::= {id-TA-ECDSA 5}
```

9.2.10 EFDIRInfo

数据结构封装了包含在主文件中的透明基本文件 EF.DIR 内容的完整副本。

```
  EFDIRInfo ::= SEQUENCE {
      protocol                        OBJECT IDENTIFIER(id-EFDIR),
      eFDIR                           OCTET STRING
  }

id-EFDIR OBJECT IDENTIFIER ::={
  id-icao-mrtd-security 13
}
```

9.2.11 芯片存储

为表明对协议和支持参数的支持，电子机读旅行证件芯片应在透明基本文件中提供SecurityInfos（这些文件的通用结构见Doc 9303号文件第10部分）：

- 如果电子机读旅行证件芯片支持PACE，则主文件中包含的文件EF.CardAccess是必要的，并且应包含PACE所必需的相关SecurityInfos：
  - PACEInfo
  - PACEDomainParameterInfo
- 如果：
  - 电子机读旅行证件芯片支持利用芯片认证映射的PACE，或
  - 电子机读旅行证件芯片支持主文件中的终端认证，或
  - 电子机读旅行证件芯片支持主文件中的芯片认证，

  则主文件包含的文件EF.CardSecurity是必要的，

  并且应包含下列SecurityInfos：
  - 芯片认证所必需的ChipAuthenticationInfo
  - PACE-CAM/芯片认证所必需的ChipAuthenticationPublicKeyInfo
  - 终端认证所必需的TermninalAuthenticationInfo
  - 如果芯片上存在多个电子机读旅行证件应用程序EFDIRInfo
  - EF.CardAccess包含的SecurityInfos。
- 如果电子机读旅行证件芯片支持具有通用/集成映射的PACE，
- 则电子机读旅行证件应用包含的文件DG14是必要的，
- 电子机读旅行证件芯片支持电子机读旅行证件应用程序中的终端认证，或
- 电子机读旅行证件芯片支持电子机读旅行证件应用程序中的芯片认证，

并且应包含下列SecurityInfos：

- 芯片认证所必需的ChipAuthenticationInfo
- 芯片认证所必需的ChipAuthenticationPublicKeyInfo
- 终端认证所必需的TermninalAuthenticationInfo
- EF.CardAccess中包含的SecurityInfos。

- SecurityInfos全集（包括DOC 9303号文件中未载明的EF.CardAccess所包含的SecurityInfos）应额外存储在电子机读旅行证件应用的EF.DG14（见Doc 9303号文件第10部分）中。

文件可包含本规范范围之外的其他SecurityInfos。

注：虽然存储在EF.DG14中的SecurityInfos和EF.CardSecurity的真实性由被动认证确保，但文件EF.CardAccess是未受保护的。

9.3　应用协议数据单元（APDUs）

9.3.1　扩展长度

根据加密对象（例如，公钥、签名）的大小，必须使用带有扩展长度域的应用协议数据单元，以便将该数据发送给电子机读旅行证件芯片。关于扩展长度的具体信息见[ISO/IEC 7816-4]。

9.3.1.1　电子机读旅行证件芯片

对于电子机读旅行证件芯片而言，对扩展长度的支持是有条件的。如果签发国选择的加密算法和密钥长度需要使用扩展长度，电子机读旅行证件芯片应支持扩展长度。如果电子机读旅行证件芯片支持扩展长度，必须在[ISO/IEC 7816-4]中规定的ATR/ATS或者EF.ATR/INFO中表明。

9.3.1.2　终端

对于终端来说，对扩展长度的支持是必要的。在使用该选项之前，终端应检查电子机读旅行证件芯片ATR/ATS或者EF.ATR/INFO中是否表明支持扩展长度。终端不得使用除下列命令之外的应用协议数据单元的扩展长度，除非电子机读旅行证件芯片的精确输入和输出缓冲区大小在ATR/ATS或EF.ATR/INFO中有明确说明。

- MSE:Set KAT
- GENERAL AUTHENTICATE

9.3.2　命令链

命令链必须用于GENERAL AUTHENTICATE命令，以将命令序列和PACE协议实施连接起来。命令链不得用于其他目的，除非芯片有明确指示。关于命令链的详情见[ISO/IEC 7816-4]。

9.3.3　数据对象

命令或响应APDU的发送者必须按照APDU描述中确定的顺序传输数据域中的数据对象。

注：不要求以任何顺序接受数据对象，但增强了某些命令的互操作性，例如MSE:Set AT/GENERAL AUTHENTICATE。但是，在PSO:Verify Certifcate等命令的情况下需要审慎，因为加密原因，顺序是固定的。

9.4 公钥数据对象

公钥数据对象是一个结构化的BERTLV结构，包含一个客体标识符和几个嵌套在持卡人公钥模版0x7F49内的上下文特定数据对象。

- 客体标识符是应用特定的，不仅涉及公钥格式（即上下文特定数据对象），也涉及其用法。
- 上下文特定数据对象由客体标识符定义，包含公钥值和域参数。

本规范中使用的公钥数据对象格式表述如下。

9.4.1 数据对象编码

无符号整数可转换为采用二进制整数表示（高位优先）的八位字节串。应使用八位字节的最小数字，即不得使用高字节值为0x00的的八位字节。

为编码椭圆曲线点，应根据[TR-03111]使用未压缩编码。

9.4.2 RSA公钥

RSA 公钥中包含的数据对象如表 9 所示。数据对象的顺序是固定的。

表 9 RSA 公钥

数据对象	符号	标签	类型	CV 认证
对象标识符		0x06	对象标识符	m
复合模量	*n*	0x81	对象标识符	m
公共指数	*e*	0x82	对象标识符	m

9.4.3 Diffie Hellman公钥

DH公钥包含的数据对象见表10。数据对象的顺序是固定的。

表10.　DH公钥数据对象

数据对象	符号	标签	类型
客体标识符		0x06	客体标识符
素数模	p	0x81	无符号整数
子群的次序	q	0x82	无符号整数
发生器	g	0x83	无符号整数
公钥值	y	0x84	无符号整数

注：密钥各部分采用无符号整数进行编码表明，每部分编码的长度要尽可能最小，即去掉值为0x00的高位字节。特别是，DH公钥编码后的长度要小于素数的长度。

9.4.4　椭圆曲线公钥

表11给出了椭圆曲线公钥内包含的数据对象。数据对象顺序是固定的，有条件的域参数必须是除了余因子之外全部都有，或者全部都无，如下所示：

表11　ECDH公钥的数据对象

数据对象	标记	标签	类型
客体标识符		0x06	客体标识符
素数模	p	0x81	无符号整数
第一系数	a	0x82	无符号整数
第二系数	b	0x83	无符号整数
基点	G	0x84	椭圆曲线点
基点次序	r	0x85	无符号整数
公钥点	Y	0x86	椭圆曲线点
辅因子	f	0x87	无符号整数

9.4.5　临时公钥

对于临时公钥而言，格式和域参数均已知。因此，只用明文公钥值，即Diffie-Hellman公钥的公钥值y和椭圆曲线公钥的公钥点Y，可用于在上下文特定数据对象中传递临时公钥。

注：建议对临时公钥进行验证。对于DH，验证算法需要电子机读旅行证件芯片知道比PKCS#3通常提供的域参数更多的信息。

9.5 域参数

除了PACEInfo包含的域参数以外，所有域参数应在AlgorithmIdentifier中给出（参见第9.1节）。

在PACEInfo内，表12中描述的标准化域参数的ID可直接引用。PACEDomainParameterInfo提供的具体域参数不得使用为标准化域参数预留的ID。

9.5.1 标准化域参数

应该使用下表所述的标准化域参数ID。具体域参数不得使用为标准化域参数预留的ID。

应该使用下列客体标识符引用客体标识符AlgorithmIdentifier（参见第9.1节）中的标准化域参数：

```
standardizedDomainParameters OBJECT IDENTIFIER ::= {
  bsi-de algorithms(1) 2
}
```

在AlgorithmIdentifier内，该客体标识符可引用表9中的标准化域参数的ID为整数，其包含在客体标识符AlgorithmIdentifier内，以parameters存在。

表12 标准化域参数

标识符	名称	大小（位）	类型	参考
0	1024-bit MODP Group with 160-bit Prime Order Subgroup	1024/160	GFP	[RFC 5114]
1	2048-bit MODP Group with 224-bit Prime Order Subgroup	2048/224	GFP	[RFC 5114]
2	2048-bit MODP Group with 256-bit Prime Order Subgroup	2048/256	GFP	[RFC 5114]
3-7	RFU			
8	NIST P-192 (secp192r1)	192	ECP	[RFC 5114], [FIPS 186-4]
9	BrainpoolP192r1	192	ECP	[RFC 5639]
10	NIST P-224 (secp224r1) *	224	ECP	[RFC 5114], [FIPS 186-4]
11	BrainpoolP224r1	224	ECP	[RFC 5639]

标识符	名称	大小（位）	类型	参考
12	NIST P-256 (secp256r1)	256	ECP	[RFC 5114], [FIPS 186-4]
13	BrainpoolP256r1	256	ECP	[RFC 5639]
14	BrainpoolP320r1	320	ECP	[RFC 5639]
15	NIST P-384 (secp384r1)	384	ECP	[RFC 5114], [FIPS 186-4]
16	BrainpoolP384r1	384	ECP	[RFC 5639]
17	BrainpoolP512r1	512	ECP	[RFC 5639]
18	NIST P-521 (secp521r1)	521	ECP	[RFC 5114], [FIPS 186-4]
19-31	RFU			

* 该曲线不能与合成映射一起使用。

9.5.2　显式域参数

DH或ECDH的客体标识符dhpublicnumber或ecPublicKey应分别用于引用AlgorithmIdentifier（参见第9.1节）内的具体域参数：

```
dhpublicnumber OBJECT IDENTIFIER ::= {
  iso(1) member-body(2) us(840) ansi-x942(10046) number-type(2) 1
}

ecPublicKey OBJECT IDENTIFIER ::= {
  iso(1) member-body(2) us(840) ansi-x962(10045) keyType(2) 1
}
```

采用椭圆曲线时，域参数必须在ECParameters结构中明确描述，作为parameters包含在AlgorithmIdentifier内，即不得使用指定曲线和具体域参数。

9.6 密钥协商算法

该规范支持下表总结的Diffie-Hellman和 Elliptic Curve Diffie-Hellman密钥协商：

表7 密钥协商算法

算法/格式	DH	ECDH
密钥协商算法	[PKCS#3]	ECKA [TR-03111]
X.509公钥格式	[X9.42]	[TR-03111]
TLV公钥格式	TLV，参见第9.4.3节	TLV，参见第9.4.4节
临时公钥验证	[RFC 2631]	[TR-03111]

9.7 密钥派生机制

9.7.1 密钥派生函数

密钥派生函数**KDF**(K,c)定义如下：

输入：需要下列输入：

- 共享秘密值K（必要的）
- 32位高位优先的整数计数器c（必要的）

输出：八位组字节串密钥数据。

动作：执行下列动作：

- keydata = **H**(K || c)
- 输出keydata的八位组字节串

密钥派生函数**KDF**(K,c)需要一个合适的用**H**()表示的散列函数，即散列函数的位长应大于或等于派生密钥的位长。散列值应被解读为高位优先字节输出。

注：共享密码K被限定为八位组字节串。如果共享秘密通过ECKA[TR03111]生成，则应使用生成点的x坐标。

9.7.1.1　3DES

为了派生128位的（不含奇偶校验位是112位）3DES [FIPS 46-3]密钥，应使用散列函数SHA-1 [FIPS 180-2]，并必须进行下列额外步骤：

- 使用keydata的1至8个八位组构成keydataA，使用keydata的9至16个八位组组成keydataB；其他八位组不使用。
- 调整keydataA和keydataB的奇偶校验位以构成正确的的DES密钥（选择性的）。

9.7.1.2　AES

为派生128位AES [FIPS 197]密钥，应使用散列函数SHA-1 [FIPS 180-4]，并必须进行下列额外步骤：

- 使用keydata的1至16个八位组；其他八位组不使用。

为派生192位和256位AES [FIPS 197]密钥，应使用SHA-256 [FIPS 180-4]。为导出192位AES密钥，必须进行下列额外步骤：

- 使用keydta的1至24个八位组；其他八位组不使用。

9.7.2　证件基本访问密钥

根据密钥种子（K）计算双密钥3DES密钥在确立证件基本访问密钥K_{Enc} = **KDF**(K,1)和K_{MAC} = **KDF**(K,2)时使用。

9.7.3　PACE

使$\mathbf{KDF}_{\pi}(\pi)$ = **KDF**(f(π),3)成为密钥派生函数，从而根据口令π派生出加密密钥。表14给出了口令的编码，即K= f(π)：

表11　口令编码

口令	编码
MRZ	SHA-1（证件号 ‖ 出生日期 ‖ 截止日期）
CAN	[ISO/IEC 8859-1] 编码字符串

注：用作输入的证件号始终是完整的证件号。对于证件号超过九个字符的TD 1型证件，证件号需要从证件号字段和机读区的可选数据字段连接起来，不包括填充字符。另见Doc 9303-5号文件第4.2.2节的注j)。

9.7.4 安全通讯密钥

利用共享密钥k可分别通过$\mathbf{KDF}_{Enc}(K) = \mathbf{KDF}(K,1)$ 和$\mathbf{KDF}_{MAC}(K) = \mathbf{KDF}(K,2)$ 派生出加密密钥和认证密钥。

9.8 安全通讯

9.8.1 会话启动

安全通讯一旦确立，会话便可启动。在会话中，可更改安全通讯密钥（即由BAC、PACE或芯片认证确立的密钥）。

安全通讯基于3DES[FIPS 46-3]或者AES [FIPS 197] 的加密-认证模式实现，即填充、加密数据，随后将格式化后的加密数据作为输入进行认证计算。会话密钥应使用第9.7.1节所述的密钥派生函数派生。

注：填充通常由安全通讯层执行，因此底层消息认证码不需要进行任何内部填充。

9.8.2 发送序列计数器

发送序列计数器（SSC）应采用无符号整数。SSC的位长应等于用于安全通讯的分组密文的分组长度，即对于3DES为64位，对于AES为128位。

在每次生成命令或者响应APDU之前，都应增加SSC，即如果起始值是x，在第一个命令里SSC的值是x+1。第一个响应的SSC值为x+2。

如果重启安全通讯，使用SSC如下：

- 用于密钥协商的命令使用旧的会话密钥和旧的SSC进行保护。这特别适用于用于会话密钥协商的最后一个命令的响应。
- 为SSC设定新的初始值，关于3DES见第9.8.6.3节和关于AES见第9.8.7.3节。
- 使用新会话密钥和新SSC保护后续命令/响应。

9.8.3 会话终止

只有在安全通讯错误发生时或者收到明文APDU时，电子机读旅行证件芯片才必须中断安全通讯。

如果中断安全通讯，电子机读旅行证件芯片应删除所存储的会话密钥，并重置终端访问权限。

注：当终止会话时，电子机读旅行证件芯片可隐式选择主文件。

9.8.4　SM APDU（安全通讯应用协议数据单元）的消息结构

SM数据对象必须按下列顺序使用（见[ISO/IEC 7816-4]）：

- Command APDU：　[DO‘85’ or DO‘87’] [DO‘97’] DO‘8E’。
- Response APDU：　[DO‘85’ or DO‘87’] [DO‘99’] DO‘8E’。

当 INS 是偶数时，应使用 DO‘87’，当 INS 是奇数时，应使用 DO‘85’。

所有SM数据对象必须按照[ISO/IEC 7816-4]中指定的BER TLV格式进行编码。MAC计算必须包含命令报头，因此必须使用类字节CLA = 0x0C。

应用安全通讯后，Lc的实际值将被修改为Lc’。如果需要，可将适当的数据对象包含进APDU数据部分，以传递Lc的原值。

图5给出了在数据和*Le*可用的情况下，如何将未受保护的命令APDU转变为受保护的命令APDU的过程。如果无数据可用，则忽略建立DO ‘87’。如果*Le*不可用，忽略建立DO ‘97’。为避免含糊不清，建议不要使用Le数据对象的空值字段（另见[ISO/IEC 7816-4]的第10.4节）。

图6给出了在数据可用的情况下，如何将未受保护的响应APDU转变为受保护的响应APDU的过程。如果无数据可用，则忽略建立DO ‘87’。

9.8.5　SM错误

发生下述情况时，电子机读旅行证件应用程序的安全信道会终止。

- 非接触式IC被切断电源；或者
- 非接触式IC在解读一条命令时识别出安全通讯错误。在这种情况下，状态字节必须在无安全通讯的情况下被返回。

如果安全通讯终止，电子机读旅行证件芯片会删除存储的会话密钥，并重置终端的访问权限。

注：也可能有非接触式IC终止会话的其他情况发生。但不可能提供这类情况的完整清单。

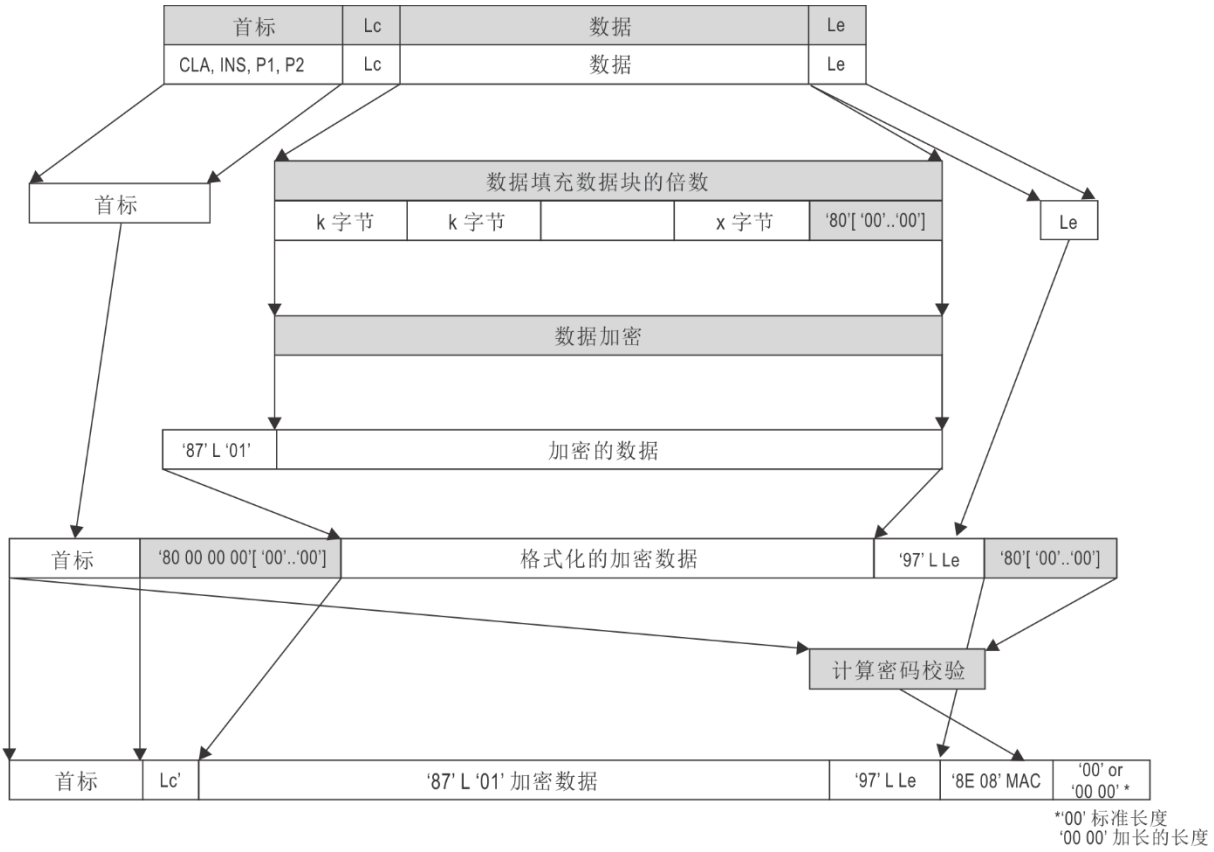

图5　计算偶数INS字节的SM命令APDU

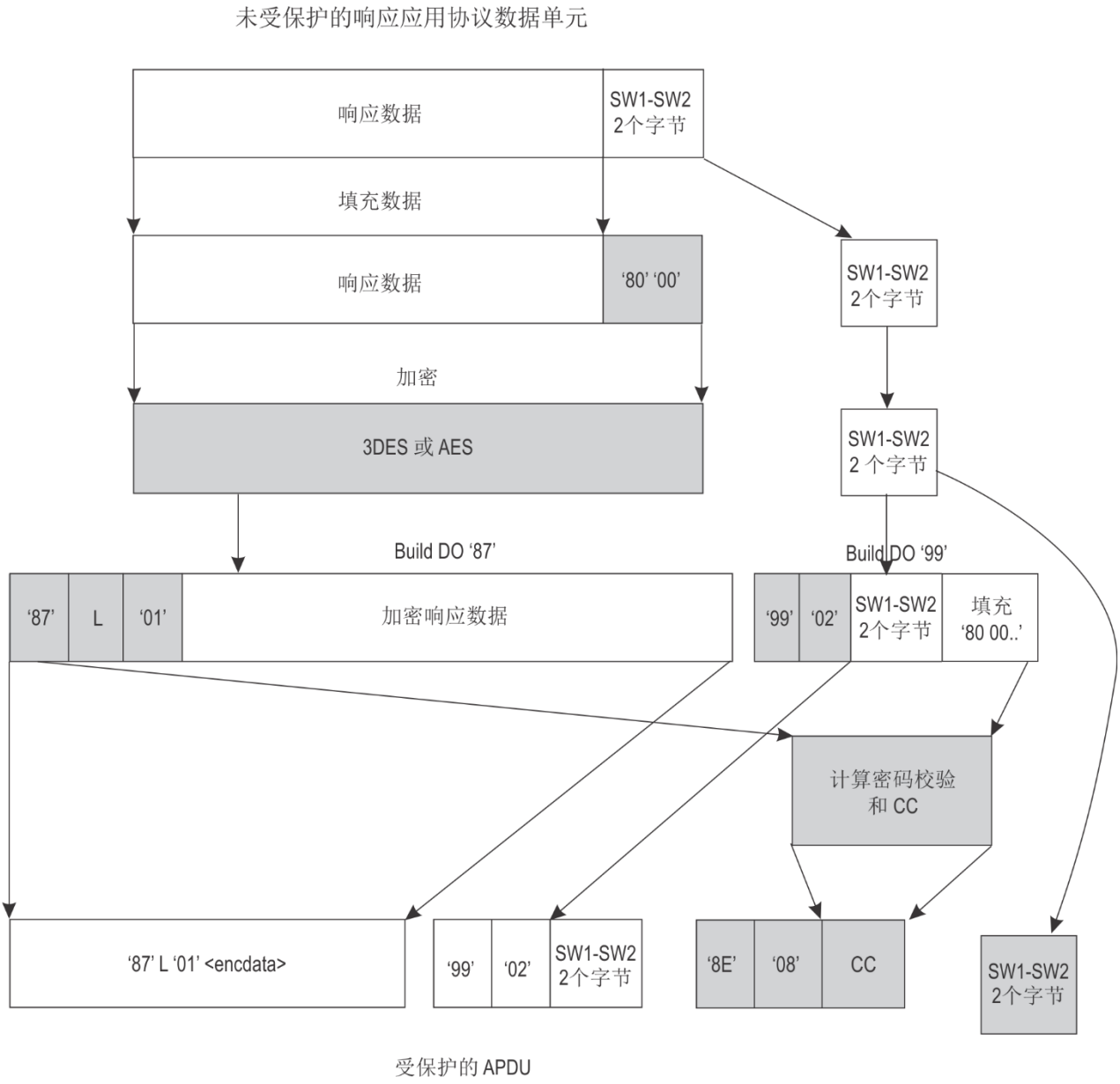

图6　计算偶数INS字节的SM响应

9.8.6 3DES 工作模式

9.8.6.1 加密

根据[ISO/IEC 11568-2]采用CBC模式的双钥3DES算法，初始化向量IV为0（即0x00 00 00 00 00 00 00 00）。根据[ISO/IEC 9797-1]填充方法2进行填充。

9.8.6.2 消息认证

密码校验和采用[ISO/IEC 9797-1]消息认证码算法3进行计算，其中算法为分组密码DES，初始化向量IV为0（8字节），采用[ISO/IEC 9797-1]填充方法2进行填充。消息认证码长度必须为8字节。

成功认证后，要进行MAC计算的报文应置于SSC（发送序列计数器）之前。

9.8.6.3 发送序列计数器

对于BAC（基本访问控制）后进行安全通讯，SSC应利用RND.IC和RND.IFD各自的四个最低有效字节连接后的数据进行初始化：

SSC = RND.IC（4个最低有效字节）|| RND.IFD（4个最低有效字节）。

在所有其他情况下，发送序列计数器应初始化为零（即0x00 00 00 00 00 00 00 00）。

9.8.7 AES工作模式

9.8.7.1 加密

消息加密时，应根据[ISO/IEC 10116]用密钥KS_{Enc}进行CBC模式的AES [FIPS 197]加密，其中IV = **E**(KS_{Enc}, SSC)。

9.8.7.2 消息认证

消息认证时，应用KS_{MAC}进行CMAC模式[SP 800-38B]的AES运算，MAC长度为8字节。要进行MAC计算的报文应置于SSC（发送序列计数器）之前。

9.8.7.3 发送序列计数器

发送序列计数器应初始化为零（即0x00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00）。

10. 参考文献（规范性）

[X9.42] ANSI: X9.42，金融服务业公钥密码：使用离散对数密码的对称密钥协议，1999年

[ISO/IEC 7816-4] ISO/IEC 7816-4:2013识别卡 — 集成电路卡 — 第4部分：交换的组织、安全和命令

[ISO/IEC 7816-8] ISO/IEC 7816-8:2019 识别卡 — 集成电路卡 — 第 8 部分：安全操作指令和机制

[ISO/IEC 8859-1] ISO/IEC 8859-1:1998信息技术 — 8位单字节编码图形字符集 — 第1部分：拉丁字母表1号

[ISO/IEC 9796-2] ISO/IEC 9796-2:2010信息技术 — 安全技术 — 消息恢复数字签名方法— 第2部分：基于整数分解的机制

[ISO/IEC 9797-1] ISO/IEC 9797-1:2011信息技术 — 安全技术 — 消息认证码（MAC） — 第1部分：使用分组密码的机制

[ISO/IEC 10116] ISO/IEC 10116:2017信息技术 — 安全技术 — n位分组密码运行模式

[ISO/IEC 11568-2] ISO/IEC 11568-2:2012金融服务 — 密钥管理（零售） — 第2部分：对称密码，其密钥管理和生命周期

[ISO/IEC 11770-2] ISO/IEC 11770-2:2018信息技术 — 安全技术 — 密钥管理 — 第2部分：使用对称技术的机制

[FIPS 46-3] NIST FIPS PUB 46-3，数据加密标准（DES），1999年

[FIPS 180-4] NIST FIPS PUB 180-4，安全散列标准，2015年

[FIPS 186-4] NIST FIPS PUB 186-4，数字签名标准（DSS），2013年

[FIPS 197] NIST FIPS PUB 197，高级加密标准（AES）的规范，2001年

[SP 800-38B] NIST特别出版物800-38B，关于分组密码运行模式的建议：CMAC认证模式，2005年

[RFC 2631] Rescorla, Eric：RFC 2631 DH密钥协商方法，1999年

[RFC 3447] Jonsson, Jakob和Kaliski, Burt：RFC 3447, 公钥密码标准(PKCS) #1: RSA 密码规范第2.1版，2003年

[RFC 5114] Lepinski, Matt; Kent, Stephen：RFC 5114采用IETF标准的额外DH群组，2008年

[RFC 5280] D. Cooper, S. Santesson, S. Farrell, S. Boyen, R. Housley, W. Polk, RFC 5280因特网X.509公钥基础设施证书和证书撤销列表（CRL）概况，2008年

[RFC 5639] Lochter, Manfred; Merkle, Johannes: RFC 5639椭圆曲线密码（ECC）Brainpool标准曲线和曲线生成，2010年

[TR-03110] BSI：技术指导原则TR-03110：机读旅行证件的高级安全机制

[TR-03111] BSI：技术指导原则TR-03111：椭圆曲线密码，2.0版，2012年

[PKCS#1] RSA实验室，PKCS#1 v2.2：RSA密码标准，2012年

[PKCS#3] RSA实验室，PKCS#3：DH密钥协商标准，1993年

[Keesing2009] J. Bender, D. Kügler：PACE解决方案简介：Keesing证件和身份期刊，第30期，Keesing，2009年

[BFK2009] J. Bender, M. Fischlin, D. Kügler：PACE密钥协商协议安全分析：ISC会议记录2009，LNCS，卷5735，Springer, 2009年.

[BCIMRT2010] Brier, Eric; Coron, Jean-Sébastien; Icart, Thomas; Madore, David; Randriam, Hugues; and Tibouch, Mehdi，有效不可微散列为普通椭圆曲线，密码学进展 — CRYPTO2010年，Springer-Verlag，2010年

— — — — — — — —

第11部分附录A

根据MRZ派生访问密钥的熵（资料性）

基本访问控制，因其简单的特点，被证明是一种非常成功的协议，目前几乎所有电子机读旅行证件都支持基本访问控制。

基本访问控制提供的安全性受到协议设计的限制。证件基本访问密钥（K_{Enc}和K_{MAC}）由随机性非常有限的打印数据生成。用于生成密钥的数据包括证件号、出生日期和截止日期。因此，所得到的密钥有一个相对低的熵，并且安全强度较低。实际的熵主要取决于证件号的类型。对一个十年有效期的旅行证件来说，密钥的最大强度大约为：

- 数字证件号56位（$365^2 * 10^{12}$种可能性）
- 字母数字证件号73位（$365^2*36^9*10^3$种可能性）。

特别是在第二种情况下，该估算要求证件号的选择应随机、均匀分布，通常情况下却并不如此。根据攻击者的掌握情况的多少，证件基本访问密钥的实际熵可能更低，例如，如果攻击者知道所有正在使用的证件号，或者能够将证件号与截止日期相关联。

提高基本访问控制的安全性并没有直接的方法，因其局限性是基于对称密码（“秘密密钥”）的协议设计所固有的。一个密码学上的强访问控制机制必须（额外）使用非对称（“公钥”）密码。

口令认证连接确立（PACE）的设计是用来解决该问题的。PACE采取非对称密码来建立会话密钥，其强度与所使用的口令的熵无关。如果PACE由256位椭圆曲线密码和AES-128（通常选择）实现，会话密钥的熵为128位。

必须区分两种类型的攻击：

- 非法浏览：这是一种在线攻击，即攻击者试图实时访问非接触式IC，例如通过猜测口令。如果用来保护非接触式IC的协议无密码缺陷，则攻击者的成功可能性取决于攻击者访问集成电路的时间、尝试猜测密码一次的持续时间，以及通行证的熵。
- 窃听：这是一种线下攻击，即攻击者试图在不访问非接触式IC的情况下解密所拦截的通信。如果用来建立会话密钥的协议无密码缺陷，成功可能性取决于会话密钥的强度以及攻击者拥有的计算能力。

为获取进一步信息，会话密钥熵概论基本访问控制和PACE的对比见[Keesing2009]，PACE密码分析见[BFK2009]。

———————————

第11部分附录B

ECDH合成映射点编码（资料性）

B.1 点编码方法的高级描述

算法采用曲线参数(a, b, p, f)作为输入，其中(a, b)是曲线系数，p是定义曲线E的素数域的特征，其中曲线方程为

$$E : y^2 \equiv x^3 + ax + b \ (mod\ p).$$

E的阶形如fq，其中q是某个素数，f 称为余因子。PACE v2要求选择q子群$E[q]$中的点作为生成元。点编码还将数字t 作为输入，使得

$$0 < t < p,$$

并且在固定时间内返回一个属于$E[q]$的点。如[BCIMRT2010]所述，点编码有两种类型，取决于所使用的坐标系：

- B.2描述使用仿射坐标(x, y) 输出椭圆曲线点地情况；
- B.3描述使用雅克比坐标(X, Y, Z) 输出椭圆曲线点地情况。

不管做何选择，在

$$x = XZ^{-2} \bmod p,\ y = YZ^{-3} \bmod p$$

的意义上，生成的点是相同的。因此，PACE v2在实现后续阶段（椭圆曲线DH密钥交换阶段）时，能够利用与椭圆曲线运行的加密应用程序接口（API）界面最匹配的方案。

如下文所述，仿射坐标的点编码大约需要两个模幂运算的模p ，而雅克比坐标的点编码仅需要一个。

注意对于两个可用的实施方法来说，点编码明确要求$p \equiv 3 \bmod 4$。

B.2 仿射坐标的实现

按照下文实现算法：

输入：曲线参数(a, b, p, f)和t，使得$0 < t < p$

输出：E的素数阶子群$E[q]$内的点 (x, y)

1. 计算$\alpha = -t^2 \bmod p$
2. 计算$X_2 = -ba^{-1}(1+(\alpha+\alpha^2)^{-1}) \bmod p$
3. 计算$X_3 = \alpha X_2 \bmod p$
4. 计算$h_2 = (X_2)^3 + a X_2 + b \bmod p$
5. 计算$h_3 = (X_3)^3 + a X_3 + b \bmod p$
6. 计算$U = t^3 h_2 \bmod p$
7. 计算$A = (h_2)^{p-1-(p+1)/4} \bmod p$
8. 如果$A^2 h_2 = 1 \bmod p$限定$(x, y) = (X_2, A h_2 \bmod p)$
9. 否则限定$(x, y) = (X_3, A U \bmod p)$
10. 输出$(x, y) = [f] (x, y)$。

实现注释

忽视模乘法和模加法，上述实现的执行时间由两个模幂运算决定：

- 可改写步骤2

  $$X_2 = -ba^{-1}(1+(\alpha+\alpha^2)^{-1}) = -b(1+\alpha+\alpha^2)\,(a(\alpha+\alpha^2))^{p-2} \bmod p$$

  基本上等于指数$p-2$的模幂运算；

- 步骤7是指数$p-1-(p+1)/4$的模幂运算。

注：步骤10需要由余因子f进行标量乘法。对很多曲线来说，余因子等于1，因此可避免该标量乘法。

B.3 雅克比坐标的实现

按照下文实现算法：

输入：曲线参数(a, b, p, f)和t，使得$0 < t < p$

输出：E的素数阶子群$E[q]$内的点(X, Y, Z)

1. 计算$\alpha = -t^2 \bmod p$
2. 计算$Z = a(\alpha+\alpha^2) \bmod p$
3. 计算$X_2 = -bZ\,(1+\alpha+\alpha^2) \bmod p$
4. 计算$X_3 = \alpha X_2 \bmod p$
5. 计算 $h_2 = (X_2)^3 + a X_2 Z^4 + b Z^6 \bmod p$
6. 计算$h_3 = (X_3)^3 + a X_3 Z^4 + b Z^6 \bmod p$
7. 计算$U = -\alpha\, t\, h_2 \bmod p$
8. 计算$A = (h_2)^{p-1-(p+1)/4} \bmod p$
9. 如果$A^2 h_2 = 1 \bmod p$限定$(X, Y, Z) = (X_2, A h_2 \bmod p, Z)$
10. 否则限定$(X, Y, Z) = (X_3, A U \bmod p, Z)$
11. 输出$(X, Y, Z) = [f](X, Y, Z)$。

实现注释

忽略模乘和模加，上述实现的执行时间由步骤7的单一模幂运算决定。因此，可以预计其时间大约是实现仿射坐标的两倍。

注：当余因子f等于1时可以完全避免步骤10中的标量乘法。

— — — — — — — — —

第11部分附录C

询问语义学（资料性）

考虑一个基于签名的电子机读旅行证件芯片（IC）和终端（IFD）之间的询问-应答协议，其中电子机读旅行证件芯片希望证实知晓其密钥SK_{IC}：

- 终端发送一个随机选择的询问c至机读旅行证件芯片。
- 机读旅行证件芯片以签名s=**Sign**(SK_{IC},c)作为应答。

虽然这是一个非常简单有效的协议，但电子机读旅行证件芯片实际上是在不知晓该消息语义学的情况下对消息c签名的。因为签名提供了可传递的真实性证据，原则上可说服任何第三方相信电子机读旅行证件芯片确实签名该消息。

虽然c应为一个随机位串，但终端也能以一种无法预知但可（公开）验证的方式生成该位串，例如，让SK_{IFD}成为终端密钥，且

$$c = \textbf{Sign}(SK_{IFD},ID_{IC}||Date||Time||Location)$$

是通过使用消息恢复的签名方法生成的询问。签名可确保终端确实生成了这一询问。由于终端签名的可传递性，信任终端并知晓相应公钥PK_{IFD}的任何第三方都可通过验证该签名确认是否正确创建了询问。此外，由于电子机读旅行证件芯片对询问签名的可传递性，第三方可推断该断言是真的：某一特定日期特定时间某一特定地点的电子机读旅行证件是真的。

从积极方面看，各国可在其内部使用询问语义学，例如，证明某一个人确实已移居入境。从消极方面看，这种证据在追踪人员方面可能被滥用。特别是由于主动认证并非限于授权终端，因此滥用是可能的。最糟糕的情境可能是支持主动认证但不支持基本访问控制的电子机读旅行证件芯片。在这种情况下，在一些突出位置放置一些安全硬件模块，可建立一个强大的追踪系统。由于有签名，所发生的运行记录不可能被伪造。基本访问控制一定程度上可减少这一问题，因为与持证人的互动是必要的。尽管这一问题仍然存在，但至少被限制在持证人旅行证件被读取的地点，例如，航空公司或者旅馆。

有人可能提出反对意见说，特别是在非接触情境下，询问可能被窃听，在不同日期、时间或地点可能被滥用，从而使得证据至少变得不可靠。尽管窃听询问在技术上是可能的，但这一论点仍然站不住脚。假设信任一终端正确生成询问，并可假设终端在开始主动认证之前已检查过电子机读旅行证件芯片的身份。因此，被窃听的询问将包括一个与签名询问的证明人身份不同的一个身份。

———————————

第11部分附录D

实例：基本访问控制（资料性）

D.1 从密钥种子（K_{SEED}）计算密钥

本节提供一个从种子值K_{seed}派生出3DES密钥的实例。本程序将作为基本访问控制实例的一个“子程序”。

输入：

K_{seed} = ‘239AB9CB282DAF66231DC5A4DF6BFBAE’

计算加密密钥（c = ‘00000001’）：

1. 连接K_{seed} 和c：
   D = ‘239AB9CB282DAF66231DC5A4DF6BFBAE00000001’

2. 计算D的SHA-1散列：
   H_{SHA-1}(D) = ‘AB94FCEDF2664EDFB9B291F85D7F77F27F2F4A9D’

3. 形成拟用作3DES第一和第二密钥的DES密钥K_a和$K_{b,}$（即3DES密钥是K_a和 K_b的连接）：
   $K_{a'}$ = ‘AB94FCEDF2664EDF’
   $K_{b'}$ = ‘B9B291F85D7F77F2’

4. 调整奇偶校验位：
   K_a = ‘AB94FDECF2674FDF’
   K_b = ‘B9B391F85D7F76F2’

计算消息认证码计算密钥（c = ‘00000002’）：

1. 连接K_{seed} 和c：
   D = ‘239AB9CB282DAF66231DC5A4DF6BFBAE00000002’

2. 计算D的SHA-1散列：
   H_{SHA-1}(D) = ‘7862D9ECE03C1BCD4D77089DCF131442814EA70A’

3. 形成密钥$K_{a'}$ 和$K_{b'}$：
   $K_{a'}$ = ‘7862D9ECE03C1BCD’
   $K_{b'}$ = ‘4D77089DCF131442’

4. 调整奇偶校验位：
   K_a = ‘7962D9ECE03D1ACD’
   K_b = ‘4C76089DCE131543’

D.2 证件基本访问密钥（K_{ENC}和K_{MAC}）的派生

本节提供如何从机读区派生基本访问密钥的实例。

TD2机读区，证件号超过9个字符

1. 读取MRZ：

   MRZ = I<UTOSTEVENSON<<PETER<JOHN<<<<<<<<<<
   D23145890<UTO3407127M95071227349<<<8

2. 根据MRZ创建“MRZ信息”

   证件号 = D23145890734 校验数位 = 9
   出生日期 = 340712 校验数位 = 7
   截止日期 = 950712 校验数位 = 2
   机读区信息 = D23145890734934071279507122

继续第3步。

TD2机读区，证件号超过9个字符

1. 读取MRZ：

   MRZ = I<UTOERIKSSON<<ANNA<MARIA<<<<<<<<<<<
   L898902C<3UTO6908061F9406236<<<<<<<8

2. 根据MRZ创建“机读区信息”：

   证件号 = L898902C< 校验数位 = 3
   出生日期 = 690806 校验数位 = 1
   截止日期 = 940623 校验数位 = 6
   机读区信息 = L898902C<3690806194062 36

继续第3步。

TD1机读区，证件号超过9个字符

1. 读取机读区

   MRZ = I<UTOD23145890<7349<<<<<<<<<<<
   3407127M9507122UTO<<<<<<<<<<<2
   STEVENSON<<PETER<JOHN<<<<<<<<<

2. 从机读区创建“机读区信息”

   证件号 = D23145890734 校验数位 = 9
   出生日期 = 340712 校验数位 = 7
   截止日期 = 950712 校验数位 = 2
   机读区信息 = D23145890734934071279507122

继续第3步。

TD1机读区，证件号超过9个字符

1. 读取MRZ

   MRZ = I<UTOL898902C<3<<<<<<<<<<<<<<<
   6908061F9406236UTO<<<<<<<<<<<1
   ERIKSSON<<ANNA<MARIA<<<<<<<<<<

2. 根据MRZ创建“机读区信息”

证件号　　= L898902C<　　校验数位 = 3
出生日期　= 690806　　校验数位 = 1
截止日期　= 940623　　校验数位 = 6
机读区信息 = L898902C<3690806194062 36

3. 计算“机读区信息”的SHA-1散列：

H_{SHA-1}（机读区信息） = ‘239AB9CB282DAF66231DC5A4DF6BFBAEDF477565’

4. 用最有效的16字节形成K_{seed}：

K_{seed} = ‘239AB9CB282DAF66231DC5A4DF6BFBAE’

5. 根据第9.7.1节/附录D.1计算基本访问密钥（K_{Enc} 和K_{MAC}）：

K_{Enc} = ‘AB94FDECF2674FDFB9B391F85D7F76F2’

K_{MAC} = ‘7962D9ECE03D1ACD4C76089DCE131543’

D.3　会话密钥的认证和建立

本节提供执行基本访问控制的实例。

查验系统：

1. 向eMRTD非接触式IC请求一个8字节随机数：

命令APDU：				
CLA	**INS**	**P1**	**P2**	**Le**
00	84	00	00	08

响应APDU：	
响应数据域	**SW1-SW2**
RND.IC	9000

RND.IC = ‘4608F91988702212’

2. 生成一个8字节随机数和一个16字节随机数：

RND.IFD = ‘781723860C06C226’
K_{IFD} = ‘0B795240CB7049B01C19B33E32804F0B’

3. 连接RND.IFD, RND.IC和K_{IFD}:

S = ‘781723860C06C2264608F91988702212
0B795240CB7049B01C19B33E32804F0B’

4. 用3DES密钥K_{Enc}为S加密：

E_{IFD} = ‘72C29C2371CC9BDB65B779B8E8D37B29
ECC154AA56A8799FAE2F498F76ED92F2’

5. 用3DES密钥K_{MAC}计算E_{IFD}的MAC（消息认证码）：
   M_{IFD} = ‘5F1448EEA8AD90A7’

6. 创建EXTERNAL AUTHENTICATE命令数据，并将命令APDU发送到eMRTD非接触式IC：
   cmd_data = ‘72C29C2371CC9BDB65B779B8E8D37B29ECC154AA
   56A8799FAE2F498F76ED92F25F1448EEA8AD90A7’

命令APDU：						
CLA	**INS**	**P1**	**P2**	**Lc**	**命令数据域**	**Le**
00	82	00	00	28	cmd_data	28

电子机读旅行证件非接触式IC：

1. 解码和验证接收的数据，并将RND.IC与得到GET CHALLENGE命令时的响应作比较。

2. 生成一个16字节的随机数：
   K_{IC} = ‘0B4F80323EB3191CB04970CB4052790B’

3. 计算K_{IFD}和K_{IC}的XOR：
   K_{seed} = ‘0036D272F5C350ACAC50C3F572D23600’

4. 根据第9.7.1节/附录D.1计算会话密钥（KS_{Enc} 和KS_{MAC}）：
   KS_{Enc} = ‘979EC13B1CBFE9DCD01AB0FED307EAE5’
   KS_{MAC} = ‘F1CB1F1FB5ADF208806B89DC579DC1F8’

5. 计算发送序列计数器：
   SSC = ‘887022120C06C226’

6. 连接RND.IC、RND.IFD和K_{IC}：
   R = ‘4608F91988702212781723860C06C226
   0B4F80323EB3191CB04970CB4052790B’

7. 用3DES密钥K_{Enc}为R加密：
   E_{IC} = ‘46B9342A41396CD7386BF5803104D7CE
   DC122B9132139BAF2EEDC94EE178534F’

8. 用3DES密钥K_{MAC}计算E_{IC}的MAC（消息认证码）：
   M_{IC} = ‘2F2D235D074D7449’

9. 创建EXTERNAL AUTHENTICATE的响应数据，并将响应APDU发送到查验系统：
   resp_data =‘46B9342A41396CD7386BF5803104D7CEDC122B91
   32139BAF2EEDC94EE178534F2F2D235D074D7449’

响应APDU：	
响应数据域	**SW1-SW2**
resp_data	9000

查验系统：

1. 解密并验证接收到的数据，并比较接收到的RND.IFD和生成的RND.IFD。.
2. 计算K_{IFD}和K_{IC}的XOR：
   K_{seed} = '0036D272F5C350ACAC50C3F572D23600'
3. 根据第9.7.1节/附录D.1计算会话密钥（KS_{Enc}和KS_{MAC}）：
   KS_{Enc} = '979EC13B1CBFE9DCD01AB0FED307EAE5'
   KS_{MAC} = 'F1CB1F1FB5ADF208806B89DC579DC1F8'
4. 计算发送序列计数器：
   SSC = '887022120C06C226'

D.4　安全通讯

在认证和建立会话密钥后，查验系统选择EF.COM（文件ID = '011E'），并通过使用安全通讯来读取数据。将使用计算出的KS_{Enc}、KS_{MAC}和SSC（查验系统较早的第3步和第4步）。

首先，将选择EF.COM，读取该文件的前四个字节，从而可确定结构长度，然后读取剩下的字节。

1. 选择EF.COM
   不受保护的命令APDU：

CLA	INS	P1	P2	Lc	命令数据域
00	A4	02	0C	02	01 1E

a) 掩码分类字节，并填充命令报头：
   CmdHeader = '0CA4020C80000000'

b) 填充数据：
   Data = '011E800000000000'

c) 用KS_{Enc}加密数据：
   EncryptedData = '6375432908C044F6'

d) 构建DO'87'：
   DO87 = '8709016375432908C044F6'

e) 连接CmdHeader（命令报头）和DO‘87’：
M = ‘0CA4020C8000000087090163754329 08C044F6’

f) 计算M的消息认证码：

i) SSC加1：
SSC = ‘887022120C06C227’

ii) 连接SSC和M，并增加填充：
N = ‘887022120C06C2270CA4020C80000000
8709016375432908C044F68000000000’

iii) 用KS_{MAC}计算N的消息认证码：
CC = ‘BF8B92D635FF24F8’

g) 建立DO‘8E’：
DO8E = ‘8E08BF8B92D635FF24F8’

h) 构建并发送受保护的APDU：
ProtectedAPDU = ‘0CA4020C158709016375432908C0
44F68E08BF8B92D635FF24F800’

i) 接收eMRTD非接触式IC的响应APDU：
RAPDU = ‘990290008E08FA855A5D4C50A8ED9000’

j) 通过计算DO‘99’的消息认证码，验证RAPDU CC：

i) SSC加1：
SSC = ‘887022120C06C228’

ii) 连接SSC和DO‘99’，并填充：
K = ‘887022120C06C2289902900080000000’

iii) 用KS_{MAC} 计算消息认证码：
CC’ = ‘FA855A5D4C50A8ED’

iv) 将CC’与 RAPDU的DO‘8E’数据作比较。
‘FA855A5D4C50A8ED’ == ‘FA855A5D4C50A8ED’ ? YES。

2. 读取前四个字节的二进制数：

不受保护的命令APDU：

CLA	INS	P1	P2	Le
00	B0	00	00	04

a) 掩码分类字节，并填充命令报头：
CmdHeader = ‘0CB0000080000000’

b) 建立DO‘97’：
DO97 = ‘970104’

c) 连接CmdHeader和DO‘97’:
M = ‘0CB0000080000000970104’

d) 计算M的MAC：
i) SSC加1 ：
SSC = ‘887022120C06C229’

ii) 连接SSC和M，并增加填充：
N = ‘887022120C06C2290CB00000
8000000097010480000000000’

iii) 用KS_{MAC}计算N的消息认证码：
CC = ‘ED6705417E96BA55’

e) 建立DO‘8E’：
DO8E = ‘8E08ED6705417E96BA55’

f) 构建并发送protected APDU（受保护的应用协议数据单元）：
ProtectedAPDU = ‘0CB000000D9701048E08ED6705417E96BA5500’

g) 接收电子机读旅行证件非接触式IC的响应APDU（应用协议数据单元）：
RAPDU = ‘8709019FF0EC34F992265199029000
8E08AD55CC17140B2DED9000’

h) 通过计算DO‘87’和DO‘99’连接的消息认证码，验证RAPDU CC：
i) SSC加1 ：
SSC = ‘887022120C06C22A’

ii) 连接SSC、DO‘87’和DO‘99’，并增加填充：
K = ‘887022120C06C22A8709019F
F0EC34F99226519902900080’

iii) 用KS_{MAC}计算消息认证码：
CC’ = ‘AD55CC17140B2DED’

iv) 将CC’与RAPDU的DO‘8E’数据作比较：
‘AD55CC17140B2DED’ == ‘AD55CC17140B2DED’ ? YES。

i) 用KS_{Enc}解码DO‘87’数据：
解码数据 = ‘60145F01’

j) 确定结构长度：
L = ‘14’ + 2 = 22字节

3. 从偏移位置4读取剩余18个字节的二进制数：

不受保护的命令APDU（应用协议数据单元）：

CLA	INS	P1	P2	Le
00	B0	00	04	12

a) 掩码分类字节，并填充命令报头：
CmdHeader = ‘0CB0000480000000’

b) 建立DO‘97’：
DO97 = ‘970112’

c) 连接CmdHeader和DO‘97’：
M = ‘0CB000048000000970112’

d) 计算M的消息认证码：

i) SSC加1：
SSC = ‘887022120C06C22B’

ii) 连接SSC和M，并增加填充：
N = ‘887022120C06C22B0CB00004
8000000097011280000000000’

iii) 用KS_{MAC}计算N的消息认证码：
CC = ‘2EA28A70F3C7B535’

e) 建立DO‘8E’：
DO8E = ‘8E082EA28A70F3C7B535’

f) 构建并发送受保护的APDU（应用协议数据单元）：
受保护的APDU = ‘0CB000040D9701128E082EA28A70F3C7B53500’

g) 接收eMRTD非接触式IC的响应APDU：
RAPDU = ‘871901FB9235F4E4037F2327DCC8964F1F9B8C30F42
C8E2FFF224A990290008E08C8B2787EAEA07D749000’

h) 通过计算DO‘87’和DO‘99’连接的消息认证码，验证RAPDU CC：

i) SSC加1：
SSC = ‘887022120C06C22C’

ii) 连接SSC、DO‘87’和DO‘99’，并增加填充：
K = ‘887022120C06C22C871901FB9235F4E4037F232
7DCC8964F1F9B8C30F42C8E2FFF224A99029000’

iii) 用KS_{MAC}计算消息认证码：
CC’ = ‘C8B2787EAEA07D74’

iv) 将CC’与RAPDU的DO‘8E’数据作比较：
‘C8B2787EAEA07D74’ == ‘C8B2787EAEA07D74’ ? YES

i) 用KS_{Enc}解密DO‘87’数据：
解密数据 = ‘04303130365F36063034303030305C026175’

结果：

EF.COM数据 = ‘60145F0104303130365F36063034303030305C026175’

— — — — — — — —

第11部分附录E

实例：被动认证（资料性）

步骤1　　从非接触式IC上读取证件安全对象（SO_D）（可选包含证件签名者证书（C_{DS}））。

步骤2　　从证书安全对象（SO_D）读取证书签名者（DS）。

步骤3　　查验系统使用证件签名者公钥验证SO_D。

步骤4　　查验系统使用国家签名CA公钥验证C_{DS}。

如果步骤3和4中的两次验证都是正确的，这就确保SO_D的内容是可信的，并可在查验过程中使用。

步骤5　　从LDS（逻辑数据结构）中读取相关的数据组。

步骤6　　计算相关数据组的散列。

步骤7　　将计算出的散列值与SO_D中的相应散列值作比较。

如果步骤7中的散列值是相同的，这就确保数据组的内容是真实的，没有被改变。

———————

第11部分附录F

实例：主动认证（资料性）

本实例使用下列设定：

1. 基于整数因数分解的机制： RSA
2. 模数长度(k)： 1024位(128字节)
3. 散列算法： SHA-1

查验系统：

步骤1 生成一个8字节随机数：
RND.IFD = ‘F173589974BF40C6’

步骤2 构建内部认证命令，并将命令APDU发送到电子机读旅行证件的非接触式IC：

命令APDU

CLA	INS	P1	P2	Lc	命令数据域	Le
00	88	00	00	08	RND.IFD	00

eMRTD非接触式IC：

步骤3 根据接收到的APDU确定M_2：
M_2 = ‘F173589974BF40C6’

步骤4 创建报尾：
T = ‘BC’（即，SHA-1）
T（八位字节内的T长度）= 1

步骤5 确定长度：
a. $c = k – L_h – 8t – 4 = 1024 – 160 – 8 – 4 = 852$位
b. $L_{M1} = c – 4 = 848$位

步骤6 生成长度为L_{M1}的随机数M_1：
M_1 = ‘9D2784A67F8E7C659973EA1AEA25D95B
6C8F91E5002F369F0FBDCE8A3CEC1991
B543F1696546C5524CF23A5303CD6C98
599F40B79F377B5F3A1406B3B4D8F967
84D23AA88DB7E1032A405E69325FA91A
6E86F5C71AEA978264C4A207446DAD4E
7292E2DCDA3024B47DA8’

步骤7　创建M：

M = M_1 | M_2 = ‘9D2784A67F8E7C659973EA1AEA25D95B
6C8F91E5002F369F0FBDCE8A3CEC1991
B543F1696546C5524CF23A5303CD6C98
599F40B79F377B5F3A1406B3B4D8F967
84D23AA88DB7E1032A405E69325FA91A
6E86F5C71AEA978264C4A207446DAD4E
7292E2DCDA3024B47DA8F173589974BF
40C6’

步骤8　计算M的SHA-1散列：

H = SHA-1(M) = ‘C063AA1E6D22FBD976AB0FE73D94D2D9
C6D88127’

步骤9[2]　构建消息代表：

F = ‘6A’ | M_1 | H | T =
‘6A9D2784A67F8E7C659973EA1AEA25D9
5B6C8F91E5002F369F0FBDCE8A3CEC19
91B543F1696546C5524CF23A5303CD6C
98599F40B79F377B5F3A1406B3B4D8F9
6784D23AA88DB7E1032A405E69325FA9
1A6E86F5C71AEA978264C4A207446DAD
4E7292E2DCDA3024B47DA8C063AA1E6D
22FBD976AB0FE73D94D2D9C6D88127BC’

步骤10　用主动认证私钥F加密，形成签名：

S = ‘756B683B036A6368F4A2EB29EA700F96
E26100AFC0809F60A91733BA29CAB362
8CB1A017190A85DADE83F0B977BB513F
C9C672E5C93EFEBBE250FE1B722C7CEE
F35D26FC8F19219C92D362758FA8CB0F
F68CEF320A8753913ED25F69F7CEE772
6923B2C43437800BBC9BC028C49806CF
2E47D16AE2B2CC1678F2A4456EF98FC9’

步骤11　为INTERNAL AUTHENTICATE构建响应数据，并将响应APDU发送到查验系统：

响应APDU：

响应数据域	**SW1-SW2**
S	9000

2. 由于已知部分（RND.IFD）没有被返回，但必须由 IFD 本身附加，所以部分恢复适用（‘6A’）。

查验系统：

步骤12　用公钥对签名进行解密：

F = ‘6A9D2784A67F8E7C659973EA1AEA25D9
5B6C8F91E5002F369F0FBDCE8A3CEC19
91B543F1696546C5524CF23A5303CD6C
98599F40B79F377B5F3A1406B3B4D8F9
6784D23AA88DB7E1032A405E69325FA9
1A6E86F5C71AEA978264C4A207446DAD
4E7292E2DCDA3024B47DA8C063AA1E6D
22FBD976AB0FE73D94D2D9C6D88127BC’

步骤13　通过报尾T*，确定散列算法：

T = ‘BC’ (i.e. SHA-1)

步骤14　提取散列：

D = ‘C063AA1E6D22FBD976AB0FE73D94D2D9
C6D88127’

步骤15　提取M_1：

M_1 = ‘9D2784A67F8E7C659973EA1AEA25D95B
6C8F91E5002F369F0FBDCE8A3CEC1991
B543F1696546C5524CF23A5303CD6C98
599F40B79F377B5F3A1406B3B4D8F967
84D23AA88DB7E1032A405E69325FA91A
6E86F5C71AEA978264C4A207446DAD4E
7292E2DCDA3024B47DA8’

步骤16　报头表明部分恢复，但签名具有模数长度，所以将M_1与已知的M_2连接（即：RND.IFD）：

M* = ‘9D2784A67F8E7C659973EA1AEA25D95B
6C8F91E5002F369F0FBDCE8A3CEC1991
B543F1696546C5524CF23A5303CD6C98
599F40B79F377B5F3A1406B3B4D8F967
84D23AA88DB7E1032A405E69325FA91A
6E86F5C71AEA978264C4A207446DAD4E
7292E2DCDA3024B47DA8F173589974BF
40C6’

步骤17　计算M*的SHA-1散列：

D* = ‘C063AA1E6D22FBD976AB0FE73D94D2D9
C6D88127’

步骤18　比较D和D*：

D等于D*，则验证成功。

— — — — — — — —

第11部分附录G

实例：口令认证连接确立（PACE）— 通用映射
（资料性）

本附录提供第4.4节规定的使用通用映射的PACE协议的两个实例。第一个实例基于ECDH密钥交换协议，而第二个实例基于DH密钥交换协议。表中包含的所有数字用十六进制表示。

在两个实例中，MRZ用作口令。这也导致对称密钥K_π相同。包括校验数位在内的MRZ相关数据域是：

- 证件号：T220001293；
- 出生日期：6408125；
- 截止日期：1010318。

因此，机读区编码K和派生加密密钥K_π 为

K	7E2D2A41 C74EA0B3 8CD36F86 3939BFA8 E9032AAD
K_π	89DED1B2 6624EC1E 634C1989 302849DD

G.1 基于椭圆曲线密钥交换协议（ECDH）的实例

本实例基于应用标准化BrainpoolP256r1域参数的ECDH（见[RFC 5639]）。

第一节介绍相应的PACEInfo。随后列举并审查所交换的APDU应用协议数据单元，包括生成的所有随机数和临时密钥。

椭圆曲线参数

通过使用标准化域参数，数据结构PACEInfo给出执行PACE需要的全部信息。特别是，不需要PACEDomainParameterInfo。

PACEInfo	3012060A 04007F00 07020204 02020201 0202010D

下表逐项列出了PACEInfo的具体结构。

标记	长度	值	ASN.1类型	注
30	12		序列	PACEInfo
06	0A	04 00 7F 00 07 02 02 04 02 02	客体标识符	利用ECDH密钥交换协议、通用映射和AES 128会话密钥的PACE
02	01	02	整数	第2版
02	01	0D	整数	Brainpool P256r1标准化域参数

为方便起见，下表给出了BrainpoolP256r1域参数的ASN.1编码。

标记	长度	值	ASN.1类型	注
30	81 EC		序列	域参数
06	07	2A 86 48 CE 3D 02 01	客体标识符	算法id-ecPublicKey
30	81 E0		序列	域参数
02	01	01	整数	版本
30	2C		序列	基础域
06	07	2A 86 48 CE 3D 01 01	客体标识符	素数域
02	21	00 A9 FB 57 DB A1 EE A9 BC 3E 66 0A 90 9D 83 8D 72 6E 3B F6 23 D5 26 20 28 20 13 48 1D 1F 6E 53 77	整数	素数p
30	44		序列	曲线方程
04	20	7D 5A 09 75 FC 2C 30 57 EE F6 75 30 41 7A FF E7 FB 80 55 C1 26 DC 5C 6C E9 4A 4B 44 F3 30 B5 D9	八位字节串	参数a
04	20	26 DC 5C 6C E9 4A 4B 44 F3 30 B5 D9 BB D7 7C BF 95 84 16 29 5C F7 E1 CE 6B CC DC 18 FF 8C 07 B6	八位字节串	参数b

04	41		八位字节串		群发生器G
		04	-		未压缩点
		8B D2 AE B9 CB 7E 57 CB 2C 4B 48 2F FC 81 B7 AF B9 DE 27 E1 E3 BD 23 C2 3A 44 53 BD 9A CE 32 62	-		X坐标
		54 7E F8 35 C3 DA C4 FD 97 F8 46 1A 14 61 1D C9 C2 77 45 13 2D ED 8E 54 5C 1D 54 C7 2F 04 69 97	-		Y坐标
02	21	00 A9 FB 57 DB A1 EE A9 BC 3E 66 0A 90 9D 83 8D 71 8C 39 7A A3 B5 61 A6 F7 90 1E 0E 82 97 48 56 A7	整数		群序n
02	01	01	整数		余因子f

基于ECDH实例的应用流程

为了初始化PACE，终端发送命令MSE:Set AT至芯片。

T>C :	00 22 C1 A4 0F 80 0A 04 00 7F 00 07 02 02 04 02 02 83 01 01
C>T :	90 00

此处T>C是从终端发送给芯片的APDU的缩写，C>T表示从芯片发送给终端的响应。命令编码如下表所示。

命令				
CLA	00		明文	
INS	22		管理安全环境	
P1/P2	C1 A4		设定相互认证的认证模板	
Lc	0F		数据域的长度	
数据	标记	长度	值	注
	80	0A	04 00 7F 00 07 02 02 04 02 02	密码机制：利用ECDH密钥交换协议、通用映射和AES 128会话密钥的PACE
	83	01	01	口令：机读区

响应		
状态字节	90 00	正常操作

加密随机数

然后，芯片随机生成随机数，并通过K_π将其加密。

解密后得到的随机数s	3F00C4D3 9D153F2B 2A214A07 8D899B22
加密后得到的随机数z	95A3A016 522EE98D 01E76CB6 B98B42C3

终端询问加密随机数。

T>C：	10 86 00 00 02 7C 00 00
C>T：	7C 12 80 10 95 A3 A0 16 52 2E E9 8D 01 E7 6C B6 B9 8B 42 C3 90 00

命令APDU和相应响应的编码见下表。

命令				
CLA	10		命令链	
INS	86		GENERAL AUTHENTICATE	
P1/P2	00 00		默认的密钥和协议	
Lc	02		数据长度	
数据	标记	长度	值	注
	7C	00	-	缺失
Le	00		响应数据域的预期最大字节长度是256	
响应				
数据	标记	长度	值	注
	7C	12		动态认证数据
	80	10	95 A3 A0 16 52 2E E9 8D 01 E7 6C B6 B9 8B 42 C3	加密随机数
状态字节	90 00		正常操作	

映射随机数

随机数通过通用映射被映射到一个临时群发生器。下表收集了所需的随机选择的临时密钥。

终端的私钥	7F4EF07B 9EA82FD7 8AD689B3 8D0BC78C F21F249D 953BC46F 4C6E1925 9C010F99
终端的公钥	7ACF3EFC 982EC455 65A4B155 129EFBC7 4650DCBF A6362D89 6FC70262 E0C2CC5E, 544552DC B6725218 799115B5 5C9BAA6D 9F6BC3A9 618E70C2 5AF71777 A9C4922D
芯片的私钥	498FF497 56F2DC15 87840041 839A8598 2BE7761D 14715FB0 91EFA7BC E9058560
芯片的公钥	824FBA91 C9CBE26B EF53A0EB E7342A3B F178CEA9 F45DE0B7 0AA60165 1FBA3F57, 30D8C879 AAA9C9F7 3991E61B 58F4D52E B87A0A0C 709A49DC 63719363 CCD13C54
共享秘密H	60332EF2 450B5D24 7EF6D386 8397D398 852ED6E8 CAF6FFEE F6BF85CA 57057FD5, 0840CA74 15BAF3E4 3BD414D3 5AA4608B 93A2CAF3 A4E3EA4E 82C9C13D 03EB7181
映射所得生成元$\hat{G}$	8CED63C9 1426D4F0 EB1435E7 CB1D74A4 6723A0AF 21C89634 F65A9AE8 7A9265E2, 8C879506 743F8611 AC33645C 5B985C80 B5F09A0B 83407C1B 6A4D857A E76FE522

由终端和芯片交换下列APDU，以映射随机数。

T>C：	10 86 00 00 45 7C 43 81 41 04 7A CF 3E FC 98 2E C4 55 65 A4 B1 55 12 9E FB C7 46 50 DC BF A6 36 2D 89 6F C7 02 62 E0 C2 CC 5E 54 45 52 DC B6 72 52 18 79 91 15 B5 5C 9B AA 6D 9F 6B C3 A9 61 8E 70 C2 5A F7 17 77 A9 C4 92 2D 00
C>T：	7C 43 82 41 04 82 4F BA 91 C9 CB E2 6B EF 53 A0 EB E7 34 2A 3B F1 78 CE A9 F4 5D E0 B7 0A A6 01 65 1F BA 3F 57 30 D8 C8 79 AA A9 C9 F7 39 91 E6 1B 58 F4 D5 2E B8 7A 0A 0C 70 9A 49 DC 63 71 93 63 CC D1 3C 54 90 00

APDU的结构可说明如下：

命令					
CLA	10		命令链		
INS	86		GENERAL AUTHENTICATE		
P1/P2	00 00		默认的密钥和协议		
Lc	45		数据长度		
数据集	标记	长度	值	注	
	7C	43	-	动态认证数据	
	81	41		映射数据	
			04		未压缩点
			7A CF 3E FC 98 2E ... C2 CC 5E		x坐标
			54 45 52 DC B6 72 ... C4 92 2D		y坐标
Le	00		响应数据域的预期最大字节长度是256		
响应					
数据	标记	长度	值	注	
	7C	43		动态认证数据	
	82	41		映射数据	
			04		未压缩点
			82 4F BA 91 C9 CB ... BA 3F 57		x坐标
			30 D8 C8 79 AA A9 ... D1 3C 54		y坐标
状态字节	90 00		正常运行		

执行密钥协商

在第三步，芯片和终端通过使用前一步的临时群发生器确定的新的域参数，执行一个匿名ECDH密钥交换协议的密钥协商。仅需要x坐标作为共享秘密，因为密钥导出函数(KDF)仅使用第一个坐标派生会话密钥。

终端的私钥	A73FB703 AC1436A1 8E0CFA5A BB3F7BEC 7A070E7A 6788486B EE230C4A 22762595
终端的公钥	2DB7A64C 0355044E C9DF1905 14C625CB A2CEA487 54887122 F3A5EF0D 5EDD301C, 3556F3B3 B186DF10 B857B58F 6A7EB80F 20BA5DC7 BE1D43D9 BF850149 FBB36462
芯片的私钥	107CF586 96EF6155 053340FD 633392BA 81909DF7 B9706F22 6F32086C 7AFF974A
芯片的公钥	9E880F84 2905B8B3 181F7AF7 CAA9F0EF B743847F 44A306D2 D28C1D9E C65DF6DB, 7764B222 77A2EDDC 3C265A9F 018F9CB8 52E111B7 68B32690 4B59A019 3776F094
共享秘密	28768D20 701247DA E81804C9 E780EDE5 82A9996D B4A31502 0B273319 7DB84925

密钥协商执行如下：

T>C :	10 86 00 00 45 7C 43 83 41 04 2D B7 A6 4C 03 55 04 4E C9 DF 19 05 14 C6 25 CB A2 CE A4 87 54 88 71 22 F3 A5 EF 0D 5E DD 30 1C 35 56 F3 B3 B1 86 DF 10 B8 57 B5 8F 6A 7E B8 0F 20 BA 5D C7 BE 1D 43 D9 BF 85 01 49 FB B3 64 62 00
C>T :	7C 43 84 41 04 9E 88 0F 84 29 05 B8 B3 18 1F 7A F7 CA A9 F0 EF B7 43 84 7F 44 A3 06 D2 D2 8C 1D 9E C6 5D F6 DB 77 64 B2 22 77 A2 ED DC 3C 26 5A 9F 01 8F 9C B8 52 E1 11 B7 68 B3 26 90 4B 59 A0 19 37 76 F0 94 90 00

通过下表审查密钥协商编码：

命令				
CLA	10		命令链	
INS	86		GENERAL AUTHENTICATE	
P1/P2	00 00		默认的密钥和协议	
Lc	45		数据长度	
数据	标记	长度	值	注
	7C	43	-	动态认证数据
	83	41		终端的临时公钥
			04	未压缩点

			2D B7 A6 4C 03 55 ... DD 30 1C		x坐标
			35 56 F3 B3 B1 86 ... B3 64 62		y坐标
Le	00		响应数据域的预期最大字节长度为256		
响应					
数据	标记	长度	值	注	
	7C	43		动态认证数据	
	84	41		芯片的临时公钥	
			04		未压缩点
			9E 88 0F 84 29 05 ... 5D F6 DB		x坐标
			77 64 B2 22 77 A2 ... 76 F0 94		y坐标
状态字节	90 00		正常操作		

通过密钥导出函数（KDF），从共享秘密派生AES 128会话密钥KS_{Enc}和KS_{MAC}。分别为：

KS_{Enc}	F5F0E35C 0D7161EE 6724EE51 3A0D9A7F
KS_{MAC}	FE251C78 58B356B2 4514B3BD 5F4297D1

相互认证

利用下列输入数据，通过KS_{MAC}派生出认证令牌：

T_{IFD}的输入数据	7F494F06 0A04007F 00070202 04020286 41049E88 0F842905 B8B3181F 7AF7CAA9 F0EFB743 847F44A3 06D2D28C 1D9EC65D F6DB7764 B22277A2 EDDC3C26 5A9F018F 9CB852E1 11B768B3 26904B59 A0193776 F094
T_{IC}的输入数据	7F494F06 0A04007F 00070202 04020286 41042DB7 A64C0355 044EC9DF 190514C6 25CBA2CE A4875488 7122F3A5 EF0D5EDD 301C3556 F3B3B186 DF10B857 B58F6A7E B80F20BA 5DC7BE1D 43D9BF85 0149FBB3 6462

输入数据的编码如下所示：

标记	长度	值	ASN.1类型	注
7F49	4F		公钥	T_{IFD}的输入数据
06	0A	04 00 7F 00 07 02 02 04 02 02	客体标识符	利用ECDH密钥交换协议、通用映射和AES 128会话密钥的PACE
86	41		椭圆曲线点	芯片的临时公共点
		04		未压缩点t
		9E 88 0F 84 29 ... 5D F6 DB		x坐标
		77 64 B2 22 77 ... 76 F0 94		y坐标

标记	长度	值	ASN.1类型	注
7F49	4F		公钥	T_{IC}的输入数据
06	0A	04 00 7F 00 07 02 02 04 02 02	客体标识符	利用ECDH密钥交换协议、通用映射和AES 128会话密钥的PACE
86	41		椭圆曲线点	终端的临时公共点
		04		未压缩点
		2D B7 A6 4C 03 ... DD 30 1C		x坐标
		35 56 F3 B3 B1 ... B3 64 62		y坐标

所计算的认证令牌是：

T_{IFD}	C2B0BD78 D94BA866
T_{IC}	3ABB9674 BCE93C08

最后，交换并验证这些令牌。

T>C：	00 86 00 00 0C 7C 0A 85 08 C2 B0 BD 78 D9 4B A8 66 00
C>T：	7C 0A 86 08 3A BB 96 74 BC E9 3C 08 90 00

G.2 基于DH的实例

第二个实例基于使用带有[RFC 5114]规定的160位素数阶子群的1024位MODP群的DH。群参数是：

素数*p*	B10B8F96 A080E01D DE92DE5E AE5D54EC 52C99FBC FB06A3C6 9A6A9DCA 52D23B61 6073E286 75A23D18 9838EF1E 2EE652C0 13ECB4AE A9061123 24975C3C D49B83BF ACCBDD7D 90C4BD70 98488E9C 219A7372 4EFFD6FA E5644738 FAA31A4F F55BCCC0 A151AF5F 0DC8B4BD 45BF37DF 365C1A65 E68CFDA7 6D4DA708 DF1FB2BC 2E4A4371
子群生成元*g*	A4D1CBD5 C3FD3412 6765A442 EFB99905 F8104DD2 58AC507F D6406CFF 14266D31 266FEA1E 5C41564B 777E690F 5504F213 160217B4 B01B886A 5E91547F 9E2749F4 D7FBD7D3 B9A92EE1 909D0D22 63F80A76 A6A24C08 7A091F53 1DBF0A01 69B6A28A D662A4D1 8E73AFA3 2D779D59 18D08BC8 858F4DCE F97C2A24 855E6EEB 22B3B2E5
g 的素数阶*q*	F518AA87 81A8DF27 8ABA4E7D 64B7CB9D 49462353

第一节介绍了PACEInfo。随后，列出并审查了包含生成的所有随机数和临时密钥的经交换的应用协议数据单元。

DH参数

由数据结构PACEInfo给出PACE的相关信息。

PACEInfo	3012060A 04007F00 07020204 01020201 02020100

PACEInfo的详细结构为：

标记	长度	值	ASN.1类型	注	
30	12		序列	PACEInfo	
06	0A	04 00 7F 00 07 02 02 04 01 02	客体标识符		客体标识符：利用椭圆曲线密钥交换协议、通用映射和AES 128会话密钥的PACE
02	01	02	整数		第2版
02	01	00	整数		RFC 5114规定的标准化1024位群

基于DH的实例的应用流程

为了初始化PACE，终端发送命令MSE:AT到芯片。

T>C :	00 22 C1 A4 0F 80 0A 04 00 7F 00 07 02 02 04 01 02 83 01 01
C>T :	90 00

下表说明了命令编码。

命令				
CLA	00		明文	
INS	22		管理安全环境	
P1/P2	C1 A4		设定相互认证的认证模板	
Lc	0F		数据域长度	
数据	标记	长度	值	注
	80	0A	04 00 7F 00 07 02 02 04 01 02	客体标识符：密码机制：利用ECDH密钥交换协议、通用映射和AES 128会话密钥的PACE
	83	01	01	口令：MRZ
响应				
状态字节	90 00		正常操作	

加密随机数

下一步，终端询问一个芯片的随机数。

解密后得到的随机数s	FA5B7E3E 49753A0D B9178B7B 9BD898C8
加密后得到的随机数z	854D8DF5 827FA685 2D1A4FA7 01CDDDCA

通信如下所示。

T>C :	10 86 00 00 02 7C 00 00
C>T :	7C 12 80 10 85 4D 8D F5 82 7F A6 85 2D 1A 4F A7 01 CD DD CA 90 00

下表说明了命令APDU和响应的编码。

命令				
CLA	10		命令链	
INS	86		GENERAL AUTHENTICATE	
P1/P2	00 00		默认的密钥和协议	
Lc	02		数据长度	
数据	标记	长度	值	注
	7C	00	-	缺失
Le	00		响应数据域的预期最大字节长度是256	
响应				
数据	标记	长度	值	注
	7C	12		动态认证数据
	80	10	85 4D 8D F5 82 7F A6 85 2D 1A 4F A7 01 CD DD CA	加密随机数
状态字节	90 00		正常操作	

映射随机数

通过通用映射，随机数被映射到一个临时的群生成元。为此，由终端和芯片随机生成下列临时密钥。

终端的私钥	5265030F 751F4AD1 8B08AC56 5FC7AC95 2E41618D
终端的公钥	23FB3749 EA030D2A 25B278D2 A562047A DE3F01B7 4F17A154 02CB7352 CA7D2B3E B71C343D B13D1DEB CE9A3666 DBCFC920 B49174A6 02CB4796 5CAA73DC 702489A4 4D41DB91 4DE9613D C5E98C94 160551C0 DF86274B 9359BC04 90D01B03 AD54022D CB4F57FA D6322497 D7A1E28D 46710F46 1AFE710F BBBC5F8B A166F431 1975EC6C
芯片的私钥	66DDAFEA C1609CB5 B963BB0C B3FF8B3E 047F336C
芯片的公钥	78879F57 225AA808 0D52ED0F C890A4B2 5336F699 AA89A2D3 A189654A F70729E6 23EA5738 B26381E4 DA19E004 706FACE7 B235C2DB F2F38748 312F3C98 C2DD4882 A41947B3 24AA1259 AC22579D B93F7085 655AF308 89DBB845 D9E6783F E42C9F24 49400306 254C8AE8 EE9DD812 A804C0B6 6E8CAFC1 4F84D825 8950A91B 44126EE6
共享秘密H	5BABEBEF 5B74E5BA 94B5C063 FDA15F1F 1CDE9487 3EE0A5D3 A2FCAB49 F258D07F 544F13CB 66658C3A FEE9E727 389BE3F6 CBBBD321 28A8C21D D6EEA3CF 7091CDDF B08B8D00 7D40318D CCA4FFBF 51208790 FB4BD111 E5A968ED 6B6F08B2 6CA87C41 0B3CE0C3 10CE104E ABD16629 AA48620C 1279270C B0750C0D 37C57FFF E302AE7F
映射所得生成元$\hat{G}$	7C9CBFE9 8F9FBDDA 8D143506 FA7D9306 F4CB17E3 C71707AF F5E1C1A1 23702496 84D64EE3 7AF44B8D BD9D45BF 6023919C BAA027AB 97ACC771 666C8E98 FF483301 BFA4872D EDE9034E DFACB708 14166B7F 36067682 9B826BEA 57291B5A D69FBC84 EF1E7790 32A30580 3F743417 93E86974 2D401325 B37EE856 5FFCDEE6 18342DC5

由终端和芯片交换下列APDU以映射随机数。

T>C:	10 86 00 00 86 7C 81 83 81 81 80 23 FB 37 49 EA 03 0D 2A 25 B2 78 D2 A5 62 04 7A DE 3F 01 B7 4F 17 A1 54 02 CB 73 52 CA 7D 2B 3E B7 1C 34 3D B1 3D 1D EB CE 9A 36 66 DB CF C9 20 B4 91 74 A6 02 CB 47 96 5C AA 73 DC 70 24 89 A4 4D 41 DB 91 4D E9 61 3D C5 E9 8C 94 16 05 51 C0 DF 86 27 4B 93 59 BC 04 90 D0 1B 03 AD 54 02 2D CB 4F 57 FA D6 32 24 97 D7 A1 E2 8D 46 71 0F 46 1A FE 71 0F BB BC 5F 8B A1 66 F4 31 19 75 EC 6C 00
C>T:	7C 81 83 82 81 80 78 87 9F 57 22 5A A8 08 0D 52 ED 0F C8 90 A4 B2 53 36 F6 99 AA 89 A2 D3 A1 89 65 4A F7 07 29 E6 23 EA 57 38 B2 63 81 E4 DA 1 9E0 04 70 6F AC E7 B2 35 C2 DB F2 F3 87 48 31 2F 3C 98 C2 DD 48 82 A4 19 47 B3 24 AA 12 59 AC 22 57 9D B9 3F 70 85 65 5A F3 08 89 DB B8 45 D9 E6 78 3F E4 2C 9F 24 49 40 03 06 25 4C 8A E8 EE 9D D8 12 A8 04 C0 B6 6E 8C AF C1 4F 84 D8 25 89 50 A9 1B 44 12 6E E6 90 00

应用协议数据单元结构可描述如下：

命令				
CLA	10		命令链	
INS	86		GENERAL AUTHENTICATE	
P1/P2	00 00		默认的密钥和协议	
Lc	86		数据长度	
数据	标记	长度	值	注
	7C	81 83	-	动态认证数据
	81	81 80	23 FB 37 49 EA 03 ... 75 EC 6C	映射数据
Le	00		响应数据域的预期最大字节长度为256	

响应				
数据	标记	长度	值	注
	7C	81 83		动态认证数据
	82	81 80	ED 0F C8 90 A4 B2 ... 12 6E E6	映射数据
状态字节	90 00		正常操作	

执行密钥协商

随后，芯片和终端通过使用前一步骤临时群生成元确定的新的域参数，执行一个匿名DH密钥协商。

终端的私钥	89CCD99B 0E8D3B1F 11E1296D CA68EC53 411CF2CA
终端的公钥	00907D89 E2D425A1 78AA81AF 4A7774EC 8E388C11 5CAE6703 1E85EECE 520BD911 551B9AE4 D04369F2 9A02626C 86FBC674 7CC7BC35 2645B616 1A2A42D4 4EDA80A0 8FA8D61B 76D3A154 AD8A5A51 786B0BC0 71470578 71A92221 2C5F67F4 31731722 36B7747D 1671E6D6 92A3C7D4 0A0C3C5C E397545D 015C175E B5130551 EDBC2EE5 D4
芯片的私钥	A5B78012 6B7C980E 9FCEA1D4 539DA1D2 7C342DFA
芯片的公钥	075693D9 AE941877 573E634B 6E644F8E 60AF17A0 076B8B12 3D920107 4D36152B D8B3A213 F53820C4 2ADC79AB 5D0AEEC3 AEFB9139 4DA476BD 97B9B14D 0A65C1FC 71A0E019 CB08AF55 E1F72900 5FBA7E3F A5DC4189 9238A250 767A6D46 DB974064 386CD456 743585F8 E5D90CC8 B4004B1F 6D866C79 CE0584E4 9687FF61 BC29AEA1
共享秘密	6BABC7B3 A72BCD7E A385E4C6 2DB2625B D8613B24 149E146A 629311C4 CA6698E3 8B834B6A 9E9CD718 4BA8834A FF5043D4 36950C4C 1E783236 7C10CB8C 314D40E5 990B0DF7 013E64B4 549E2270 923D06F0 8CFF6BD3 E977DDE6 ABE4C31D 55C0FA2E 465E553E 77BDF75E 3193D383 4FC26E8E B1EE2FA1 E4FC97C1 8C3F6CFF FE2607FD

密钥协商执行如下：

T>C:	10 86 00 00 86 7C 81 83 83 81 80 90 7D 89 E2 D4 25 A1 78 AA 81 AF 4A 77 74 EC 8E 38 8C 11 5C AE 67 03 1E 85 EE CE 52 0B D9 11 55 1B 9A E4 D0 43 69 F2 9A 02 62 6C 86 FB C6 74 7C C7 BC 35 26 45 B6 16 1A 2A 42 D4 4E DA 80 A0 8F A8 D6 1B 76 D3 A1 54 AD 8A 5A 51 78 6B 0B C0 71 47 05 78 71 A9 22 21 2C 5F 67 F4 31 73 17 22 36 B7 74 7D 16 71 E6 D6 92 A3 C7 D4 0A 0C 3C 5C E3 97 54 5D 01 5C 17 5E B5 13 05 51 ED BC 2E E5 D4 00
C>T:	7C 81 83 84 81 80 07 56 93 D9 AE 94 18 77 57 3E 63 4B 6E 64 4F 8E 60 AF 17 A0 07 6B 8B 12 3D 92 01 07 4D 36 15 2B D8 B3 A2 13 F5 38 20 C4 2A DC 79 AB 5D 0A EE C3 AE FB 91 39 4D A4 76 BD 97 B9 B1 4D 0A 65 C1 FC 71 A0 E0 19 CB 08 AF 55 E1 F7 29 00 5F BA 7E 3F A5 DC 41 89 92 38 A2 50 76 7A 6D 46 DB 97 40 64 38 6C D4 56 74 35 85 F8 E5 D9 0C C8 B4 00 4B 1F 6D 86 6C 79 CE 05 84 E4 96 87 FF 61 BC 29 AE A1 90 00

命令				
CLA	10		命令链	
INS	86		GENERAL AUTHENTICATE	
P1/P2	00 00		默认的密钥和协议	
Lc	86		数据长度	
数据	标记	长度	值	注
	7C	81 83	-	动态认证数据
	83	81 80	90 7D 89 E2 D4 25 ... 2E E5 D4	终端的临时公钥
Le	00		响应数据域的预期最大字节长度为256	

响应				
数据	标记	长度	值	注
	7C	81 83		动态认证数据
	84	81 80	07 56 93 D9 AE 94 ... 29 AE A1	芯片的临时公钥
状态字节	90 00		正常操作	

通过利用密钥导出函数从共享秘密派生AES 128会话密钥KS_{Enc}和KS_{MAC}。

KS_{Enc}	2F7F46AD CC9E7E52 1B45D192 FAFA9126
KS_{MAC}	805A1D27 D45A5116 F73C5446 9462B7D8

相互认证

从下列输入数据构建认证令牌。

T_{IFD}的输入数据	7F49818F 060A0400 7F000702 02040102 84818007 5693D9AE 94187757 3E634B6E 644F8E60 AF17A007 6B8B123D 9201074D 36152BD8 B3A213F5 3820C42A DC79AB5D 0AEEC3AE FB91394D A476BD97 B9B14D0A 65C1FC71 A0E019CB 08AF55E1 F729005F BA7E3FA5 DC418992 38A25076 7A6D46DB 97406438 6CD45674 3585F8E5 D90CC8B4 004B1F6D 866C79CE 0584E496 87FF61BC 29AEA1
T_{IC}的输入数据	7F49818F 060A0400 7F000702 02040102 84818090 7D89E2D4 25A178AA 81AF4A77 74EC8E38 8C115CAE 67031E85 EECE520B D911551B 9AE4D043 69F29A02 626C86FB C6747CC7 BC352645 B6161A2A 42D44EDA 80A08FA8 D61B76D3 A154AD8A 5A51786B 0BC07147 057871A9 22212C5F 67F43173 172236B7 747D1671 E6D692A3 C7D40A0C 3C5CE397 545D015C 175EB513 0551EDBC 2EE5D4

输入的数据编码显示如下：

标记	长度	值	ASN.1类型	注	
7F49	81 8F		公钥	T_{IFD}的输入数据	
06	0A	04 00 7F 00 07 02 02 04 01 02	客体标识符		利用ECDH密钥交换协议、通用映射和AES 128会话密钥的PACE
84	81 80	07 56 93 D9 AE ... 29 AE A1	无符号整数		芯片的临时公钥

标记	长度	值	ASN.1类型	注	
7F49	81 8F		公钥	T_{IC}的输入数据	
06	0A	04 00 7F 00 07 02 02 04 01 02	客体标识符		利用ECDH密钥交换协议、通用映射和AES 128会话密钥的PACE
84	81 80	90 7D 89 E2 D4 ... 2E E5 D4	无符号整数		终端的临时公钥

所计算的认证令牌为：

T_{IFD}	B46DD9BD 4D98381F
T_{IC}	917F37B5 C0E6D8D1

最后，交换并验证这些令牌。

T>C：	00 86 00 00 0C 7C 0A 85 08 B4 6D D9 BD 4D 98 38 1F 00
C>T：	7C 1B 86 08 91 7F 37 B5 C0 E6 D8 D1 87 0F 44 45 54 45 53 54 43 56 43 41 30 30 30 30 33

命令				
CLA	00		明文	
INS	86		GENERAL AUTHENTICATE	
P1/P2	00 00		默认的密钥和协议	
Lc	0C		数据长度	
数据	标记	长度	值	注
	7C	0A	-	动态认证数据
	85	08	B4 6D D9 BD 4D 98 38 1F	终端的认证令牌
Le	00		响应数据域的预期最大字节长度为256	
响应				
数据	标记	长度	值	注
	7C	0A		动态认证数据
	86	08	91 7F 37 B5 C0 E6 D8 D1	芯片的认证令牌
状态字节	90 00		正常运行	

— — — — — — — —

第11部分附录H

实例：口令认证连接确立（PACE）— 合成映射
（资料性）

本附录提供关于利用合成映射的PACE协议的两个实例。第一个实例基于椭圆曲线密钥交换协议（ECDH），第二个实例基于DH密钥交换协议（DH）。使用来自之前实例的由机读区派生的密钥*K*。

H.1 基于ECDH的实例

本实例基于BrainpoolP256r1椭圆曲线。本实例中使用的分组密码是AES-128。提示一下，曲线参数如下：

素数p	A9FB57DB A1EEA9BC 3E660A90 9D838D72 6E3BF623 D5262028 2013481D 1F6E5377
参数a	7D5A0975 FC2C3057 EEF67530 417AFFE7 FB8055C1 26DC5C6C E94A4B44 F330B5D9
参数b	26DC5C6C E94A4B44 F330B5D9 BBD77CBF 95841629 5CF7E1CE 6BCCDC18 FF8C07B6
群生成元G的x坐标	8BD2AEB9 CB7E57CB 2C4B482F FC81B7AF B9DE27E1 E3BD23C2 3A4453BD 9ACE3262
群生成元G的y坐标	547EF835 C3DAC4FD 97F8461A 14611DC9 C2774513 2DED8E54 5C1D54C7 2F046997
群阶n	A9FB57DB A1EEA9BC 3E660A90 9D838D71 8C397AA3 B561A6F7 901E0E82 974856A7
余因子f	01

加密密钥如下：

K_π	591468CD A83D6521 9CCCB856 0233600F

加密随机数

芯片选择随机数s，并使用K_π进行加密，随后将加密得到的随机数z发送到终端。

解密后得到的随机数s	2923BE84 E16CD6AE 529049F1 F1BBE9EB
加密后得到的随机数z	143DC40C 08C8E891 FBED7DED B92B64AD

映射随机数

随机选择一个随机数t并用明文发送。随后使用t和s计算合成映射。首先，将从AES派生的伪随机函数R_p运用到s和t。随后，根据结果使用点编码f_G来计算映射的生成元$\hat{G}=f_G(R_p(s,t))$。

随机数t	5DD4CBFC 96F5453B 130D890A 1CDBAE32
伪随机R(s,t)	E4447E2D FB3586BA C05DDB00 156B57FB B2179A39 49294C97 25418980 0C517BAA 8DA0FF39 7ED8C445 D3E421E4 FEB57322
$R_p(s,t)$	A2F8FF2D F50E52C6 599F386A DCB595D2 29F6A167 ADE2BE5F 2C3296AD D5B7430E
映射的生成元$\hat{G}$ 的x坐标	8E82D315 59ED0FDE 92A4D049 8ADD3C23 BABA94FB 77691E31 E90AEA77 FB17D427
映射的生成元$\hat{G}$的y坐标	4C1AE14B D0C3DBAC 0C871B7F 36081693 64437CA3 0AC243A0 89D3F266 C1E60FAD

执行密钥协商

芯片和终端通过使用它们的密钥和映射的生成元$\hat{G}$，执行一个匿名DH密钥协商。共享秘密K是协商的x坐标。

芯的私钥SK_{IC}	107CF586 96EF6155 053340FD 633392BA 81909DF7 B9706F22 6F32086C 7AFF974A
芯片的公钥PK_{IC}	67F78E5F 7F768608 2B293E8D 087E0569 16D0F74B C01A5F89 57D0DE45 691E51E8 932B69A9 62B52A09 85AD2C0A 271EE6A1 3A8ADDDC D1A3A994 B9DED257 F4D22753
终端的私钥SK_{IFD}	A73FB703 AC1436A1 8E0CFA5A BB3F7BEC 7A070E7A 6788486B EE230C4A 22762595
终端的公钥PK_{IFD}	89CBA23F FE96AA18 D824627C 3E934E54 A9FD0B87 A95D1471 DC1C0ABF DCD640D4 6755DE9B 7B778280 B6BEBD57 439ADFEB 0E21FD4E D6DF4257 8C13418A 59B34C37

共享秘密K	4F150FDE 1D4F0E38 E95017B8 91BAE171 33A0DF45 B0D3E18B 60BA7BEA FDC2C713

利用规范格式[1]，通过使用散列函数SHA-1：K_{Enc}=SHA-1(K||0x00000001)和K_{MAC}=SHA-1(K||0x00000002)，从K导派生出会话密钥K_{Enc}和K_{MAC}。然后，仅使用散列的前16个八位字节得到下列结果，：

K_{Enc}	0D3FEB33 251A6370 893D62AE 8DAAF51B
K_{MAC}	B01E89E3 D9E8719E 586B50B4 A7506E0B

相互认证

在下述输入数据和密钥K_{MAC}的基础上使用CMAC来计算认证令牌。

T_{IC}的输入数据	7F494F06 0A04007F 00070202 04040286 410489CB A23FFE96 AA18D824 627C3E93 4E54A9FD 0B87A95D 1471DC1C 0ABFDCD6 40D46755 DE9B7B77 8280B6BE BD57439A DFEB0E21 FD4ED6DF 42578C13 418A59B3 4C37
T_{IFD}的输入数据	7F494F06 0A04007F 00070202 04040286 410467F7 8E5F7F76 86082B29 3E8D087E 056916D0 F74BC01A 5F8957D0 DE45691E 51E8932B 69A962B5 2A0985AD 2C0A271E E6A13A8A DDDCD1A3 A994B9DE D257F4D2 2753

相应的认证令牌为：

T_{IC}	75D4D96E 8D5B0308
T_{IFD}	450F02B8 6F6A0909

H.2 基于DH的实例

本实例基于带有160位素数阶子群的1 024位MODP群。用于本实例的分组密码是AES-128。

群参数是：

素数p	B10B8F96 A080E01D DE92DE5E AE5D54EC 52C99FBC FB06A3C6 9A6A9DCA 52D23B61 6073E286 75A23D18 9838EF1E 2EE652C0 13ECB4AE A9061123 24975C3C D49B83BF ACCBDD7D 90C4BD70 98488E9C 219A7372 4EFFD6FA E5644738 FAA31A4F F55BCCC0 A151AF5F 0DC8B4BD 45BF37DF 365C1A65 E68CFDA7 6D4DA708 DF1FB2BC 2E4A4371
子群生成元g	A4D1CBD5 C3FD3412 6765A442 EFB99905 F8104DD2 58AC507F D6406CFF 14266D31 266FEA1E 5C41564B 777E690F 5504F213 160217B4 B01B886A 5E91547F 9E2749F4 D7FBD7D3 B9A92EE1 909D0D22 63F80A76 A6A24C08 7A091F53 1DBF0A01 69B6A28A D662A4D1 8E73AFA3 2D779D59 18D08BC8 858F4DCE F97C2A24 855E6EEB 22B3B2E5
g的素数阶q	F518AA87 81A8DF27 8ABA4E7D 64B7CB9D 49462353

使用下列加密密钥：

K_π	591468CD A83D6521 9CCCB856 0233600F

加密随机数

芯片选择随机数s，并使用K_π进行加密，随后将加密得到的随机数z发送到终端。

解密随机数s	FA5B7E3E 49753A0D B9178B7B 9BD898C8
加密随机数z	9ABB8864 CA0FF155 1E620D1E F4E13510

映射随机数

随机选择一个随机数t并用明文发送。随后使用t和s计算合成映射。首先，将从AES派生的伪随机函数Rp运用到s和t。随后，根据结果使用点编码f_g。

随机数t	B3A6DB3C 870C3E99 245E0D1C 06B747DE
伪随机R(s,t)	EAB98D13 E0905295 2AA72990 7C3C9461 84DEA0FE 74AD2B3A F506F0A8 3018459C 38099CD1 F7FF4EA0 A078DB1F AC136550 5E3DC855 00EF95E2 0B4EEF2E 88489233 BEE0546B 472F994B 618D1687 02406791 DEEF3CB4 810932EC 278F3533 FDB860EB 4835C36F A4F1BF3F A0B828A7 18C96BDE 88FBA38A 3E6C35AA A1095925 1EB5FC71 0FC18725 8995944C 0F926E24 9373F485
$R_p(s,t)$	A0C7C50C 002061A5 1CC87D25 4EF38068 607417B6 EE1B3647 3CFB800D 2D2E5FA2 B6980F01 105D24FA B22ACD1B FA5C8A4C 093ECDFA FE6D7125 D42A843E 33860383 5CF19AFA FF75EFE2 1DC5F6AA 1F9AE46C 25087E73 68166FB0 8C1E4627 AFED7D93 570417B7 90FF7F74 7E57F432 B04E1236 819E0DFE F5B6E77C A4999925 328182D2
映射所得的生成元$\hat{g}= f_g(R_p(s,t))$	1D7D767F 11E333BC D6DBAEF4 0E799E7A 926B9697 3550656F F3C83072 6D118D61 C276CDCC 61D475CF 03A98E0C 0E79CAEB A5BE2557 8BD4551D 0B109032 36F0B0F9 76852FA7 8EEA14EA 0ACA87D1 E91F688F E0DFF897 BBE35A47 2621D343 564B262F 34223AE8 FC59B664 BFEDFA2B FE7516CA 5510A6BB B633D517 EC25D4E0 BBAA16C2

执行密钥协商

芯片和终端使用其密钥和映射生成元$\hat{g}$执行一个匿名DH密钥协商。

芯片的私钥SK_{IC}	020F018C 7284B047 FA7721A3 37EFB7AC B1440BB3 0C5252BD 41C97C30 C994BB78 E9F0C5B3 2744D840 17D21FFA 6878396A 6469CA28 3EF5C000 DAF7D261 A39AB886 0ED4610A B5343390 897AAB5A 7787E4FA EFA0649C 6A94FDF8 2D991E8E 3FC332F5 142729E7 040A3F7D 5A4D3CD7 5CBEE1F0 43C1CAD2 DD484FEB 4ED22B59 7D36688E

芯片的公钥PK_{IC}	928D9A0F 9DBA450F 13FC859C 6F290D1D 36E42431 138A4378 500BEB4E 0401854C FF111F71 CB6DC1D0 335807A1 1388CC8E AA87B079 07AAD9FB A6B169AF 6D8C26AF 8DDDC39A DC3AD2E3 FF882B84 D23E9768 E95A80E4 746FB07A 9767679F E92133B4 D379935C 771BD7FB ED6C7BB4 B1708B27 5EA75679 524CDC9C 6A91370C C662A2F3
终端的私钥SK_{IFD}	4BD0E547 40F9A028 E6A515BF DAF96784 8C4F5F5F FF65AA09 15947FFD 1A0DF2FA 6981271B C905F355 1457B7E0 3AC3B806 6DE4AA40 6C1171FB 43DD939C 4BA16175 103BA3DE E16419AA 248118F9 0CC36A3D 6F4C3736 52E0C3CC E7F0F1D0 C5425B36 00F0F0D6 A67F004C 8BBA33F2 B4733C72 52445C1D FC4F1107 203F71D2 EFB28161
终端的公钥PK_{IFD}	0F0CC629 45A80292 51FB7EF3 C094E12E C68E4EF0 7F27CB9D 9CD04C5C 4250FAE0 E4F8A951 557E929A EB48E5C6 DD47F2F5 CD7C351A 9BD2CD72 2C07EDE1 66770F08 FFCB3702 62CF308D D7B07F2E 0DA9CAAA 1492344C 85290691 9538C98A 4BA4187E 76CE9D87 832386D3 19CE2E04 3C3343AE AE6EDBA1 A9894DC5 094D22F7 FE1351D5
共享秘密K	419410D6 C0A17A4C 07C54872 CE1CBCEB 0A2705C1 A434C8A8 9A4CFE41 F1D78124 CA7EC52B DE7615E5 345E48AB 1ABB6E7D 1D59A57F 3174084D 3CA45703 97C1F622 28BDFDB2 DA191EA2 239E2C06 0DBE3BBC 23C2FCD0 AF12E0F9 E0B99FCF 91FF1959 011D5798 B2FCBC1F 14FCC24E 441F4C8F 9B08D977 E9498560 E63E7FFA B3134EA7

通过使用散列函数SHA-1: K_{Enc}=SHA-1(K||0x00000001)和K_{MAC}=SHA-1(K||0x00000002)，从K中导出会话密钥K_{Enc}和K_{MAC}。然后，仅使用散列的前16个八位字节得出如下结果：

K_{Enc}	01AFC10C F87BE36D 8179E873 70171F07
K_{MAC}	23F0FBD0 5FD6C7B8 B88F4C83 09669061

相互认证

在下述输入数据和密钥K_{MAC}的基础上使用CMAC来计算认证令牌。

T_{IC}的输入数据	7F49818F 060A0400 7F000702 02040302 8481800F 0CC62945 A8029251 FB7EF3C0 94E12EC6 8E4EF07F 27CB9D9C D04C5C42 50FAE0E4 F8A95155 7E929AEB 48E5C6DD 47F2F5CD 7C351A9B D2CD722C 07EDE166 770F08FF CB370262 CF308DD7 B07F2E0D A9CAAA14 92344C85 29069195 38C98A4B A4187E76 CE9D8783 2386D319 CE2E043C 3343AEAE 6EDBA1A9 894DC509 4D22F7FE 1351D5
T_{IFD}的输入数据	7F49818F 060A0400 7F000702 02040302 84818092 8D9A0F9D BA450F13 FC859C6F 290D1D36 E4243113 8A437850 0BEB4E04 01854CFF 111F71CB 6DC1D033 5807A113 88CC8EAA 87B07907 AAD9FBA6 B169AF6D 8C26AF8D DDC39ADC 3AD2E3FF 882B84D2 3E9768E9 5A80E474 6FB07A97 67679FE9 2133B4D3 79935C77 1BD7FBED 6C7BB4B1 708B275E A7567952 4CDC9C6A 91370CC6 62A2F3

相应的认证令牌为：

T_{IC}	C2F04230 187E1525
T_{IFD}	55D61977 CBF5307E

— — — — — — — — —

第 11 部分　附录 I

实例：口令认证连接确立（PACE）— PACE 芯片认证映射

（资料性）

本附录提供了基于椭圆曲线密钥交换协议（ECDH）使用芯片认证映射的 PACE 协议的实例。表中包含的所有数字用十六进制表示。

机读区 MRZ 被用作口令。包括校验数位在内的机读区相关数据域是:

- 证件号：C11T002JM4；
- 出生日期：9608122；
- 截至日期：2310314。

因此，机读区编码 K 和派生加密密钥 K_π 为

K	894D03F1 48C6265E 89845B21 8856EA34 D00EF8E8
K_π	4E6F6FBF 7BE748B9 32C7B741 61BBA9DF

I.1　基于椭圆曲线密钥交换协议（ECDH）的实例

本实例基于应用标准化BrainpoolP256r1域参数的ECDH（见[RFC 5639]）。

第一节介绍相应的PACEInfo。随后列举并审查所交换的APDU应用协议数据单元，包括生成的所有随机数和临时密钥。

椭圆曲线参数

通过使用标准化域参数，数据结构 PACEInfo 给出执行 PACE 需要的全部信息。特别是，不需要 PACEDomainParameterInfo。

PACEInfo	3012060A 04007F00 07020204 06020201 0202010D

下表逐项列出了 PACEInfo 的具体结构。

标记	长度	值	ASN.1类型	注
30	12		序列	PACEInfo
06	0A	04 00 7F 00 07 02 02 04 06 02	客体标识符	利用ECDH密钥交换协议、芯片认证映射和AES 128会话密钥的PACE
02	01	02	整数	第2版
02	01	0D	整数	Brainpool P256r1标准化域参数

为方便起见，下表给出了 BrainpoolP256r1 域参数的 ASN.1 编码。

标记	长度	值	ASN.1类型	注
30	81 EC		序列	域参数
06	07	2A 86 48 CE 3D 02 01	客体标识符	算法 id-ecPublicKey
30	81 E0		序列	域参数
02	01	01	整数	版本
30	2C		序列	基础域
06	07	2A 86 48 CE 3D 01 01	客体标识符	素数域
02	21	00 A9 FB 57 DB A1 EE A9 BC 3E 66 0A 90 9D 83 8D 72 6E 3B F6 23 D5 26 20 28 20 13 48 1D 1F 6E 53 77	整数	素数 p
30	44		序列	曲线方程
04	20	7D 5A 09 75 FC 2C 30 57 EE F6 75 30 41 7A FF E7 FB 80 55 C1 26 DC 5C 6C E9 4A 4B 44 F3 30 B5 D9	八位字节串	参数 a
04	20	26 DC 5C 6C E9 4A 4B 44 F3 30 B5 D9 BB D7 7C BF 95 84 16 29 5C F7 E1 CE 6B CC DC 18 FF 8C 07 B6	序列	参数 b

标记	长度	值	ASN.1类型	注
04	41		八位字节串	群发生器 G
		04	-	未压缩点
		8B D2 AE B9 CB 7E 57 CB 2C 4B 48 2F FC 81 B7 AF B9 DE 27 E1 E3 BD 23 C2 3A 44 53 BD 9A CE 32 62	-	x 坐标
		54 7E F8 35 C3 DA C4 FD 97 F8 46 1A 14 61 1D C9 C2 77 45 13 2D ED 8E 54 5C 1D 54 C7 2F 04 69 97	-	y 坐标
02	21	00 A9 FB 57 DB A1 EE A9 BC 3E 66 0A 90 9D 83 8D 71 8C 39 7A A3 B5 61 A6 F7 90 1E 0E 82 97 48 56 A7	整数	群序 n
02	01	01	整数	余因子 f

基于ECDH实例的应用流程

为了初始化PACE，终端发送命令MSE:AT至芯片。

T>C：	`00 22 C1 A4 0F 80 0A 04 00 7F 00 07 02 02 04 06 02 83 01 01`
C>T：	`90 00`

此处 T>C 是从终端发送给芯片的 APDU 的缩写，C>T 表示从芯片发送给终端的响应。命令编码如下表所示。

命令				
CLA	00		明文	
INS	22		管理安全环境	
P1/P2	C1 A4		设定相互认证的认证模板	
Lc	0F		数据域的长度	
数据	标记	长度	值	注
	80	0A	04 00 7F 00 07 02 02 04 06 02	密码机制：利用 ECDH 密钥交换协议、芯片认证映射和 AES 128 会话密钥的 PACE
	83	01	01	口令：MRZ

响应		
状态字节	90 00	正常操作

加密随机数

然后，芯片随机生成随机数，并通过K_π将其加密。

解密后得到的随机数s	658B860B C94DF6F0 44FCE6D5 C82CF8E5
加密后得到的随机数z	CB60E8E0 D85B76A9 BD304747 C2AD42E2

终端询问加密随机数。

T>C：	10 86 00 00 02 7C 00 00
C>T：	7C 12 80 10 CB 60 E8 E0 D8 5B 76 A9 BD 30 47 47 C2 AD 42 E2 90 00

命令 APDU 和相应响应的编码见下表。

命令				
CLA	10		命令链	
INS	86		GENERAL AUTHENTICATE	
P1/P2	00 00		默认的密钥和协议	
Lc	02		数据长度	
数据	标记	长度	值	注
	7C	00	-	缺失
Le	00		响应数据域的预期最大字节长度是 256	
响应				
数据	标记	长度	值	注
	7C	12		动态认证数据
	80	10	CB60E8E0 D85B76A9 BD304747 C2AD42E2	加密随机数
状态字节	90 00		正常操作	

映射随机数

随机数通过通用映射被映射到一个临时群发生器。下表也收集了所需的随机选择的临时密钥。

终端的私钥	5D8BB87B D74D985A 4B7D4325 B9F7B976 FE835122 77340079 8914AA22 738135CC
终端的公钥	7F1D410A DB7DDB3B 84BF1030 800981A9 105D7457 B4A3ADE0 02384F30 86C67EDE 1AB88910 4A27DB6D 842B0190 20FBF3CE ACB0DC62 7F7BDCAC 29969E19 D0E553C1
芯片的私钥	9E56A6B5 9C95D06E CE5CD10F 983BB2F4 F1943528 E577F238 81D89D8C 3BBEE0AA
芯片的公钥	A234236A A9B9621E 8EFB73B5 245C0E09 D2576E52 77183C12 08BDD552 80CAE8B3 04F36571 3A356E65 A451E165 ECC9AC0A C46E3771 342C8FE5 AEDD0926 85338E23
共享秘密H	2C1DCC17 73346492 C6636A36 EE4B965E 292E9AAE 7EE37736 EF58B9D0 A043F348 403A8CF3 3CA7DC0D 9DF61D08 89CE2442 4FF97C1A AD48A5CA 2A554B07 1EF7638D
映射所得生成元$\hat{G}$	89F0B5EA BF3BE293 C75903A3 98613192 5C9F5B51 5CA95AF4 85DC7E88 6F03245D 44BEFB2D D3A0DBD7 1CB5E618 971CF474 7F12B79E 548379A4 0E45963B AAF3E829

终端和芯片交换下列 APDU，以映射随机数。

T>C :	10 86 00 00 45 7C 43 81 41 04 7F 1D 41 0A DB 7D DB 3B 84 BF 10 30 80 09 81 A9 10 5D 74 57 B4 A3 AD E0 02 38 4F 30 86 C6 7E DE 1A B8 89 10 4A 27 DB 6D 84 2B 01 90 20 FB F3 CE AC B0 DC 62 7F 7B DC AC 29 96 9E 19 D0 E5 53 C1 00
C>T :	7C 43 82 41 04 A2 34 23 6A A9 B9 62 1E 8E FB 73 B5 24 5C 0E 09 D2 57 6E 52 77 18 3C 12 08 BD D5 52 80 CA E8 B3 04 F3 65 71 3A 35 6E 65 A4 51 E1 65 EC C9 AC 0A C4 6E 37 71 34 2C 8F E5 AE DD 09 26 85 33 8E 23 90 00

APDU 的结构可说明如下：

命令					
CLA	10		命令链		
INS	86		GENERAL AUTHENTICATE		
P1/P2	00 00		默认的密钥和协议		
Lc	45		数据长度		
数据	标记	长度	值	注	
	7C	43	-	动态认证数据	
	81	41		映射数据	
			04		未压缩点
			7F 1D 41 0A ... 86 C6 7E DE		x坐标
			1A B8 89 10... D0 E5 53 C1		y坐标
Le	00		响应数据域的预期最大字节长度是 256		
响应					
数据	标记	长度	值	注	
	7C	43		动态认证数据	
	82	41		映射数据	
			04		未压缩点
			A2 34 23 6A ... 80 CA E8 B3		x坐标
			04 F3 65 71... 85 33 8E 23		y坐标
状态字节	90 00		正常运行		

执行密钥协商

在第三步，芯片和终端通过使用前一步的临时群发生器确定的新的域参数，执行一个匿名 ECDH 密钥交换协议的密钥协商。仅需要 x 坐标作为共享秘密，因为密钥导出函数（KDF）仅使用第一个坐标派生会话密钥。

终端的私钥	76ECFDAA 9841C323 A3F5FC5E 88B88DB3 EFF7E35E BF57A7E6 946CB630 006C2120
终端的公钥	446C9340 84D9DAB8 63944F21 9520076C 29EE3F7A E6722B11 FF319EC1 C7728F95 5483400B FF60BF0C 59292700 09277DC2 A515E125 75010AD9 BA916CF1 BF86FEFC
芯片的私钥	CD626EF3 C256E235 FE8912CA C28279E6 26008EDA 6B3A05C4 CF862A3B DAB79E78
芯片的公钥	02AD566F 3C6EC7F9 324509AD 50A51FA5 2030782A 4968FCFE DF737DAE A9933331 11C3B9B4 C2287789 BD137E7F 8AA882E2 A3C633CC D6ECC2C6 3C57AD40 1A09C2E1
共享秘密	67950559 D0C06B4D 4B86972D 14460837 461087F8 419FDBC3 6AAF6CEA AC462832

密钥协商执行如下：

T>C：	10 86 00 00 45 7C 43 83 41 04 44 6C 93 40 84 D9 DA B8 63 94 4F 21 95 20 07 6C 29 EE 3F 7A E6 72 2B 11 FF 31 9E C1 C7 72 8F 95 54 83 40 0B FF 60 BF 0C 59 29 27 00 09 27 7D C2 A5 15 E1 25 75 01 0A D9 BA 91 6C F1 BF 86 FE FC 00
C>T：	7C 43 84 41 04 02 AD 56 6F 3C 6E C7 F9 32 45 09 AD 50 A5 1F A5 20 30 78 2A 49 68 FC FE DF 73 7D AE A9 93 33 31 11 C3 B9 B4 C2 28 77 89 BD 13 7E 7F 8A A8 82 E2 A3 C6 33 CC D6 EC C2 C6 3C 57 AD 40 1A 09 C2 E1 90 00

通过下表审查密钥协商编码：

命令				
CLA	10		命令链	
INS	86		GENERAL AUTHENTICATE	
P1/P2	00 00		默认的密钥和协议	
Lc	45		数据长度	
数据	标记	长度	值	注
	7C	43	-	动态认证数据
	83	41		终端的临时公钥
			04	未压缩点

			44 6C 93 40 ... C7 72 8F 95		x坐标
			54 83 40 0B ... BF 86 FE FC		y坐标
Le	00		响应数据域的预期最大字节长度为 256		
响应					
数据	标记	长度	值	注	
	7C	43		动态认证数据	
	84	41		芯片的临时公钥	
			04		未压缩点
			02 AD 56 6F ... A9 93 33 31		x坐标
			11 C3 B9 B4 ... 1A 09 C2 E1		y坐标
状态字节	90 00		正常运行		

通过密钥导出函数（KDF），从共享秘密派生 AES 128 会话密钥 KS_{Enc} 和 KS_{MAC}。分别为：

KS_{Enc}	0A9DA4DB 03BDDE39 FC5202BC 44B2E89E
KS_{MAC}	4B1C0649 1ED5140C A2B537D3 44C6C0B1

相互认证

利用下列输入数据，通过 KS_{MAC} 派生出认证令牌：

T_{IFD}的输入数据	7F494F06 0A04007F 00070202 04060286 410402AD 566F3C6E C7F93245 09AD50A5 1FA52030 782A4968 FCFEDF73 7DAEA993 333111C3 B9B4C228 7789BD13 7E7F8AA8 82E2A3C6 33CCD6EC C2C63C57 AD401A09 C2E1
T_{IC}的输入数据	7F494F06 0A04007F 00070202 04060286 4104446C 934084D9 DAB86394 4F219520 076C29EE 3F7AE672 2B11FF31 9EC1C772 8F955483 400BFF60 BF0C5929 27000927 7DC2A515 E1257501 0AD9BA91 6CF1BF86 FEFC

输入数据的编码如下所示：

标记	长度	值	ASN.1类型	注
7F49	4F		公钥	T_{IFD} 的输入数据
06	0A	04 00 7F 00 07 02 02 04 06 02	客体标识符	利用 ECDH 密钥交换协议、芯片认证映射和 AES 128 会话密钥的 PACE
86	41		椭圆曲线点	芯片的临时公共点
		04		未压缩点
		02 AD 56 6F... A9 93 33 31		x 坐标
		11 C3 B9 B4 ... 1A 09 C2 E1		y 坐标

标记	长度	值	ASN.1类型	注
7F49	4F		公钥	T_{IC} 的输入数据
06	0A	04 00 7F 00 07 02 02 04 06 02	客体标识符	利用 ECDH 密钥交换协议、芯片认证映射和 AES 128 会话密钥的 PACE
86	41		椭圆曲线点	终端的临时公共点
		04		未压缩点
		44 6C 93 40 ... C7 72 8F 95		x 坐标
		54 83 40 0B ... BF 86 FE FC		y 坐标

所计算的认证令牌是：

T_{IFD}	E86BD060 18A1CD3B
T_{IC}	8596CF05 5C67C1A3

最后，交换并验证这些令牌。

T>C :	00 86 00 00 0C 7C 0A 85 08 E8 6B D0 60 18 A1 CD 3B 00
C>T :	7C 3C 86 08 85 96 CF 05 5C 67 C1 A3 8A 30 1E EA 96 4D AA E3 72 AC 99 0E 3E FD E6 33 33 53 BF C8 9A 67 04 D9 3D A8 79 8C F7 7F 5B 7A 54 BD 10 CB A3 72 B4 2B E0 B9 B5 F2 8A A8 DE 2F 4F 92 90 00

通过下表审查相互认证编码：

命令					
CLA	00		无命令链（链中的最后一个命令）		
INS	86		GENERAL AUTHENTICATE		
P1/P2	00 00		默认的密钥和协议		
Lc	0C		数据长度		
数据	标记	长度	值	注	
	7C	0A	-	动态认证数据	
	85	08		终端的认证令牌	
			E8 6B D0 60 18 A1 CD 3B		T_{IFD}
Le	00		响应数据域的预期最大字节长度为 256		
响应					
数据	标记	长度	值	注	
	7C	3C		动态认证数据	
	86	08		芯片的认证令牌	
			85 96 CF 05 5C 67 C1 A3		T_{IC}
	8A	30			x 坐标
			1E EA 96 4D … DE 2F 4F 92		加密的芯片认证数据
状态字节	90 00		正常运行		

芯片认证

从 EF.CardSecurity 获得 `ChipAuthenticationPublicKeyInfo`

ChipAuthenticationPublicKeyInfo	`30620609 04007F00 07020201 02305230`
	`0C060704 007F0007 01020201 0D034200`
	`04187270 9494399E 7470A643 1BE25E83`
	`EEE24FEA 568C2ED2 8DB48E05 DB3A610D`
	`C884D256 A40E35EF CB59BF67 53D3A489`
	`D28C7A4D 973C2DA1 38A6E7A4 A08F68E1`
	`6F02010D`

下表逐项列出了 ChipAuthenticationPublicKeyInfo 具体结构。

标记	长度	值	ASN.1类型	注	
30	62		序列	ChipAuthenticationPublicKeyInfo	
06	09	04 00 7F 00 07 02 02 01 02	客体标识符		id-PK-ECDH
30	52		序列		SubjectPublicKeyInfo
30	0C		序列		Brainpool P256r1 标准化域参数
06	07	04 00 7F 00 07 01 02	客体标识符		标准化域参数
02	01	0D	整数		Brainpool256r1
03	42	00 04 18 72 70 … 8F 68 E1 6F	字符串		芯片认证公钥
02	01	0D	整数		keyID 13

下列数据用于芯片认证：

加密的芯片认证数据	1EEA964D AAE372AC 990E3EFD E6333353 BFC89A67 04D93DA8 798CF77F 5B7A54BD 10CBA372 B42BE0B9 B5F28AA8 DE2F4F92
解密的芯片认证数据	85DC3FA9 3D0952BF A82F5FD1 89EE75BD 82F11D1F 0B8ED4BF 5319AC9B 53C426B3
用于加密/解密芯片数据的 IV $IV = E(KS_{ENC}, -1)$	F6A3B75A1 E933941 DD7A13E2 520779DF
通过 GENERAL AUTHENTICATE 映射随机数得出的芯片公钥 $PK_{MAP,IC}$	A234236A A9B9621E 8EFB73B5 245C0E09 D2576E52 77183C12 08BDD552 80CAE8B3 04F36571 3A356E65 A451E165 ECC9AC0A C46E3771 342C8FE5 AEDD0926 85338E23
通过 ChipAuthenticationPublicKeyInfo 得出的芯片的芯片认证公钥 PK_{IC}	18727094 94399E74 70A6431B E25E83EE E24FEA56 8C2ED28D B48E05DB 3A610DC8 84D256A4 0E35EFCB 59BF6753 D3A489D2 8C7A4D97 3C2DA138 A6E7A4A0 8F68E16F

终端验证 $PK_{MAP,IC} = KA(CA_{IC}, PK_{IC}, D_{IC})$

— — — — — — — —

第 11 部分附录 J

查验程序（资料性）

J.1 电子机读旅行证件应用程序的查验程序

本节仅载列电子机读旅行证件应用程序（“LDS1 证件”）的查验程序。

1. 访问非接触式集成电路（参见第 4.2 节）

   - 如果对集成电路的访问受到保护，则可在此步骤中使用 PACE 或 BAC，但出于安全原因建议使用 PACE。从 2018 年 1 月 1 日开始，电子机读旅行证件可能仅支持 PACE。
   - 如果集成电路和终端支持，出于性能原因应使用 PACE-CAM。
   - 集成电路允许访问电子机读旅行证件应用程序中不太敏感的数据和主文件中的 EF.CardSecurity（如果存在）。

2. 开始数据认证

   - 阅读证件安全对象并验证签名，包括证件签名者证书的链验证。

3. 芯片认证

   - 根据集成电路的支持方式，执行芯片认证或主动认证。电子机读旅行证件应用程序中 EF.DG15 的存在表明对主动认证的支持，EF.DG14 中存在相应的 SecurityInfos 表明对芯片认证的支持。
   - 如果使用有芯片认证映射的 PACE，此步骤也可作为步骤 1 的一部分执行。
   - 只有与包含用于此步骤的公钥（EF.CardSecurity、EF.DG14 或 EF.DG15）的文件的认证结合才能完成认证。

4. 额外的访问控制

   - 如果电子机读旅行证件被配置为要求访问敏感数据，即 EF.DG3 和/或 EF.DG4，则必须执行终端认证。

5. 读取数据

   - 一旦授予必要的访问权限，即可开始读取数据，例如，在步骤 1 后可以读取不太敏感的数据。
   - 读取的数据未经验证，不得将数据视为真实数据（步骤 2）。

J.2 多用途电子机读旅行证件的查验程序

本节说明为电子机读旅行证件设计的查验程序，除了电子机读旅行证件应用程序（“LDS2 文件”）外，还包含一个或多个应用。此程序也可仅访问电子机读旅行证件应用程序。

1. 访问非接触式集成电路（参见第 4.2 节）

   - 在此设置中，只有 PACE 可用于访问集成电路。
   - 如果集成电路和终端支持，出于性能原因应使用 PACE-CAM。
   - 集成电路允许访问电子机读旅行证件应用程序中不太敏感的数据和主文件中的 EF.CardSecurity。

2. 检查 EF.CardSecurity 是否存在

   - 如果 EF.CardSecurity 不存在，则电子机读旅行证件不支持主文件中的认证（暗示集成电路仅包含电子机读旅行证件应用程序）。在这种情况下，选择电子机读旅行证件应用程序并继续执行本附录 J.1 节中程序的第 2 步。

3. 开始数据认证

   - 读取 EF.CardSecurity 并验证签名，包括证件签名者证书的链验证。
   - 来自电子机读旅行证件应用程序的数据通过证件安全对象进行保护，在读取来自该应用的数据时，必须对其进行验证。来自其他应用的数据受到数据签名的保护，在读取这些数据时也必须对其进行验证。

4. 芯片认证

   - 在主文件执行芯片认证。如果 EF.CardSecurity 中的 SecurityInfos 没有包含必要的信息，则集成电路不支持主文件的认证。在这种情况下，选择电子机读旅行证件应用程序并继续执行本附录 J.1 节中程序的第 2 步。
   - 如果使用有芯片认证映射的 PACE，此步骤也可作为步骤 1 的一部分执行。
   - 只有与包含用于此步骤的公钥（EF.CardSecurity）的文件的认证结合才能完成认证。

5. 额外的访问控制

   - 执行终端认证
   - 如果只需要对电子机读旅行证件应用程序中不太敏感的数据进行读取访问，则可以跳过此步骤。

6. 读/写数据

- 读/写数据包括选择包含文件的应用。
- 一旦授予必要的访问权限，即可开始读取数据，例如，在步骤 1 后，可以读取电子机读旅行证件应用程序不太敏感的数据。
- 读取的数据未经验证，不得将数据视为真实数据（步骤 3）。

— — — — — — — —

第 11 部分附录 K

欧洲扩展访问控制（资料性）

本文件定义的终端认证基于欧盟使用的扩展访问控制（参见[TR-03110]），以保护对存储在 LDS1 应用程序中的指纹的访问。本附录指出[TR-03110]与本文件定义的协议之间的差异。

根据[TR-03110]，用于访问配备 EAC 的电子机读旅行证件的高级查验程序包括以下步骤：

1. 执行芯片访问程序（见第 4.2 节）并选择电子机读旅行证件应用程序；

2. 在电子机读旅行证件应用程序中进行芯片认证（见第 6.2 节）并启动被动认证（见第 5.1 节）；

3. 在电子机读旅行证件应用程序中执行终端认证（见下文）（见第 7.1 节）。

注：芯片和终端认证均在欧洲扩展访问控制的电子机读旅行证件应用程序中执行。本文件中的规范允许根据上下文在电子机读旅行证件应用程序或主文件中执行这些协议。

K.1 访问权

表 K-1. 查验系统的授权

7	6	5	4	3	2	1	0	说明
x	x	-	-	-	-	-	-	作用（见第 Doc 9303-12 号文件）
-	-	x	x	x	x	x	x	访问权
-	-	x	x	x	x	-	-	RFU
-	-	-	-	-	-	1	-	读取访问电子机读旅行证件应用程序：DG3（虹膜）
-	-	-	-	-	-	-	1	读取访问电子机读旅行证件应用程序：DG3（指纹）

电子机读旅行证件应用程序以外的应用程序中数据组的访问权限通过 Doc9303 号文件第 12 部分和第 10 部分定义的授权扩展传送。指纹（和虹膜）的访问权限通过证书持有人授权模板传送：

有关有效访问权限的计算，请参阅第 7.1.4.3.6 节。

K.2 EF.CVCA

根据规范，集成电路作为终端认证的一部分用于证书验证的已知信任点（Certificate Authority References）作为 PACE 协议的一部分传输到 IFD（见第 4.4.3.5 节）。

欧洲扩展访问控制在电子机读旅行证件应用程序中定义了一个透明文件 EF.CVCA。其规范转载如下：

表 K-2. 基本文件 EF.CVCA

文件名称	EF.CVCA
文件编号	0x011C（默认）
短文件编号	0x1C（(默认)
读取权限	PACE
写入权限	从不（仅用于内部更新）
大小	用值为 0x00 的 8 位字节填充的 36 个字节（固定）
内容	［CARi］[||CARi-1］[||0x00..00］

如表 K-2 所述，如果集成电路支持电子机读旅行证件应用程序中的终端认证，则必须在电子机读旅行证件应用程序中的透明基本文件 EF.CVCA 中提及适用于查验系统的 CVCA 公钥。

这个文件应包含一系列适用于终端认证的认证机构参考（CAR）数据对象（见 Doc 9303-12 号文件）。

- 它应包含至多两个认证机构参考数据对象。
- 最近的认证机构参考应该是这个列表中的第一个数据对象。
- 必须通过数字为 0x00 的 8 位字节来填充文件。

文件 EF.CVCA 具有默认 EF 标识符和短 EF 标识符。如果不能使用默认值，则（短）EF 标识符应在 `TerminalAuthenticationInfo` 的可选参数 `efCVCA` 中指定。

如果 `efCVCA` 用于指示要使用的 EF 标识符，则覆盖默认的 EF 标识符。如果在 `efCVCA` 中没有给出短 EF 标识符，则必须使用给定的 EF 标识符明确选择文件 EF.CVCA。

```
TerminalAuthenticationInfo ::= SEQUENCE {
  protocol OBJECT IDENTIFIER(id-TA),
  version  INTEGER, -- MUST be 1
  efCVCA   FileID OPTIONAL
}

FileID ::= SEQUENCE {
  fid  OCTET STRING (SIZE(2)),
  sfid OCTET STRING (SIZE(1)) OPTIONAL
}
```

—完—

ISBN 978-92-9265-525-9